Red Hat Enterprise Linux 9

Surveillance et gestion de l'état et des performances du système

Optimisation du débit, de la latence et de la consommation d'énergie du système

Last Updated: 2024-02-22

Red Hat Enterprise Linux 9 Surveillance et gestion de l'état et des performances du système

Optimisation du débit, de la latence et de la consommation d'énergie du système

Notice légale

Résumé

Surveiller et optimiser le débit, la latence et la consommation d'énergie de Red Hat Enterprise Linux 9 dans différents scénarios.

Table des matières

RENDRE L'OPEN SOURCE PLUS INCLUSIF 10

FOURNIR UN RETOUR D'INFORMATION SUR LA DOCUMENTATION DE RED HAT 11

CHAPITRE 1. DÉMARRER AVEC TUNED 12
1.1. L'OBJECTIF DE TUNED 12
1.2. PROFILS TUNED 12
1.3. LE PROFIL TUNED PAR DÉFAUT 13
1.4. PROFILS TUNED FUSIONNÉS 13
1.5. EMPLACEMENT DES PROFILS TUNED 14
1.6. PROFILS TUNED DISTRIBUÉS AVEC RHEL 14
1.7. PROFIL DE PARTITIONNEMENT DU PROCESSEUR TUNED 17
1.8. UTILISATION DU PROFIL DE PARTITIONNEMENT DU PROCESSEUR TUNED POUR UN RÉGLAGE À FAIBLE LATENCE 18
1.9. PERSONNALISATION DU PROFIL TUNED DE PARTITIONNEMENT DU PROCESSEUR 19
1.10. PROFILS TUNED EN TEMPS RÉEL DISTRIBUÉS AVEC RHEL 19
1.11. ACCORD STATIQUE ET DYNAMIQUE DANS TUNED 20
1.12. MODE TUNED SANS DÉMON 21
1.13. INSTALLATION ET ACTIVATION DE TUNED 21
1.14. LISTE DES PROFILS TUNED DISPONIBLES 22
1.15. DÉFINITION D'UN PROFIL TUNED 22
1.16. DÉSACTIVATION DE TUNED 24

CHAPITRE 2. PERSONNALISATION DES PROFILS TUNED 25
2.1. PROFILS TUNED 25
2.2. LE PROFIL TUNED PAR DÉFAUT 25
2.3. PROFILS TUNED FUSIONNÉS 26
2.4. EMPLACEMENT DES PROFILS TUNED 26
2.5. HÉRITAGE ENTRE LES PROFILS TUNED 27
2.6. ACCORD STATIQUE ET DYNAMIQUE DANS TUNED 28
2.7. PLUG-INS TUNED 28
2.8. PLUG-INS TUNED DISPONIBLES 30
2.9. VARIABLES DANS LES PROFILS TUNED 34
2.10. FONCTIONS INTÉGRÉES DANS LES PROFILS TUNED 35
2.11. FONCTIONS INTÉGRÉES DISPONIBLES DANS LES PROFILS TUNED 36
2.12. CRÉATION DE NOUVEAUX PROFILS TUNED 37
2.13. MODIFIER LES PROFILS TUNED EXISTANTS 38
2.14. PARAMÉTRAGE DU PLANIFICATEUR DE DISQUE À L'AIDE DE TUNED 39

CHAPITRE 3. RÉVISION D'UN SYSTÈME À L'AIDE DE L'INTERFACE TUNA 42
3.1. INSTALLATION DE L'OUTIL THON 42
3.2. VISUALISATION DE L'ÉTAT DU SYSTÈME À L'AIDE DE L'OUTIL TUNA 42
3.3. OPTIMISATION DES PROCESSEURS À L'AIDE DE L'OUTIL TUNA 43
3.4. RÉGLAGE DES IRQ À L'AIDE DE L'OUTIL TUNA 45

CHAPITRE 4. SURVEILLANCE DES PERFORMANCES À L'AIDE DES RÔLES SYSTÈME RHEL 47
4.1. PRÉPARATION D'UN NŒUD DE CONTRÔLE ET DE NŒUDS GÉRÉS À L'UTILISATION DES RÔLES SYSTÈME RHEL 47
4.2. INTRODUCTION AU RÔLE DU SYSTÈME METRICS 52
4.3. UTILISATION DU RÔLE DE SYSTÈME METRICS POUR SURVEILLER VOTRE SYSTÈME LOCAL AVEC VISUALISATION 53
4.4. L'UTILISATION DU RÔLE DE SYSTÈME METRICS PERMET DE CONFIGURER UN PARC DE SYSTÈMES INDIVIDUELS POUR QU'ILS SE SURVEILLENT EUX-MÊMES 54

4.5. UTILISATION DU RÔLE DE SYSTÈME METRICS POUR SURVEILLER UN PARC DE MACHINES DE MANIÈRE CENTRALISÉE VIA VOTRE MACHINE LOCALE 55
4.6. CONFIGURATION DE L'AUTHENTIFICATION LORS DE LA SURVEILLANCE D'UN SYSTÈME À L'AIDE DU RÔLE DE SYSTÈME METRICS 56
4.7. UTILISATION DU RÔLE DE SYSTÈME METRICS POUR CONFIGURER ET ACTIVER LA COLLECTE DE MÉTRIQUES POUR SQL SERVER 57

CHAPITRE 5. MISE EN PLACE DU PCP 59
5.1. VUE D'ENSEMBLE DU PCP 59
5.2. INSTALLATION ET ACTIVATION DE PCP 59
5.3. DÉPLOYER UNE CONFIGURATION PCP MINIMALE 60
5.4. SERVICES SYSTÈME DISTRIBUÉS AVEC PCP 61
5.5. OUTILS DISTRIBUÉS AVEC LE PCP 62
5.6. ARCHITECTURES DE DÉPLOIEMENT PCP 65
5.7. ARCHITECTURE DE DÉPLOIEMENT RECOMMANDÉE 68
5.8. FACTEURS DE DIMENSIONNEMENT 68
5.9. OPTIONS DE CONFIGURATION POUR LA MISE À L'ÉCHELLE DU PCP 69
5.10. EXEMPLE : ANALYSE DU DÉPLOIEMENT DE LA JOURNALISATION CENTRALISÉE 70
5.11. EXEMPLE : ANALYSE DU DÉPLOIEMENT DE L'INSTALLATION FÉDÉRÉE 71
5.12. ÉTABLIR DES CONNEXIONS PCP SÉCURISÉES 71
5.13. DÉPANNAGE EN CAS D'UTILISATION ÉLEVÉE DE LA MÉMOIRE 74

CHAPITRE 6. ENREGISTREMENT DES DONNÉES DE PERFORMANCE AVEC PMLOGGER 77
6.1. MODIFIER LE FICHIER DE CONFIGURATION DE PMLOGGER AVEC PMLOGCONF 77
6.2. MODIFIER MANUELLEMENT LE FICHIER DE CONFIGURATION DE PMLOGGER 77
6.3. ACTIVATION DU SERVICE PMLOGGER 78
6.4. MISE EN PLACE D'UN SYSTÈME CLIENT POUR LA COLLECTE DE DONNÉES 79
6.5. MISE EN PLACE D'UN SERVEUR CENTRAL POUR LA COLLECTE DES DONNÉES 80
6.6. REPRODUIRE LES ARCHIVES DES JOURNAUX PCP AVEC PMREP 82
6.7. ACTIVATION DES ARCHIVES PCP VERSION 3 83

CHAPITRE 7. SUIVI DES PERFORMANCES AVEC PERFORMANCE CO-PILOT 85
7.1. SURVEILLANCE DE POSTFIX AVEC PMDA-POSTFIX 85
7.2. TRACER VISUELLEMENT LES ARCHIVES DES JOURNAUX PCP AVEC L'APPLICATION PCP CHARTS 86
7.3. COLLECTE DE DONNÉES À PARTIR D'UN SERVEUR SQL À L'AIDE DE PCP 88
7.4. GÉNÉRER DES ARCHIVES PCP À PARTIR D'ARCHIVES SADC 90

CHAPITRE 8. ANALYSE DES PERFORMANCES DE XFS AVEC PCP 92
8.1. INSTALLATION MANUELLE DE XFS PMDA 92
8.2. EXAMEN DES PERFORMANCES DE XFS AVEC PMINFO 93
8.3. RÉINITIALISATION DES MESURES DE PERFORMANCE XFS AVEC PMSTORE 94
8.4. GROUPES DE MÉTRIQUES PCP POUR XFS 95
8.5. GROUPES DE MÉTRIQUES PCP PAR PÉRIPHÉRIQUE POUR XFS 96

CHAPITRE 9. MISE EN PLACE D'UNE REPRÉSENTATION GRAPHIQUE DES MESURES PCP 99
9.1. MISE EN PLACE DE PCP AVEC PCP-ZEROCONF 99
9.2. MISE EN PLACE D'UN SERVEUR GRAFANA 99
9.3. ACCÉDER À L'INTERFACE WEB DE GRAFANA 100
9.4. CONFIGURER DES CONNEXIONS SÉCURISÉES POUR GRAFANA 102
9.5. CONFIGURATION DE PCP REDIS 103
9.6. CONFIGURER DES CONNEXIONS SÉCURISÉES POUR PCP REDIS 104
9.7. CRÉATION DE PANNEAUX ET D'ALERTES DANS LA SOURCE DE DONNÉES REDIS DE PCP 105
9.8. AJOUT DE CANAUX DE NOTIFICATION POUR LES ALERTES 108
9.9. MISE EN PLACE DE L'AUTHENTIFICATION ENTRE LES COMPOSANTS DU PCP 109
9.10. INSTALLATION DE PCP BPFTRACE 110

9.11. VISUALISATION DU TABLEAU DE BORD D'ANALYSE DU SYSTÈME PCP BPFTRACE 111
9.12. INSTALLATION DE PCP VECTOR 112
9.13. VISUALISATION DE LA LISTE DE CONTRÔLE DU VECTEUR PCP 113
9.14. RÉSOLUTION DES PROBLÈMES LIÉS À GRAFANA 114

CHAPITRE 10. OPTIMISER LES PERFORMANCES DU SYSTÈME À L'AIDE DE LA CONSOLE WEB 117
10.1. OPTIONS DE RÉGLAGE DES PERFORMANCES DANS LA CONSOLE WEB 117
10.2. DÉFINITION D'UN PROFIL DE PERFORMANCE DANS LA CONSOLE WEB 117
10.3. CONTRÔLE DES PERFORMANCES SUR LE SYSTÈME LOCAL À L'AIDE DE LA CONSOLE WEB 118
10.4. SURVEILLANCE DES PERFORMANCES SUR PLUSIEURS SYSTÈMES À L'AIDE DE LA CONSOLE WEB ET DE GRAFANA 119

CHAPITRE 11. PARAMÉTRAGE DU PLANIFICATEUR DE DISQUE 122
11.1. PLANIFICATEURS DE DISQUES DISPONIBLES 122
11.2. DIFFÉRENTS ORDONNANCEURS DE DISQUES POUR DIFFÉRENTS CAS D'UTILISATION 123
11.3. LE PLANIFICATEUR DE DISQUE PAR DÉFAUT 123
11.4. DÉTERMINATION DE L'ORDONNANCEUR DE DISQUE ACTIF 124
11.5. PARAMÉTRAGE DU PLANIFICATEUR DE DISQUE À L'AIDE DE TUNED 124
11.6. DÉFINITION DE L'ORDONNANCEUR DE DISQUE À L'AIDE DES RÈGLES UDEV 126
11.7. DÉFINITION TEMPORAIRE D'UN PLANIFICATEUR POUR UN DISQUE SPÉCIFIQUE 127

CHAPITRE 12. OPTIMISER LES PERFORMANCES D'UN SERVEUR SAMBA 128
12.1. RÉGLAGE DE LA VERSION DU PROTOCOLE SMB 128
12.2. OPTIMISATION DES PARTAGES AVEC DES RÉPERTOIRES CONTENANT UN GRAND NOMBRE DE FICHIERS 128
12.3. PARAMÈTRES POUVANT AVOIR UN IMPACT NÉGATIF SUR LES PERFORMANCES 129

CHAPITRE 13. OPTIMIZING VIRTUAL MACHINE PERFORMANCE 130
13.1. WHAT INFLUENCES VIRTUAL MACHINE PERFORMANCE 130
13.2. OPTIMIZING VIRTUAL MACHINE PERFORMANCE USING TUNED 131
13.3. OPTIMIZING LIBVIRT DAEMONS 132
13.4. CONFIGURING VIRTUAL MACHINE MEMORY 134
13.5. OPTIMIZING VIRTUAL MACHINE I/O PERFORMANCE 138
13.6. OPTIMIZING VIRTUAL MACHINE CPU PERFORMANCE 140
13.7. OPTIMIZING VIRTUAL MACHINE NETWORK PERFORMANCE 152
13.8. VIRTUAL MACHINE PERFORMANCE MONITORING TOOLS 153
13.9. RESSOURCES SUPPLÉMENTAIRES 155

CHAPITRE 14. IMPORTANCE DE LA GESTION DE L'ÉNERGIE 156
14.1. PRINCIPES DE BASE DE LA GESTION DE L'ÉNERGIE 156
14.2. VUE D'ENSEMBLE DE L'AUDIT ET DE L'ANALYSE 157
14.3. OUTILS D'AUDIT 158

CHAPITRE 15. GÉRER LA CONSOMMATION D'ÉNERGIE AVEC POWERTOP 162
15.1. L'OBJECTIF DE POWERTOP 162
15.2. UTILISATION DE POWERTOP 162
15.3. STATISTIQUES POWERTOP 163
15.4. POURQUOI POWERTOP N'AFFICHE-T-IL PAS LES VALEURS DES STATISTIQUES DE FRÉQUENCE DANS CERTAINS CAS ? 165
15.5. GÉNÉRER UNE SORTIE HTML 166
15.6. OPTIMISER LA CONSOMMATION D'ÉNERGIE 166

CHAPITRE 16. DÉMARRER AVEC PERF 168
16.1. INTRODUCTION À LA PERF 168
16.2. INSTALLATION DE PERF 168

16.3. COMMANDES COURANTES DE PERF 168

CHAPITRE 17. PROFILAGE DE L'UTILISATION DE L'UNITÉ CENTRALE EN TEMPS RÉEL AVEC PERF TOP ... 170

17.1. L'OBJECTIF DE PERF TOP 170
17.2. PROFILAGE DE L'UTILISATION DE L'UNITÉ CENTRALE AVEC PERF TOP 170
17.3. INTERPRÉTATION DE LA SORTIE DE PERF TOP 171
17.4. POURQUOI PERF AFFICHE-T-IL CERTAINS NOMS DE FONCTIONS COMME DES ADRESSES DE FONCTIONS BRUTES ? 171
17.5. ACTIVATION DES DÉPÔTS DE DÉBOGAGE ET DE SOURCES 171
17.6. OBTENIR LES PAQUETS D'INFORMATIONS DE DÉBOGAGE POUR UNE APPLICATION OU UNE BIBLIOTHÈQUE À L'AIDE DE GDB 172

CHAPITRE 18. COMPTER LES ÉVÉNEMENTS PENDANT L'EXÉCUTION D'UN PROCESSUS AVEC PERF STAT 174

18.1. L'OBJECTIF DE PERF STAT 174
18.2. COMPTAGE DES ÉVÉNEMENTS AVEC PERF STAT 174
18.3. INTERPRÉTATION DE LA SORTIE DE L'ÉTAT DE PERF 175
18.4. ATTACHER LE STATUT DE PERF À UN PROCESSUS EN COURS D'EXÉCUTION 176

CHAPITRE 19. ENREGISTREMENT ET ANALYSE DES PROFILS DE PERFORMANCE AVEC PERF 177

19.1. L'OBJECTIF DE LA FICHE DE PERF 177
19.2. ENREGISTREMENT D'UN PROFIL DE PERFORMANCE SANS ACCÈS ROOT 177
19.3. ENREGISTREMENT D'UN PROFIL DE PERFORMANCE AVEC UN ACCÈS ROOT 177
19.4. ENREGISTREMENT D'UN PROFIL DE PERFORMANCE EN MODE PER-CPU 178
19.5. CAPTURER LES DONNÉES DU GRAPHIQUE D'APPEL AVEC L'ENREGISTREMENT DES PERFORMANCES 178
19.6. ANALYSE DE PERF.DATA AVEC PERF REPORT 179
19.7. INTERPRÉTATION DU RAPPORT DE PERF 180
19.8. GÉNÉRER UN FICHIER PERF.DATA LISIBLE SUR UN AUTRE APPAREIL 180
19.9. ANALYSE D'UN FICHIER PERF.DATA CRÉÉ SUR UN AUTRE APPAREIL 181
19.10. POURQUOI PERF AFFICHE-T-IL CERTAINS NOMS DE FONCTIONS COMME DES ADRESSES DE FONCTIONS BRUTES ? 182
19.11. ACTIVATION DES DÉPÔTS DE DÉBOGAGE ET DE SOURCES 182
19.12. OBTENIR LES PAQUETS D'INFORMATIONS DE DÉBOGAGE POUR UNE APPLICATION OU UNE BIBLIOTHÈQUE À L'AIDE DE GDB 183

CHAPITRE 20. ÉTUDIER LES UNITÉS CENTRALES OCCUPÉES À L'AIDE DE LA PERF 185

20.1. AFFICHAGE DES ÉVÉNEMENTS DE L'UNITÉ CENTRALE QUI ONT ÉTÉ COMPTABILISÉS AVEC PERF STAT 185
20.2. AFFICHAGE DE L'UNITÉ CENTRALE SUR LAQUELLE LES ÉCHANTILLONS ONT ÉTÉ PRÉLEVÉS AVEC LE RAPPORT DE PERF 185
20.3. AFFICHAGE D'UNITÉS CENTRALES SPÉCIFIQUES LORS DU PROFILAGE AVEC PERF TOP 186
20.4. SURVEILLANCE D'UNITÉS CENTRALES SPÉCIFIQUES GRÂCE À L'ENREGISTREMENT ET AU RAPPORT DE PERFORMANCE 186

CHAPITRE 21. CONTRÔLER LA PERFORMANCE DES APPLICATIONS AVEC PERF 188

21.1. ATTACHER UNE FICHE DE PERF À UN PROCESSUS EN COURS 188
21.2. CAPTURER LES DONNÉES DU GRAPHIQUE D'APPEL AVEC L'ENREGISTREMENT DES PERFORMANCES 188
21.3. ANALYSE DE PERF.DATA AVEC PERF REPORT 189

CHAPITRE 22. CRÉER DES ROBES DE CHAMBRE AVEC PERF 191

22.1. CRÉER DES ROBES DE CHAMBRE AU NIVEAU DE LA FONCTION AVEC PERF 191
22.2. CRÉER DES UPROBES SUR DES LIGNES DANS UNE FONCTION AVEC PERF 191
22.3. SORTIE D'UN SCRIPT PERF DES DONNÉES ENREGISTRÉES AU COURS DES SONDES ASCENDANTES

192

CHAPITRE 23. PROFILER LES ACCÈS À LA MÉMOIRE AVEC PERF MEM 194
23.1. L'OBJECTIF DE PERF MEM 194
23.2. ÉCHANTILLONNAGE DE L'ACCÈS À LA MÉMOIRE AVEC PERF MEM 194
23.3. INTERPRÉTATION DU RAPPORT DE PERF MEM 196

CHAPITRE 24. DÉTECTER LES FAUX PARTAGES 198
24.1. L'OBJECTIF DE PERF C2C 198
24.2. DÉTECTION DE LA CONTENTION DES LIGNES DE CACHE AVEC PERF C2C 198
24.3. VISUALISATION D'UN FICHIER PERF.DATA ENREGISTRÉ AVEC PERF C2C RECORD 199
24.4. INTERPRÉTATION DU RAPPORT DE PERF C2C 201
24.5. DÉTECTION DES FAUX PARTAGES AVEC PERF C2C 202

CHAPITRE 25. DÉMARRER AVEC FLAMEGRAPHS 205
25.1. INSTALLATION DE FLAMEGRAPHS 205
25.2. CRÉATION DE GRAPHES DE FLAMME SUR L'ENSEMBLE DU SYSTÈME 205
25.3. CRÉATION DE GRAPHES DE FLAMME SUR DES PROCESSUS SPÉCIFIQUES 206
25.4. INTERPRÉTATION DES DIAGRAMMES DE FLAMME 207

CHAPITRE 26. SURVEILLANCE DES PROCESSUS POUR DÉTECTER LES GOULETS D'ÉTRANGLEMENT AU NIVEAU DES PERFORMANCES À L'AIDE DES TAMPONS CIRCULAIRES DE PERF 209
26.1. TAMPONS CIRCULAIRES ET INSTANTANÉS SPÉCIFIQUES À UN ÉVÉNEMENT AVEC PERF 209
26.2. COLLECTE DE DONNÉES SPÉCIFIQUES POUR SURVEILLER LES GOULETS D'ÉTRANGLEMENT AU NIVEAU DES PERFORMANCES À L'AIDE DES TAMPONS CIRCULAIRES DE PERF 209

CHAPITRE 27. AJOUTER ET SUPPRIMER DES TRACEPOINTS D'UN COLLECTEUR DE PERF EN COURS D'EXÉCUTION SANS ARRÊTER OU REDÉMARRER PERF 211
27.1. AJOUTER DES TRACEPOINTS À UN COLLECTEUR PERF EN COURS D'EXÉCUTION SANS ARRÊTER OU REDÉMARRER PERF 211
27.2. SUPPRIMER LES TRACEPOINTS D'UN COLLECTEUR DE PERF EN COURS D'EXÉCUTION SANS ARRÊTER OU REDÉMARRER PERF 212

CHAPITRE 28. PROFILER L'ALLOCATION DE MÉMOIRE AVEC NUMASTAT 213
28.1. STATISTIQUES NUMASTAT PAR DÉFAUT 213
28.2. VISUALISATION DE L'ALLOCATION DE MÉMOIRE AVEC NUMASTAT 213

CHAPITRE 29. CONFIGURATION D'UN SYSTÈME D'EXPLOITATION POUR OPTIMISER L'UTILISATION DE L'UNITÉ CENTRALE 215
29.1. OUTILS DE SURVEILLANCE ET DE DIAGNOSTIC DES PROBLÈMES LIÉS AU PROCESSEUR 215
29.2. TYPES DE TOPOLOGIE DE SYSTÈME 216
29.3. CONFIGURATION DU TEMPS DE RÉPONSE DU NOYAU 218
29.4. APERÇU D'UNE DEMANDE D'INTERRUPTION 220

CHAPITRE 30. OPTIMISATION DE LA POLITIQUE D'ORDONNANCEMENT 223
30.1. CATÉGORIES DE POLITIQUES D'ORDONNANCEMENT 223
30.2. ORDONNANCEMENT STATIQUE DES PRIORITÉS AVEC SCHED_FIFO 223
30.3. ORDONNANCEMENT PRIORITAIRE À LA RONDE AVEC SCHED_RR 224
30.4. ORDONNANCEMENT NORMAL AVEC SCHED_OTHER 224
30.5. DÉFINITION DES RÈGLES DE L'ORDONNANCEUR 225
30.6. OPTIONS DE POLITIQUE POUR LA COMMANDE CHRT 226
30.7. MODIFIER LA PRIORITÉ DES SERVICES PENDANT LE PROCESSUS DE DÉMARRAGE 226
30.8. CARTE DES PRIORITÉS 228
30.9. PROFIL DE PARTITIONNEMENT DU PROCESSEUR TUNED 228
30.10. UTILISATION DU PROFIL DE PARTITIONNEMENT DU PROCESSEUR TUNED POUR UN RÉGLAGE À FAIBLE LATENCE 230

30.11. PERSONNALISATION DU PROFIL TUNED DE PARTITIONNEMENT DU PROCESSEUR 230

CHAPITRE 31. OPTIMISER LES PERFORMANCES DU RÉSEAU 232
31.1. RÉGLAGE DES PARAMÈTRES DE LA CARTE RÉSEAU 232
31.2. RÉGLAGE DE L'ÉQUILIBRAGE DES IRQ 236
31.3. AMÉLIORER LA LATENCE DU RÉSEAU 239
31.4. AMÉLIORER LE DÉBIT DE GRANDES QUANTITÉS DE FLUX DE DONNÉES CONTIGUS 243
31.5. OPTIMISATION DES CONNEXIONS TCP POUR UN DÉBIT ÉLEVÉ 246
31.6. OPTIMISATION DES CONNEXIONS UDP 251
31.7. IDENTIFIER LES GOULOTS D'ÉTRANGLEMENT DE LA MÉMOIRE TAMPON DE LA SOCKET DE LECTURE DE L'APPLICATION 257
31.8. OPTIMISATION DES APPLICATIONS AVEC UN GRAND NOMBRE DE REQUÊTES ENTRANTES 259
31.9. ÉVITER LES CONFLITS DE VERROUILLAGE DE LA FILE D'ATTENTE D'ÉCOUTE 260
31.10. OPTIMISATION DU PILOTE DE PÉRIPHÉRIQUE ET DE LA CARTE D'INTERFACE RÉSEAU 265
31.11. CONFIGURATION DES PARAMÈTRES DE DÉLESTAGE DE LA CARTE RÉSEAU 267
31.12. RÉGLAGE DE LA COALESCENCE D'INTERRUPTION 270
31.13. AVANTAGES DES HORODATAGES TCP 274
31.14. CONTRÔLE DE FLUX POUR LES RÉSEAUX ETHERNET 274

CHAPITRE 32. FACTEURS AFFECTANT LES PERFORMANCES DES E/S ET DU SYSTÈME DE FICHIERS 276
32.1. OUTILS DE SURVEILLANCE ET DE DIAGNOSTIC DES PROBLÈMES D'E/S ET DE SYSTÈME DE FICHIERS 276
32.2. OPTIONS DE RÉGLAGE DISPONIBLES POUR LE FORMATAGE D'UN SYSTÈME DE FICHIERS 278
32.3. OPTIONS DE RÉGLAGE DISPONIBLES POUR LE MONTAGE D'UN SYSTÈME DE FICHIERS 279
32.4. TYPES D'ÉLIMINATION DES BLOCS INUTILISÉS 280
32.5. CONSIDÉRATIONS SUR LA MISE AU POINT DES DISQUES À SEMI-CONDUCTEURS 281
32.6. PARAMÈTRES DE RÉGLAGE DU BLOC GÉNÉRIQUE 282

CHAPITRE 33. UTILISER SYSTEMD POUR GÉRER LES RESSOURCES UTILISÉES PAR LES APPLICATIONS .. 284
33.1. ALLOCATION DES RESSOURCES SYSTÈME À L'AIDE DE SYSTEMD 284
33.2. RÔLE DE SYSTEMD DANS LA GESTION DES RESSOURCES 285
33.3. VUE D'ENSEMBLE DE LA HIÉRARCHIE DE SYSTEMD POUR LES CGROUPS 285
33.4. LISTE DES UNITÉS SYSTEMD 287
33.5. VISUALISATION DE LA HIÉRARCHIE DES GROUPES DE CONTRÔLE DE SYSTEMD 288
33.6. VISUALISATION DES GROUPES DE PROCESSUS 290
33.7. CONTRÔLE DE LA CONSOMMATION DES RESSOURCES 291
33.8. UTILISATION DES FICHIERS UNITAIRES DE SYSTEMD POUR FIXER DES LIMITES AUX APPLICATIONS 292
33.9. UTILISATION DE LA COMMANDE SYSTEMCTL POUR FIXER DES LIMITES AUX APPLICATIONS 293
33.10. DÉFINITION DE L'AFFINITÉ PAR DÉFAUT DE L'UNITÉ CENTRALE PAR LE BIAIS DE LA CONFIGURATION DU GESTIONNAIRE 294
33.11. CONFIGURATION DES POLITIQUES NUMA À L'AIDE DE SYSTEMD 294
33.12. OPTIONS DE CONFIGURATION DE LA POLITIQUE NUMA POUR SYSTEMD 295
33.13. CRÉATION DE CGROUPS TRANSITOIRES À L'AIDE DE LA COMMANDE SYSTEMD-RUN 296
33.14. SUPPRESSION DES GROUPES DE CONTRÔLE TRANSITOIRES 297

CHAPITRE 34. COMPRENDRE LES CGROUPS 299
34.1. COMPRENDRE LES GROUPES DE CONTRÔLE 299
34.2. QUE SONT LES CONTRÔLEURS DE RESSOURCES DU NOYAU ? 300
34.3. QU'EST-CE QU'UN ESPACE DE NOMS ? 301

CHAPITRE 35. AMÉLIORER LES PERFORMANCES DU SYSTÈME AVEC ZSWAP 303
35.1. QU'EST-CE QUE ZSWAP ? 303
35.2. ACTIVATION DE ZSWAP AU MOMENT DE L'EXÉCUTION 303

35.3. ACTIVATION PERMANENTE DE ZSWAP 304

CHAPITRE 36. UTILISATION DE CGROUPFS POUR GÉRER MANUELLEMENT LES CGROUPS 305
36.1. CRÉATION DE CGROUPS ET ACTIVATION DE CONTRÔLEURS DANS LE SYSTÈME DE FICHIERS CGROUPS-V2 305
36.2. CONTRÔLE DE LA RÉPARTITION DU TEMPS D'UTILISATION DE L'UNITÉ CENTRALE POUR LES APPLICATIONS EN AJUSTANT LE POIDS DE L'UNITÉ CENTRALE 308
36.3. MONTAGE DE CGROUPS-V1 310
36.4. FIXER DES LIMITES DE CPU AUX APPLICATIONS EN UTILISANT CGROUPS-V1 312

CHAPITRE 37. ANALYSE DES PERFORMANCES DU SYSTÈME AVEC BPF COMPILER COLLECTION ... 316
37.1. INSTALLATION DU PAQUETAGE BCC-TOOLS 316
37.2. UTILISATION DE CERTAINS OUTILS BCC POUR L'ANALYSE DES PERFORMANCES 316

CHAPITRE 38. CONFIGURER UN SYSTÈME D'EXPLOITATION POUR OPTIMISER L'ACCÈS À LA MÉMOIRE . 321
38.1. OUTILS DE SURVEILLANCE ET DE DIAGNOSTIC DES PROBLÈMES DE MÉMOIRE DU SYSTÈME 321
38.2. VUE D'ENSEMBLE DE LA MÉMOIRE D'UN SYSTÈME 322
38.3. PARAMÈTRES DE LA MÉMOIRE VIRTUELLE 322
38.4. PARAMÈTRES DU SYSTÈME DE FICHIERS 325
38.5. PARAMÈTRES DU NOYAU 325
38.6. RÉGLAGE DES PARAMÈTRES DU NOYAU LIÉS À LA MÉMOIRE 326

CHAPITRE 39. CONFIGURATION DE PAGES VOLUMINEUSES 328
39.1. CARACTÉRISTIQUES DE L'IMMENSE PAGE DISPONIBLE 328
39.2. PARAMÈTRES DE RÉSERVATION DES PAGES DE LA HUGETLB AU MOMENT DU DÉMARRAGE 329
39.3. CONFIGURATION DE HUGETLB AU DÉMARRAGE 330
39.4. PARAMÈTRES DE RÉSERVATION DES PAGES HUGETLB AU MOMENT DE L'EXÉCUTION 331
39.5. CONFIGURATION DE HUGETLB AU MOMENT DE L'EXÉCUTION 332
39.6. PERMETTRE LA TRANSPARENCE DES IMAGES GÉANTES 333
39.7. DÉSACTIVATION DES PAGES DE GARDE TRANSPARENTES 334
39.8. IMPACT DE LA TAILLE DE LA PAGE SUR LA TAILLE DU TAMPON DE TRANSLATION (LOOKASIDE BUFFER) 334

CHAPITRE 40. DÉMARRER AVEC SYSTEMTAP 335
40.1. L'OBJECTIF DE SYSTEMTAP 335
40.2. INSTALLATION DE SYSTEMTAP 335
40.3. PRIVILÈGES POUR EXÉCUTER SYSTEMTAP 336
40.4. EXÉCUTION DES SCRIPTS SYSTEMTAP 337

CHAPITRE 41. INSTRUMENTATION CROISÉE DE SYSTEMTAP 338
41.1. INSTRUMENTATION CROISÉE SYSTEMTAP 338
41.2. INITIALISATION DE L'INSTRUMENTATION CROISÉE DE SYSTEMTAP 339

CHAPITRE 42. SURVEILLANCE DE L'ACTIVITÉ DU RÉSEAU AVEC SYSTEMTAP 341
42.1. PROFILAGE DE L'ACTIVITÉ DU RÉSEAU AVEC SYSTEMTAP 341
42.2. TRACER LES FONCTIONS APPELÉES DANS LE CODE D'UN SOCKET RÉSEAU AVEC SYSTEMTAP 342
42.3. SURVEILLANCE DES CHUTES DE PAQUETS SUR LE RÉSEAU AVEC SYSTEMTAP 343

CHAPITRE 43. PROFILER L'ACTIVITÉ DU NOYAU AVEC SYSTEMTAP 344
43.1. COMPTER LES APPELS DE FONCTION AVEC SYSTEMTAP 344
43.2. TRACER LES APPELS DE FONCTION AVEC SYSTEMTAP 345
43.3. DÉTERMINER LE TEMPS PASSÉ DANS LE NOYAU ET L'ESPACE UTILISATEUR AVEC SYSTEMTAP 346
43.4. SURVEILLANCE DES APPLICATIONS DE SONDAGE AVEC SYSTEMTAP 347
43.5. SUIVI DES APPELS SYSTÈME LES PLUS FRÉQUEMMENT UTILISÉS AVEC SYSTEMTAP 348
43.6. SUIVI DU VOLUME D'APPELS SYSTÈME PAR PROCESSUS AVEC SYSTEMTAP 349

CHAPITRE 44. SURVEILLANCE DE L'ACTIVITÉ DES DISQUES ET DES E/S AVEC SYSTEMTAP 350
44.1. SYNTHÈSE DU TRAFIC DE LECTURE/ÉCRITURE DES DISQUES AVEC SYSTEMTAP 350
44.2. SUIVI DU TEMPS D'E/S POUR CHAQUE LECTURE OU ÉCRITURE DE FICHIER AVEC SYSTEMTAP 351
44.3. SUIVI DES E/S CUMULATIVES AVEC SYSTEMTAP 352
44.4. SURVEILLANCE DE L'ACTIVITÉ E/S SUR UN APPAREIL SPÉCIFIQUE AVEC SYSTEMTAP 352
44.5. SURVEILLANCE DES LECTURES ET DES ÉCRITURES DANS UN FICHIER AVEC SYSTEMTAP 353

RENDRE L'OPEN SOURCE PLUS INCLUSIF

Red Hat s'engage à remplacer les termes problématiques dans son code, sa documentation et ses propriétés Web. Nous commençons par ces quatre termes : master, slave, blacklist et whitelist. En raison de l'ampleur de cette entreprise, ces changements seront mis en œuvre progressivement au cours de plusieurs versions à venir. Pour plus de détails, voir le message de notre directeur technique Chris Wright.

FOURNIR UN RETOUR D'INFORMATION SUR LA DOCUMENTATION DE RED HAT

Nous apprécions vos commentaires sur notre documentation. Faites-nous savoir comment nous pouvons l'améliorer.

Soumettre des commentaires sur des passages spécifiques

1. Consultez la documentation au format **Multi-page HTML** et assurez-vous que le bouton **Feedback** apparaît dans le coin supérieur droit après le chargement complet de la page.
2. Utilisez votre curseur pour mettre en évidence la partie du texte que vous souhaitez commenter.
3. Cliquez sur le bouton **Add Feedback** qui apparaît près du texte en surbrillance.
4. Ajoutez vos commentaires et cliquez sur **Submit**.

Soumettre des commentaires via Bugzilla (compte requis)

1. Connectez-vous au site Web de Bugzilla.
2. Sélectionnez la version correcte dans le menu **Version**.
3. Saisissez un titre descriptif dans le champ **Summary**.
4. Saisissez votre suggestion d'amélioration dans le champ **Description**. Incluez des liens vers les parties pertinentes de la documentation.
5. Cliquez sur **Submit Bug**.

CHAPITRE 1. DÉMARRER AVEC TUNED

En tant qu'administrateur système, vous pouvez utiliser l'application **TuneD** pour optimiser le profil de performance de votre système pour une variété de cas d'utilisation.

1.1. L'OBJECTIF DE TUNED

TuneD est un service qui surveille votre système et optimise les performances sous certaines charges de travail. Le cœur de **TuneD** est *profiles*, qui adapte votre système à différents cas d'utilisation.

TuneD est distribué avec un certain nombre de profils prédéfinis pour des cas d'utilisation tels que :

- Haut débit
- Faible latence
- Économie d'énergie

Il est possible de modifier les règles définies pour chaque profil et de personnaliser le réglage d'un appareil particulier. Lorsque vous passez à un autre profil ou que vous désactivez **TuneD**, toutes les modifications apportées aux paramètres du système par le profil précédent reviennent à leur état d'origine.

Vous pouvez également configurer **TuneD** pour qu'il réagisse aux changements d'utilisation des appareils et ajuste les paramètres afin d'améliorer les performances des appareils actifs et de réduire la consommation d'énergie des appareils inactifs.

1.2. PROFILS TUNED

L'analyse détaillée d'un système peut prendre beaucoup de temps. **TuneD** fournit un certain nombre de profils prédéfinis pour des cas d'utilisation typiques. Vous pouvez également créer, modifier et supprimer des profils.

Les profils fournis par **TuneD** sont répartis dans les catégories suivantes :

- Profils d'économie d'énergie
- Profils d'amélioration des performances

Les profils d'amélioration des performances comprennent des profils qui se concentrent sur les aspects suivants :

- Faible latence pour le stockage et le réseau
- Débit élevé pour le stockage et le réseau
- Performances de la machine virtuelle
- Performances des hôtes de virtualisation

Syntaxe de la configuration du profil

Le fichier **tuned.conf** peut contenir une section **[main]** et d'autres sections pour configurer les instances du plug-in. Cependant, toutes les sections sont facultatives.

Les lignes commençant par le signe dièse (**#**) sont des commentaires.

Ressources supplémentaires

- **tuned.conf(5)** page de manuel.

1.3. LE PROFIL TUNED PAR DÉFAUT

Lors de l'installation, le profil le mieux adapté à votre système est sélectionné automatiquement. Actuellement, le profil par défaut est sélectionné en fonction des règles personnalisables suivantes :

Environnement	Profil par défaut	Objectif
Nœuds de calcul	**throughput-performance**	La meilleure performance en termes de débit
Machines virtuelles	**virtual-guest**	La meilleure performance. Si vous n'êtes pas intéressé par les meilleures performances, vous pouvez choisir le profil **balanced** ou **powersave**.
Autres cas	**balanced**	Performances et consommation d'énergie équilibrées

Ressources supplémentaires

- **tuned.conf(5)** page de manuel.

1.4. PROFILS TUNED FUSIONNÉS

À titre expérimental, il est possible de sélectionner plusieurs profils à la fois. **TuneD** essaiera de les fusionner pendant le chargement.

En cas de conflit, les paramètres du dernier profil spécifié sont prioritaires.

Exemple 1.1. Faible consommation d'énergie dans un invité virtuel

L'exemple suivant optimise le système pour qu'il fonctionne dans une machine virtuelle afin d'obtenir les meilleures performances et le règle simultanément pour qu'il consomme peu d'énergie, la priorité étant la faible consommation d'énergie :

```
# tuned-adm profile virtual-guest powersave
```

AVERTISSEMENT

La fusion est effectuée automatiquement sans vérifier si la combinaison de paramètres qui en résulte a un sens. Par conséquent, la fonction peut régler certains paramètres de manière opposée, ce qui peut être contre-productif : par exemple, régler le disque pour un débit élevé en utilisant le profil **throughput-performance** et régler simultanément le spindown du disque à la valeur basse par le profil **spindown-disk**.

Ressources supplémentaires

*__tuned-adm__ man page. * __tuned.conf(5)__ man page.

1.5. EMPLACEMENT DES PROFILS TUNED

TuneD stocke les profils dans les répertoires suivants :

/usr/lib/tuned/

Les profils spécifiques à la distribution sont stockés dans le répertoire. Chaque profil a son propre répertoire. Le profil consiste en un fichier de configuration principal appelé **tuned.conf**, et éventuellement d'autres fichiers, par exemple des scripts d'aide.

/etc/tuned/

Si vous devez personnaliser un profil, copiez le répertoire du profil dans le répertoire utilisé pour les profils personnalisés. S'il existe deux profils portant le même nom, c'est le profil personnalisé situé dans **/etc/tuned/** qui est utilisé.

Ressources supplémentaires

- **tuned.conf(5)** page de manuel.

1.6. PROFILS TUNED DISTRIBUÉS AVEC RHEL

La liste suivante est une liste de profils qui sont installés avec **TuneD** sur Red Hat Enterprise Linux.

NOTE

Il peut y avoir des profils plus spécifiques à un produit ou des profils de tierce partie **TuneD** disponibles. Ces profils sont généralement fournis par des paquets RPM distincts.

balanced

Le profil d'économie d'énergie par défaut. Il se veut un compromis entre les performances et la consommation d'énergie. Il utilise l'auto-scaling et l'auto-tuning dans la mesure du possible. Le seul inconvénient est l'augmentation de la latence. Dans la version actuelle de **TuneD**, il active les plugins CPU, disque, audio et vidéo, et active le gouverneur de CPU **conservative**. L'option **radeon_powersave** utilise la valeur **dpm-balanced** si elle est prise en charge, sinon elle prend la valeur **auto**.
Il remplace l'attribut **energy_performance_preference** par le paramètre d'énergie **normal**. Il modifie également l'attribut **scaling_governor** policy en **conservative** ou **powersave** CPU governor.

powersave

Un profil pour des performances maximales en matière d'économie d'énergie. Il peut limiter les performances afin de minimiser la consommation d'énergie réelle. Dans la version actuelle de **TuneD**, il permet l'autosuspend USB, l'économie d'énergie WiFi et l'économie d'énergie Aggressive Link Power Management (ALPM) pour les adaptateurs hôtes SATA. Il planifie également l'économie d'énergie multicœur pour les systèmes à faible taux de réveil et active le gouverneur **ondemand**. Il active l'économie d'énergie audio AC97 ou, selon votre système, l'économie d'énergie HDA-Intel avec un délai de 10 secondes. Si votre système contient une carte graphique Radeon prise en charge avec KMS activé, le profil la configure en économie d'énergie automatique. Sur les ASUS Eee PC, un moteur Super Hybrid dynamique est activé.
Il remplace l'attribut **energy_performance_preference** par le paramètre d'énergie **powersave** ou **power**. Il modifie également l'attribut **scaling_governor** policy en **ondemand** ou **powersave** CPU governor.

NOTE

Dans certains cas, le profil **balanced** est plus efficace que le profil **powersave**.

Considérons qu'il y a une quantité définie de travail à effectuer, par exemple un fichier vidéo qui doit être transcodé. Votre machine peut consommer moins d'énergie si le transcodage est effectué à pleine puissance, car la tâche est terminée rapidement, la machine commence à tourner au ralenti et elle peut automatiquement passer à des modes d'économie d'énergie très efficaces. En revanche, si vous transcodez le fichier avec une machine bridée, la machine consomme moins d'énergie pendant le transcodage, mais le processus prend plus de temps et l'énergie totale consommée peut être plus élevée.

C'est pourquoi le profil **balanced** peut être une meilleure option.

throughput-performance

Profil de serveur optimisé pour un débit élevé. Il désactive les mécanismes d'économie d'énergie et active les paramètres **sysctl** qui améliorent le débit des disques et des entrées-sorties réseau. Le gouverneur de CPU est réglé sur **performance**.
Il transforme les attributs **energy_performance_preference** et **scaling_governor** en profil **performance**.

accelerator-performance

Le profil **accelerator-performance** contient les mêmes réglages que le profil **throughput-performance**. En outre, il verrouille le processeur sur des états C faibles afin que la latence soit inférieure à 100us. Cela permet d'améliorer les performances de certains accélérateurs, tels que les GPU.

latency-performance

Profil de serveur optimisé pour une faible latence. Il désactive les mécanismes d'économie d'énergie et active les paramètres **sysctl** qui améliorent la latence. Le gouverneur de CPU est réglé sur **performance** et le CPU est verrouillé sur les états de faible C (par PM QoS).
Il transforme les attributs **energy_performance_preference** et **scaling_governor** en profil **performance**.

network-latency

Un profil pour l'optimisation des réseaux à faible latence. Il est basé sur le profil **latency-performance**. Il désactive en outre les pages énormes transparentes et l'équilibrage NUMA, et règle plusieurs autres paramètres liés au réseau sur **sysctl**.

Il hérite du profil **latency-performance** qui transforme les attributs **energy_performance_preference** et **scaling_governor** en profil **performance**.

hpc-compute

Un profil optimisé pour le calcul à haute performance. Il est basé sur le profil **latency-performance**.

network-throughput

Un profil pour l'optimisation du débit des réseaux. Il est basé sur le profil **throughput-performance**. Il augmente en outre les tampons réseau du noyau.
Il hérite du profil **latency-performance** ou **throughput-performance** et transforme les attributs **energy_performance_preference** et **scaling_governor** en profil **performance**.

virtual-guest

Un profil conçu pour les machines virtuelles Red Hat Enterprise Linux 9 et les invités VMWare basé sur le profil **throughput-performance** qui, entre autres tâches, diminue l'échange de mémoire virtuelle et augmente les valeurs d'avance de lecture des disques. Il ne désactive pas les barrières de disque.
Il hérite du profil **throughput-performance** et remplace les attributs **energy_performance_preference** et **scaling_governor** par le profil **performance**.

virtual-host

Profil conçu pour les hôtes virtuels sur la base du profil **throughput-performance** qui, entre autres tâches, diminue la permutation de la mémoire virtuelle, augmente les valeurs d'avance de lecture des disques et permet une valeur plus agressive de l'écriture des pages sales.
Il hérite du profil **throughput-performance** et remplace les attributs **energy_performance_preference** et **scaling_governor** par le profil **performance**.

oracle

Un profil optimisé pour les chargements de bases de données Oracle basé sur le profil **throughput-performance**. Il désactive en outre les pages énormes transparentes et modifie d'autres paramètres du noyau liés aux performances. Ce profil est fourni par le paquetage **tuned-profiles-oracle**.

desktop

Profil optimisé pour les ordinateurs de bureau, basé sur le profil **balanced**. Il permet en outre des autogroupes de planificateurs pour une meilleure réponse des applications interactives.

optimize-serial-console

Un profil qui réduit l'activité des E/S vers la console série en réduisant la valeur de printk. Cela devrait rendre la console série plus réactive. Ce profil est destiné à être utilisé en superposition à d'autres profils. Par exemple :

```
# tuned-adm profile throughput-performance optimize-serial-console
```

mssql

Un profil fourni pour Microsoft SQL Server. Il est basé sur le profil **throughput-performance**.

intel-sst

Profil optimisé pour les systèmes avec des configurations Intel Speed Select Technology définies par l'utilisateur. Ce profil est destiné à être utilisé en superposition à d'autres profils. Par exemple :

```
# tuned-adm profile cpu-partitioning intel-sst
```

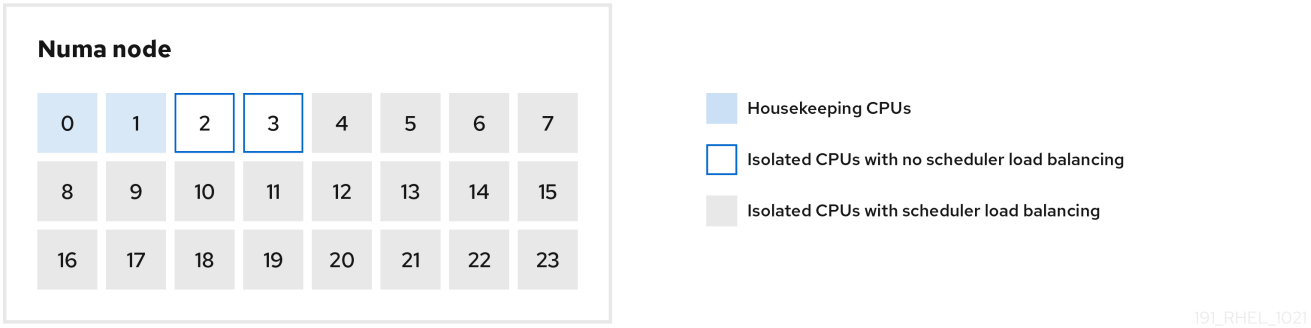
1.7. PROFIL DE PARTITIONNEMENT DU PROCESSEUR TUNED

Pour régler Red Hat Enterprise Linux 9 pour les charges de travail sensibles à la latence, Red Hat recommande d'utiliser le profil **cpu-partitioning** TuneD.

Avant Red Hat Enterprise Linux 9, la documentation Red Hat sur les faibles temps de latence décrivait les nombreuses étapes de bas niveau nécessaires à l'obtention d'un réglage des faibles temps de latence. Dans Red Hat Enterprise Linux 9, vous pouvez effectuer un réglage de faible latence plus efficacement en utilisant le profil **cpu-partitioning** TuneD. Ce profil est facilement personnalisable en fonction des exigences des applications individuelles à faible latence.

La figure suivante est un exemple d'utilisation du profil **cpu-partitioning**. Cet exemple utilise la disposition de l'unité centrale et des nœuds.

Figure 1.1. Figure partitionnement du processeur



Vous pouvez configurer le profil de partitionnement du processeur dans le fichier **/etc/tuned/cpu-partitioning-variables.conf** à l'aide des options de configuration suivantes :

CPU isolés avec répartition de la charge

Dans la figure de partitionnement des processeurs, les blocs numérotés de 4 à 23 sont les processeurs isolés par défaut. L'équilibrage de la charge des processus de l'ordonnanceur du noyau est activé sur ces CPU. Il est conçu pour les processus à faible latence avec plusieurs threads qui ont besoin de l'équilibrage de la charge du planificateur du noyau.
Vous pouvez configurer le profil de partitionnement des processeurs dans le fichier **/etc/tuned/cpu-partitioning-variables.conf** à l'aide de l'option **isolated_cores=cpu-list**, qui répertorie les processeurs à isoler qui utiliseront l'équilibrage de charge de l'ordonnanceur du noyau.

La liste des unités centrales isolées est séparée par des virgules ou vous pouvez spécifier une plage à l'aide d'un tiret, comme **3-5**. Cette option est obligatoire. Toute unité centrale absente de cette liste est automatiquement considérée comme une unité centrale de maintenance.

CPU isolés sans répartition de la charge

Dans la figure de partitionnement des CPU, les blocs numérotés 2 et 3 sont les CPU isolés qui ne fournissent pas d'équilibrage supplémentaire de la charge des processus de l'ordonnanceur du noyau.
Vous pouvez configurer le profil de partitionnement des processeurs dans le fichier **/etc/tuned/cpu-partitioning-variables.conf** à l'aide de l'option **no_balance_cores=cpu-list**, qui répertorie les processeurs à isoler qui n'utiliseront pas l'équilibrage de charge de l'ordonnanceur du noyau.

La spécification de l'option **no_balance_cores** est facultative, mais tous les processeurs de cette liste doivent être un sous-ensemble des processeurs figurant dans la liste **isolated_cores**.

Les threads d'application qui utilisent ces CPU doivent être épinglés individuellement à chaque CPU.

Unité centrale d'entretien

Toute unité centrale qui n'est pas isolée dans le fichier **cpu-partitioning-variables.conf** est automatiquement considérée comme une unité centrale de maintenance. Sur ces unités centrales, tous les services, démons, processus utilisateur, threads mobiles du noyau, gestionnaires d'interruption et temporisateurs du noyau sont autorisés à s'exécuter.

Ressources supplémentaires

- **tuned-profiles-cpu-partitioning(7)** page de manuel

1.8. UTILISATION DU PROFIL DE PARTITIONNEMENT DU PROCESSEUR TUNED POUR UN RÉGLAGE À FAIBLE LATENCE

Cette procédure décrit comment régler un système pour une faible latence en utilisant le profil **cpu-partitioning** de TuneD. Elle utilise l'exemple d'une application à faible latence qui peut utiliser **cpu-partitioning** et la disposition du processeur comme indiqué dans la figure de partitionnement du processeur.

Dans ce cas, l'application utilise :

- Un thread de lecture dédié, qui lit les données du réseau, sera placé sur l'unité centrale 2.
- Un grand nombre de threads qui traitent ces données réseau seront épinglés sur les CPU 4-23.
- Un thread d'écriture dédié qui écrit les données traitées sur le réseau sera placé sur l'unité centrale 3.

Conditions préalables

- Vous avez installé le profil TuneD **cpu-partitioning** en utilisant la commande **dnf install tuned-profiles-cpu-partitioning** en tant que root.

Procédure

1. Modifiez le fichier **/etc/tuned/cpu-partitioning-variables.conf** et ajoutez les informations suivantes :

```
# Isolated CPUs with the kernel's scheduler load balancing:
isolated_cores=2-23
# Isolated CPUs without the kernel's scheduler load balancing:
no_balance_cores=2,3
```

2. Définir le profil **cpu-partitioning** TuneD :

```
# tuned-adm profile cpu-partitioning
```

3. Reboot
   Après le redémarrage, le système est réglé pour une faible latence, conformément à l'isolation dans la figure de partitionnement des processeurs. L'application peut utiliser taskset pour affecter les threads de lecture et d'écriture aux CPU 2 et 3, et les threads d'application restants aux CPU 4 à 23.

Ressources supplémentaires

- **tuned-profiles-cpu-partitioning(7)** page de manuel

1.9. PERSONNALISATION DU PROFIL TUNED DE PARTITIONNEMENT DU PROCESSEUR

Vous pouvez étendre le profil TuneD pour apporter des modifications supplémentaires à l'accord.

Par exemple, le profil **cpu-partitioning** configure les unités centrales pour qu'elles utilisent **cstate=1**. Afin d'utiliser le profil **cpu-partitioning** mais de changer en plus l'état c du CPU de cstate1 à cstate0, la procédure suivante décrit un nouveau profil TuneD nommé *my_profile*, qui hérite du profil **cpu-partitioning** et définit ensuite l'état C-0.

Procédure

1. Créez le répertoire **/etc/tuned/my_profile**:

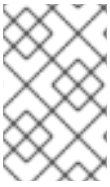
```
# mkdir /etc/tuned/my_profile
```

2. Créez un fichier **tuned.conf** dans ce répertoire et ajoutez-y le contenu suivant :

```
# vi /etc/tuned/my_profile/tuned.conf
[main]
summary=Customized tuning on top of cpu-partitioning
include=cpu-partitioning
[cpu]
force_latency=cstate.id:0|1
```

3. Utiliser le nouveau profil :

```
# tuned-adm profile my_profile
```



NOTE

Dans l'exemple partagé, un redémarrage n'est pas nécessaire. Toutefois, si les modifications apportées au profil *my_profile* nécessitent un redémarrage pour être prises en compte, redémarrez votre machine.

Ressources supplémentaires

- **tuned-profiles-cpu-partitioning(7)** page de manuel

1.10. PROFILS TUNED EN TEMPS RÉEL DISTRIBUÉS AVEC RHEL

Les profils temps réel sont destinés aux systèmes utilisant le noyau temps réel. Sans une compilation spéciale du noyau, ils ne configurent pas le système pour qu'il soit en temps réel. Sur RHEL, les profils sont disponibles dans des dépôts supplémentaires.

Les profils en temps réel suivants sont disponibles :

realtime

Utilisation sur des systèmes temps réel nus.
Fourni par le paquetage **tuned-profiles-realtime**, qui est disponible dans les dépôts RT ou NFV.

realtime-virtual-host

Utilisation dans un hôte de virtualisation configuré pour le temps réel.
Fourni par le paquetage **tuned-profiles-nfv-host**, qui est disponible dans le référentiel NFV.

realtime-virtual-guest

Utilisation dans un invité de virtualisation configuré pour le temps réel.
Fourni par le paquetage **tuned-profiles-nfv-guest**, qui est disponible dans le référentiel NFV.

1.11. ACCORD STATIQUE ET DYNAMIQUE DANS TUNED

Il est important de comprendre la différence entre les deux catégories de réglage de système que **TuneD** applique, *static* et *dynamic*, pour déterminer laquelle utiliser pour une situation ou un objectif donné.

Accord statique

Il s'agit principalement de l'application de paramètres prédéfinis **sysctl** et **sysfs** et de l'activation ponctuelle de plusieurs outils de configuration tels que **ethtool**.

Accord dynamique

Surveille la façon dont les différents composants du système sont utilisés tout au long de la durée de fonctionnement de votre système. **TuneD** ajuste les paramètres du système de façon dynamique sur la base de ces informations de surveillance.
Par exemple, le disque dur est fortement sollicité lors du démarrage et de la connexion, mais il est à peine utilisé par la suite, lorsque l'utilisateur travaille principalement avec des applications telles que des navigateurs web ou des clients de messagerie. De même, l'unité centrale et les périphériques réseau sont utilisés différemment selon les moments. **TuneD** surveille l'activité de ces composants et réagit aux changements dans leur utilisation.

Par défaut, l'optimisation dynamique est désactivée. Pour l'activer, modifiez le fichier **/etc/tuned/tuned-main.conf** et remplacez l'option **dynamic_tuning** par **1**. **TuneD** analyse alors périodiquement les statistiques du système et les utilise pour mettre à jour les paramètres de réglage de votre système. Pour configurer l'intervalle de temps en secondes entre ces mises à jour, utilisez l'option **update_interval**.

Les algorithmes de réglage dynamique actuellement mis en œuvre tentent d'équilibrer les performances et l'économie d'énergie, et sont donc désactivés dans les profils de performance. L'accord dynamique pour les plug-ins individuels peut être activé ou désactivé dans les profils **TuneD**.

Exemple 1.2. Accord statique et dynamique sur un poste de travail

Sur un poste de travail de bureau classique, l'interface réseau Ethernet est inactive la plupart du temps. Seuls quelques courriers électroniques entrent et sortent ou quelques pages web peuvent être chargées.

Pour ce type de charge, l'interface réseau n'a pas besoin de tourner à plein régime en permanence, comme c'est le cas par défaut. **TuneD** dispose d'un plug-in de surveillance et de réglage pour les périphériques réseau qui peut détecter cette faible activité et réduire automatiquement la vitesse de l'interface, ce qui se traduit généralement par une réduction de la consommation d'énergie.

Si l'activité sur l'interface augmente pendant une période prolongée, par exemple parce qu'une image de DVD est téléchargée ou qu'un e-mail contenant une pièce jointe volumineuse est ouvert, **TuneD** le détecte et règle la vitesse de l'interface au maximum afin d'offrir les meilleures

performances lorsque le niveau d'activité est élevé.

Ce principe est également utilisé pour d'autres plug-ins pour l'unité centrale et les disques.

1.12. MODE TUNED SANS DÉMON

Vous pouvez exécuter **TuneD** en mode **no-daemon**, qui ne nécessite pas de mémoire résidente. Dans ce mode, **TuneD** applique les paramètres et se termine.

Par défaut, le mode **no-daemon** est désactivé car de nombreuses fonctionnalités de **TuneD** sont absentes dans ce mode, notamment :

- Support D-Bus
- Prise en charge de la connexion à chaud
- Prise en charge du retour en arrière pour les paramètres

Pour activer le mode **no-daemon**, incluez la ligne suivante dans le fichier **/etc/tuned/tuned-main.conf**:

```
daemon = 0
```

1.13. INSTALLATION ET ACTIVATION DE TUNED

Cette procédure permet d'installer et d'activer l'application **TuneD**, d'installer les profils **TuneD** et de prédéfinir un profil **TuneD** par défaut pour votre système.

Procédure

1. Installez le paquetage **TuneD**:

   ```
   # dnf install tuned
   ```

2. Activez et démarrez le service **TuneD**:

   ```
   # systemctl enable --now tuned
   ```

3. En option, installez les profils **TuneD** pour les systèmes en temps réel :
   Pour les profils **TuneD** pour les systèmes en temps réel, activez le référentiel **rhel-9**.

   ```
   # subscription-manager repos --enable=rhel-9-for-x86_64-nfv-beta-rpms
   ```

   Installez-le.

   ```
   # dnf install tuned-profiles-realtime tuned-profiles-nfv
   ```

4. Vérifiez qu'un profil **TuneD** est actif et appliqué :

   ```
   $ tuned-adm active

   Current active profile: throughput-performance
   ```

NOTE

Le profil actif présélectionné automatiquement par TuneD diffère selon le type de machine et les paramètres du système.

```
$ tuned-adm verify

Verification succeeded, current system settings match the preset profile.
See tuned log file ('/var/log/tuned/tuned.log') for details.
```

1.14. LISTE DES PROFILS TUNED DISPONIBLES

Cette procédure dresse la liste de tous les profils **TuneD** actuellement disponibles sur votre système.

Procédure

- Pour dresser la liste de tous les profils **TuneD** disponibles sur votre système, utilisez la touche

```
$ tuned-adm list

Available profiles:
- accelerator-performance - Throughput performance based tuning with disabled higher latency STOP states
- balanced               - General non-specialized TuneD profile
- desktop                - Optimize for the desktop use-case
- latency-performance    - Optimize for deterministic performance at the cost of increased power consumption
- network-latency        - Optimize for deterministic performance at the cost of increased power consumption, focused on low latency network performance
- network-throughput     - Optimize for streaming network throughput, generally only necessary on older CPUs or 40G+ networks
- powersave              - Optimize for low power consumption
- throughput-performance - Broadly applicable tuning that provides excellent performance across a variety of common server workloads
- virtual-guest          - Optimize for running inside a virtual guest
- virtual-host           - Optimize for running KVM guests
Current active profile: balanced
```

- Pour n'afficher que le profil actuellement actif, utilisez :

```
$ tuned-adm active

Current active profile: throughput-performance
```

Ressources supplémentaires

- **tuned-adm(8)** page de manuel.

1.15. DÉFINITION D'UN PROFIL TUNED

Cette procédure permet d'activer un profil **TuneD** sélectionné sur votre système.

Conditions préalables

- Le service **TuneD** est en cours d'exécution. Voir Installation et activation de TuneD pour plus de détails.

Procédure

1. En option, vous pouvez laisser **TuneD** vous recommander le profil le plus adapté à votre système :

```
# tuned-adm recommend

throughput-performance
```

2. Activer un profil :

```
# tuned-adm profile selected-profile
```

Vous pouvez également activer une combinaison de plusieurs profils :

```
# tuned-adm profile selected-profile1 selected-profile2
```

Exemple 1.3. Une machine virtuelle optimisée pour une faible consommation d'énergie

L'exemple suivant permet d'optimiser le système pour qu'il fonctionne dans une machine virtuelle offrant les meilleures performances et de le régler simultanément pour qu'il consomme peu d'énergie, la priorité étant de consommer peu d'énergie :

```
# tuned-adm profile virtual-guest powersave
```

3. Affichez le profil **TuneD** actuellement actif sur votre système :

```
# tuned-adm active

Current active profile: selected-profile
```

4. Redémarrer le système :

```
# reboot
```

Verification steps

- Vérifiez que le profil **TuneD** est actif et appliqué :

```
$ tuned-adm verify

Verification succeeded, current system settings match the preset profile.
See tuned log file ('/var/log/tuned/tuned.log') for details.
```

Ressources supplémentaires

- **tuned-adm(8)** page de manuel

1.16. DÉSACTIVATION DE TUNED

Cette procédure désactive **TuneD** et réinitialise tous les paramètres système concernés à leur état d'origine avant que **TuneD** ne les modifie.

Procédure

- Pour désactiver temporairement tous les réglages :

```
# tuned-adm off
```

  Les réglages sont de nouveau appliqués après le redémarrage du service **TuneD**.

- Il est également possible d'arrêter et de désactiver définitivement le service **TuneD**:

```
# systemctl disable --now tuned
```

Ressources supplémentaires

- **tuned-adm(8)** page de manuel

CHAPITRE 2. PERSONNALISATION DES PROFILS TUNED

Vous pouvez créer ou modifier les profils **TuneD** afin d'optimiser les performances du système en fonction de l'utilisation que vous souhaitez en faire.

Conditions préalables

- Installez et activez **TuneD** comme décrit dans Installation et activation de TuneD pour plus de détails.

2.1. PROFILS TUNED

L'analyse détaillée d'un système peut prendre beaucoup de temps. **TuneD** fournit un certain nombre de profils prédéfinis pour des cas d'utilisation typiques. Vous pouvez également créer, modifier et supprimer des profils.

Les profils fournis par **TuneD** sont répartis dans les catégories suivantes :

- Profils d'économie d'énergie
- Profils d'amélioration des performances

Les profils d'amélioration des performances comprennent des profils qui se concentrent sur les aspects suivants :

- Faible latence pour le stockage et le réseau
- Débit élevé pour le stockage et le réseau
- Performances de la machine virtuelle
- Performances des hôtes de virtualisation

Syntaxe de la configuration du profil

Le fichier **tuned.conf** peut contenir une section **[main]** et d'autres sections pour configurer les instances du plug-in. Cependant, toutes les sections sont facultatives.

Les lignes commençant par le signe dièse (**#**) sont des commentaires.

Ressources supplémentaires

- **tuned.conf(5)** page de manuel.

2.2. LE PROFIL TUNED PAR DÉFAUT

Lors de l'installation, le profil le mieux adapté à votre système est sélectionné automatiquement. Actuellement, le profil par défaut est sélectionné en fonction des règles personnalisables suivantes :

Environnement	Profil par défaut	Objectif
Nœuds de calcul	**throughput-performance**	La meilleure performance en termes de débit

Environnement	Profil par défaut	Objectif
Machines virtuelles	**virtual-guest**	La meilleure performance. Si vous n'êtes pas intéressé par les meilleures performances, vous pouvez choisir le profil **balanced** ou **powersave**.
Autres cas	**balanced**	Performances et consommation d'énergie équilibrées

Ressources supplémentaires

- **tuned.conf(5)** page de manuel.

2.3. PROFILS TUNED FUSIONNÉS

À titre expérimental, il est possible de sélectionner plusieurs profils à la fois. **TuneD** essaiera de les fusionner pendant le chargement.

En cas de conflit, les paramètres du dernier profil spécifié sont prioritaires.

> **Exemple 2.1. Faible consommation d'énergie dans un invité virtuel**
>
> L'exemple suivant optimise le système pour qu'il fonctionne dans une machine virtuelle afin d'obtenir les meilleures performances et le règle simultanément pour qu'il consomme peu d'énergie, la priorité étant la faible consommation d'énergie :
>
> ```
> # tuned-adm profile virtual-guest powersave
> ```

AVERTISSEMENT

La fusion est effectuée automatiquement sans vérifier si la combinaison de paramètres qui en résulte a un sens. Par conséquent, la fonction peut régler certains paramètres de manière opposée, ce qui peut être contre-productif : par exemple, régler le disque pour un débit élevé en utilisant le profil **throughput-performance** et régler simultanément le spindown du disque à la valeur basse par le profil **spindown-disk**.

Ressources supplémentaires

***tuned-adm** man page. * **tuned.conf(5)** man page.

2.4. EMPLACEMENT DES PROFILS TUNED

TuneD stocke les profils dans les répertoires suivants :

/usr/lib/tuned/

Les profils spécifiques à la distribution sont stockés dans le répertoire. Chaque profil a son propre répertoire. Le profil consiste en un fichier de configuration principal appelé **tuned.conf**, et éventuellement d'autres fichiers, par exemple des scripts d'aide.

/etc/tuned/

Si vous devez personnaliser un profil, copiez le répertoire du profil dans le répertoire utilisé pour les profils personnalisés. S'il existe deux profils portant le même nom, c'est le profil personnalisé situé dans **/etc/tuned/** qui est utilisé.

Ressources supplémentaires

- **tuned.conf(5)** page de manuel.

2.5. HÉRITAGE ENTRE LES PROFILS TUNED

TuneD les profils peuvent être basés sur d'autres profils et ne modifier que certains aspects de leur profil parent.

La section **[main]** des profils **TuneD** reconnaît l'option **include**:

```
[main]
include=parent
```

Tous les paramètres du profil *parent* sont chargés dans le profil *child*. Dans les sections suivantes, le profil *child* peut remplacer certains paramètres hérités du profil *parent* ou ajouter de nouveaux paramètres qui ne sont pas présents dans le *parent* ou ajouter de nouveaux paramètres qui ne sont pas présents dans le profil.

Vous pouvez créer votre propre profil *child* dans le répertoire **/etc/tuned/** sur la base d'un profil préinstallé dans **/usr/lib/tuned/** avec seulement quelques paramètres ajustés.

Si le profil *parent* est mis à jour, par exemple après une mise à niveau de **TuneD**, les modifications sont répercutées dans le profil *child*.

Exemple 2.2. Un profil d'économie d'énergie basé sur un équilibre

Voici un exemple de profil personnalisé qui étend le profil **balanced** et définit la gestion agressive de l'alimentation des liens (ALPM) pour tous les appareils sur l'économie d'énergie maximale.

```
[main]
include=balanced

[scsi_host]
alpm=min_power
```

Ressources supplémentaires

- **tuned.conf(5)** page de manuel

2.6. ACCORD STATIQUE ET DYNAMIQUE DANS TUNED

Il est important de comprendre la différence entre les deux catégories de réglage de système que **TuneD** applique, *static* et *dynamic*, pour déterminer laquelle utiliser pour une situation ou un objectif donné.

Accord statique

Il s'agit principalement de l'application de paramètres prédéfinis **sysctl** et **sysfs** et de l'activation ponctuelle de plusieurs outils de configuration tels que **ethtool**.

Accord dynamique

Surveille la façon dont les différents composants du système sont utilisés tout au long de la durée de fonctionnement de votre système. **TuneD** ajuste les paramètres du système de façon dynamique sur la base de ces informations de surveillance.
Par exemple, le disque dur est fortement sollicité lors du démarrage et de la connexion, mais il est à peine utilisé par la suite, lorsque l'utilisateur travaille principalement avec des applications telles que des navigateurs web ou des clients de messagerie. De même, l'unité centrale et les périphériques réseau sont utilisés différemment selon les moments. **TuneD** surveille l'activité de ces composants et réagit aux changements dans leur utilisation.

Par défaut, l'optimisation dynamique est désactivée. Pour l'activer, modifiez le fichier **/etc/tuned/tuned-main.conf** et remplacez l'option **dynamic_tuning** par **1**. **TuneD** analyse alors périodiquement les statistiques du système et les utilise pour mettre à jour les paramètres de réglage de votre système. Pour configurer l'intervalle de temps en secondes entre ces mises à jour, utilisez l'option **update_interval**.

Les algorithmes de réglage dynamique actuellement mis en œuvre tentent d'équilibrer les performances et l'économie d'énergie, et sont donc désactivés dans les profils de performance. L'accord dynamique pour les plug-ins individuels peut être activé ou désactivé dans les profils **TuneD**.

Exemple 2.3. Accord statique et dynamique sur un poste de travail

Sur un poste de travail de bureau classique, l'interface réseau Ethernet est inactive la plupart du temps. Seuls quelques courriers électroniques entrent et sortent ou quelques pages web peuvent être chargées.

Pour ce type de charge, l'interface réseau n'a pas besoin de tourner à plein régime en permanence, comme c'est le cas par défaut. **TuneD** dispose d'un plug-in de surveillance et de réglage pour les périphériques réseau qui peut détecter cette faible activité et réduire automatiquement la vitesse de l'interface, ce qui se traduit généralement par une réduction de la consommation d'énergie.

Si l'activité sur l'interface augmente pendant une période prolongée, par exemple parce qu'une image de DVD est téléchargée ou qu'un e-mail contenant une pièce jointe volumineuse est ouvert, **TuneD** le détecte et règle la vitesse de l'interface au maximum afin d'offrir les meilleures performances lorsque le niveau d'activité est élevé.

Ce principe est également utilisé pour d'autres plug-ins pour l'unité centrale et les disques.

2.7. PLUG-INS TUNED

Les plug-ins sont des modules dans les profils **TuneD** que **TuneD** utilise pour surveiller ou optimiser différents dispositifs sur le système.

TuneD utilise deux types de plug-ins :

Surveillance des plug-ins

Les modules d'extension de surveillance sont utilisés pour obtenir des informations sur un système en cours d'exécution. La sortie des plug-ins de surveillance peut être utilisée par les plug-ins de réglage pour le réglage dynamique.
Les plug-ins de surveillance sont automatiquement instanciés chaque fois que leurs métriques sont nécessaires à l'un des plug-ins d'optimisation activés. Si deux plug-ins de réglage ont besoin des mêmes données, une seule instance du plug-in de surveillance est créée et les données sont partagées.

Plug-ins d'accordage

Chaque plug-in de réglage règle un sous-système individuel et prend plusieurs paramètres issus des profils TuneD. Chaque sous-système peut avoir plusieurs périphériques, tels que plusieurs CPU ou cartes réseau, qui sont gérés par des instances individuelles des plug-ins de réglage. Des paramètres spécifiques pour des périphériques individuels sont également pris en charge.

Syntaxe pour les plug-ins dans les profils TuneD

Les sections décrivant les instances de plug-in sont formatées de la manière suivante :

```
[NAME]
type=TYPE
devices=DEVICES
```

NAME

est le nom de l'instance du plug-in tel qu'il est utilisé dans les journaux. Il peut s'agir d'une chaîne arbitraire.

TYPE

est le type de plug-in d'accord.

DEVICES

est la liste des dispositifs gérés par cette instance de plug-in.
La ligne **devices** peut contenir une liste, un caractère de remplacement (*****) et une négation (**!**). S'il n'y a pas de ligne **devices**, tous les dispositifs présents ou connectés ultérieurement sur le système de l'instance de plug-in sont pris en charge par l'instance de plug-in *TYPE* sont pris en charge par l'instance de plug-in. Cela revient à utiliser l'option **devices=***.

Exemple 2.4. Correspondance entre les dispositifs de blocage et un plug-in

L'exemple suivant correspond à tous les périphériques de bloc commençant par **sd**, tels que **sda** ou **sdb**, et ne désactive pas les barrières sur ces périphériques :

```
[data_disk]
type=disk
devices=sd*
disable_barriers=false
```

L'exemple suivant fait correspondre tous les blocs à l'exception de **sda1** et **sda2**:

```
[data_disk]
type=disk
devices=!sda1, !sda2
disable_barriers=false
```

Si aucune instance d'un plug-in n'est spécifiée, le plug-in n'est pas activé.

Si le plug-in prend en charge d'autres options, celles-ci peuvent également être spécifiées dans la section du plug-in. Si l'option n'est pas spécifiée et qu'elle n'a pas été spécifiée précédemment dans le plug-in inclus, la valeur par défaut est utilisée.

Syntaxe courte du plug-in

Si vous n'avez pas besoin de noms personnalisés pour l'instance du plug-in et qu'il n'y a qu'une seule définition de l'instance dans votre fichier de configuration, **TuneD** prend en charge la syntaxe courte suivante :

```
[TYPE]
devices=DEVICES
```

Dans ce cas, il est possible d'omettre la ligne **type**. L'instance est alors désignée par un nom, identique à celui du type. L'exemple précédent pourrait alors être réécrit en :

Exemple 2.5. Correspondance entre les dispositifs de blocage à l'aide de la syntaxe courte

```
[disk]
devices=sdb*
disable_barriers=false
```

Définitions contradictoires de plug-ins dans un profil

Si la même section est spécifiée plusieurs fois à l'aide de l'option **include**, les paramètres sont fusionnés. S'ils ne peuvent pas être fusionnés en raison d'un conflit, la dernière définition conflictuelle remplace les paramètres précédents. Si vous ne savez pas ce qui a été défini précédemment, vous pouvez utiliser l'option booléenne **replace** et lui attribuer la valeur **true**. Toutes les définitions précédentes portant le même nom sont alors écrasées et la fusion n'a pas lieu.

Vous pouvez également désactiver le plug-in en spécifiant l'option **enabled=false**. L'effet est le même que si l'instance n'avait jamais été définie. La désactivation du plug-in est utile si vous redéfinissez la définition précédente à partir de l'option **include** et que vous ne souhaitez pas que le plug-in soit actif dans votre profil personnalisé.

NOTE

TuneD inclut la possibilité d'exécuter n'importe quelle commande shell dans le cadre de l'activation ou de la désactivation d'un profil d'accord. Cela vous permet d'étendre les profils **TuneD** avec des fonctionnalités qui n'ont pas encore été intégrées dans TuneD.
Vous pouvez spécifier des commandes shell arbitraires à l'aide du plug-in **script**.

Ressources supplémentaires

- **tuned.conf(5)** page de manuel

2.8. PLUG-INS TUNED DISPONIBLES

Surveillance des plug-ins

Actuellement, les plug-ins de surveillance suivants sont mis en œuvre :

disk

Obtient la charge du disque (nombre d'opérations d'E/S) par périphérique et par intervalle de mesure.

net

Obtient la charge du réseau (nombre de paquets transférés) par carte réseau et par intervalle de mesure.

load

Obtient la charge de l'unité centrale par unité centrale et l'intervalle de mesure.

Plug-ins d'accordage

Actuellement, les plug-ins d'accord suivants sont mis en œuvre. Seuls certains de ces plug-ins mettent en œuvre l'accord dynamique. Les options prises en charge par les plug-ins sont également répertoriées :

cpu

Définit le gouverneur de CPU à la valeur spécifiée par l'option **governor** et modifie dynamiquement la latence d'accès direct à la mémoire (DMA) du CPU dans le cadre de la qualité de service de la gestion de l'énergie (PM QoS) en fonction de la charge du CPU.
Si la charge du CPU est inférieure à la valeur spécifiée par l'option **load_threshold**, la latence est fixée à la valeur spécifiée par l'option **latency_high**, sinon elle est fixée à la valeur spécifiée par **latency_low**.

Vous pouvez également forcer la latence à une valeur spécifique et l'empêcher de changer dynamiquement. Pour ce faire, réglez l'option **force_latency** sur la valeur de latence requise.

eeepc_she

Définit dynamiquement la vitesse du bus frontal (FSB) en fonction de la charge du processeur.
Cette fonction, que l'on retrouve sur certains netbooks, est également connue sous le nom de Super Hybrid Engine (SHE) d'ASUS.

Si la charge du CPU est inférieure ou égale à la valeur spécifiée par l'option **load_threshold_powersave**, le plug-in fixe la vitesse du FSB à la valeur spécifiée par l'option **she_powersave**. Si la charge du CPU est supérieure ou égale à la valeur spécifiée par l'option **load_threshold_normal**, il fixe la vitesse du FSB à la valeur spécifiée par l'option **she_normal**.

L'accord statique n'est pas pris en charge et le plug-in est désactivé de manière transparente si **TuneD** ne détecte pas la prise en charge matérielle de cette fonctionnalité.

net

Configure la fonctionnalité Wake-on-LAN aux valeurs spécifiées par l'option **wake_on_lan**. Il utilise la même syntaxe que l'utilitaire **ethtool**. Il modifie aussi dynamiquement la vitesse de l'interface en fonction de son utilisation.

sysctl

Définit divers paramètres **sysctl** spécifiés par les options du plug-in.
La syntaxe est la suivante ***name=value*** où *name* est identique au nom fourni par l'utilitaire **sysctl**.

Utilisez le plug-in **sysctl** si vous devez modifier des paramètres du système qui ne sont pas couverts par d'autres plug-ins disponibles dans **TuneD**. Si les paramètres sont couverts par certains plug-ins spécifiques, préférez ces plug-ins.

usb

Définit le délai de suspension automatique des périphériques USB à la valeur spécifiée par le paramètre **autosuspend**.
La valeur **0** signifie que la suspension automatique est désactivée.

vm

Active ou désactive les grandes pages transparentes en fonction de la valeur de l'option **transparent_hugepages**.
Les valeurs valides de l'option **transparent_hugepages** sont les suivantes :

- \toujours"
- \Jamais
- "madvise"

audio

Définit le délai de suspension automatique pour les codecs audio à la valeur spécifiée par l'option **timeout**.
Actuellement, les codecs **snd_hda_intel** et **snd_ac97_codec** sont pris en charge. La valeur **0** signifie que la suspension automatique est désactivée. Vous pouvez également imposer la réinitialisation du contrôleur en définissant l'option booléenne **reset_controller** sur **true**.

disk

Définit l'ascenseur de disque à la valeur spécifiée par l'option **elevator**.
Il fixe également :

- APM à la valeur spécifiée par l'option **apm**
- Quantum de l'ordonnanceur à la valeur spécifiée par l'option **scheduler_quantum**
- Délai d'attente du disque à la valeur spécifiée par l'option **spindown**
- La valeur de l'avance de lecture du disque est fixée à la valeur spécifiée par le paramètre **readahead**
- L'avance de lecture actuelle du disque à une valeur multipliée par la constante spécifiée par l'option **readahead_multiply**

En outre, ce plug-in modifie dynamiquement les paramètres de gestion avancée de l'énergie et de temporisation du variateur en fonction de l'utilisation actuelle du variateur. Le réglage dynamique peut être contrôlé par l'option booléenne **dynamic** et est activé par défaut.

scsi_host

Ajuste les options pour les hôtes SCSI.
Il définit la gestion agressive de l'alimentation de la liaison (ALPM) à la valeur spécifiée par l'option **alpm**.

mounts

Active ou désactive les barrières pour les montages en fonction de la valeur booléenne de l'option **disable_barriers**.

script

Exécute un script ou un binaire externe lorsque le profil est chargé ou déchargé. Vous pouvez choisir un exécutable arbitraire.

IMPORTANT

Le plug-in **script** est fourni principalement à des fins de compatibilité avec les versions antérieures. Préférez d'autres plug-ins **TuneD** s'ils couvrent les fonctionnalités requises.

TuneD appelle l'exécutable avec l'un des arguments suivants :

- **start** lors du chargement du profil
- **stop** lors du déchargement du profil

Vous devez mettre en œuvre correctement l'action **stop** dans votre exécutable et annuler tous les paramètres que vous avez modifiés au cours de l'action **start**. Sinon, l'étape de retour en arrière après avoir modifié votre profil **TuneD** ne fonctionnera pas.

Les scripts Bash peuvent importer la bibliothèque Bash **/usr/lib/tuned/functions** et utiliser les fonctions qui y sont définies. N'utilisez ces fonctions que pour des fonctionnalités qui ne sont pas fournies de manière native par **TuneD**. Si le nom d'une fonction commence par un trait de soulignement, comme **_wifi_set_power_level**, considérez la fonction comme privée et ne l'utilisez pas dans vos scripts, car elle pourrait changer à l'avenir.

Spécifiez le chemin d'accès à l'exécutable à l'aide du paramètre **script** dans la configuration du plug-in.

Exemple 2.6. Exécuter un script Bash à partir d'un profil

Pour exécuter un script Bash nommé **script.sh** qui se trouve dans le répertoire du profil, utilisez :

```
[script]
script=${i:PROFILE_DIR}/script.sh
```

sysfs

Définit divers paramètres **sysfs** spécifiés par les options du plug-in.
La syntaxe est la suivante ***name=value***, où *name* est le chemin d'accès **sysfs** à utiliser.

Utilisez ce plugin si vous devez modifier certains paramètres qui ne sont pas couverts par d'autres plugins. Préférez des plugins spécifiques s'ils couvrent les paramètres requis.

video

Définit différents niveaux d'économie d'énergie sur les cartes vidéo. Actuellement, seules les cartes Radeon sont prises en charge.
Le niveau d'économie d'énergie peut être spécifié à l'aide de l'option **radeon_powersave**. Les valeurs prises en charge sont les suivantes :

- **default**
- **auto**
- **low**
- **mid**

- **high**
- **dynpm**
- **dpm-battery**
- **dpm-balanced**
- **dpm-perfomance**

Pour plus de détails, voir www.x.org. Notez que ce plug-in est expérimental et que l'option pourrait être modifiée dans les prochaines versions.

bootloader

Ajoute des options à la ligne de commande du noyau. Ce plug-in ne prend en charge que le chargeur de démarrage GRUB 2.
Un emplacement personnalisé non standard du fichier de configuration GRUB 2 peut être spécifié par l'option **grub2_cfg_file**.

Les options du noyau sont ajoutées à la configuration actuelle de GRUB et à ses modèles. Le système doit être redémarré pour que les options du noyau prennent effet.

Le passage à un autre profil ou l'arrêt manuel du service **TuneD** supprime les options supplémentaires. Si vous arrêtez ou redémarrez le système, les options du noyau persistent dans le fichier **grub.cfg**.

Les options du noyau peuvent être spécifiées par la syntaxe suivante :

```
cmdline=arg1 arg2 ... argN
```

Exemple 2.7. Modifier la ligne de commande du noyau

Par exemple, pour ajouter l'option de noyau **quiet** à un profil **TuneD**, incluez les lignes suivantes dans le fichier **tuned.conf**:

```
[bootloader]
cmdline=quiet
```

Voici un exemple de profil personnalisé qui ajoute l'option **isolcpus=2** à la ligne de commande du noyau :

```
[bootloader]
cmdline=isolcpus=2
```

2.9. VARIABLES DANS LES PROFILS TUNED

Les variables se développent au moment de l'exécution lorsqu'un profil **TuneD** est activé.

L'utilisation des variables **TuneD** réduit la quantité de données à saisir dans les profils **TuneD**.

Il n'y a pas de variables prédéfinies dans les profils **TuneD**. Vous pouvez définir vos propres variables en créant la section **[variables]** dans un profil et en utilisant la syntaxe suivante :

```
[variables]

variable_name=value
```

Pour développer la valeur d'une variable dans un profil, utilisez la syntaxe suivante :

```
${variable_name}
```

Exemple 2.8. Isolation des cœurs de l'unité centrale à l'aide de variables

Dans l'exemple suivant, la variable **${isolated_cores}** se développe en **1,2**; le noyau démarre donc avec l'option **isolcpus=1,2**:

```
[variables]
isolated_cores=1,2

[bootloader]
cmdline=isolcpus=${isolated_cores}
```

Les variables peuvent être spécifiées dans un fichier séparé. Par exemple, vous pouvez ajouter les lignes suivantes à **tuned.conf**:

```
[variables]
include=/etc/tuned/my-variables.conf

[bootloader]
cmdline=isolcpus=${isolated_cores}
```

Si vous ajoutez l'option **isolated_cores=1,2** au fichier **/etc/tuned/my-variables.conf**, le noyau démarre avec l'option **isolcpus=1,2**.

Ressources supplémentaires

- **tuned.conf(5)** page de manuel

2.10. FONCTIONS INTÉGRÉES DANS LES PROFILS TUNED

Les fonctions intégrées se développent en cours d'exécution lorsqu'un profil **TuneD** est activé.

Vous pouvez :

- Utiliser diverses fonctions intégrées avec des variables **TuneD**
- Créer des fonctions personnalisées en Python et les ajouter à **TuneD** sous forme de plug-ins

Pour appeler une fonction, utilisez la syntaxe suivante :

```
${f :function_name:argument_1:argument_2}
```

Pour développer le chemin du répertoire où se trouvent le profil et le fichier **tuned.conf**, utilisez la fonction **PROFILE_DIR**, qui requiert une syntaxe spéciale :

```
${i:PROFILE_DIR}
```

Exemple 2.9. Isolation des cœurs de processeur à l'aide de variables et de fonctions intégrées

Dans l'exemple suivant, la variable **${non_isolated_cores}** se développe en **0,3-5** et la fonction intégrée **cpulist_invert** est appelée avec l'argument **0,3-5**:

```
[variables]
non_isolated_cores=0,3-5

[bootloader]
cmdline=isolcpus=${f:cpulist_invert:${non_isolated_cores}}
```

La fonction **cpulist_invert** inverse la liste des unités centrales. Pour une machine à 6 CPU, l'inversion est **1,2**, et le noyau démarre avec l'option de ligne de commande **isolcpus=1,2**.

Ressources supplémentaires

- **tuned.conf(5)** page de manuel

2.11. FONCTIONS INTÉGRÉES DISPONIBLES DANS LES PROFILS TUNED

Les fonctions intégrées suivantes sont disponibles dans tous les profils **TuneD**:

PROFILE_DIR

Renvoie le chemin du répertoire où se trouvent le profil et le fichier **tuned.conf**.

exec

Exécute un processus et renvoie sa sortie.

assertion

Compare deux arguments. S'ils sont *do not match*, la fonction enregistre le texte du premier argument et interrompt le chargement du profil.

assertion_non_equal

Compare deux arguments. S'ils sont *match*, la fonction enregistre le texte du premier argument et interrompt le chargement du profil.

kb2s

Convertit les kilo-octets en secteurs de disque.

s2kb

Convertit les secteurs du disque en kilo-octets.

strip

Crée une chaîne de caractères à partir de tous les arguments passés et supprime les espaces blancs en début et en fin de chaîne.

virt_check

Vérifie si **TuneD** s'exécute à l'intérieur d'une machine virtuelle (VM) ou sur du métal nu :

- À l'intérieur d'une VM, la fonction renvoie le premier argument.
- Sur le métal nu, la fonction renvoie le deuxième argument, même en cas d'erreur.

cpulist_invert

Inverse une liste d'unités centrales pour obtenir son complément. Par exemple, sur un système avec 4 CPU, numérotés de 0 à 3, l'inversion de la liste **0,2,3** est **1**.

cpulist2hex

Convertit une liste de CPU en un masque de CPU hexadécimal.

cpulist2hex_invert

Convertit une liste de CPU en un masque de CPU hexadécimal et l'inverse.

hex2cpulist

Convertit un masque de CPU hexadécimal en une liste de CPU.

cpulist_online

Vérifie si les unités centrales de la liste sont en ligne. Renvoie la liste contenant uniquement les unités centrales en ligne.

cpulist_present

Vérifie si les unités centrales de la liste sont présentes. Renvoie la liste contenant uniquement les unités centrales présentes.

cpulist_unpack

Décompresse une liste de CPU sous la forme de **1-3,4** à **1,2,3,4**.

cpulist_pack

Fournit une liste d'unités centrales sous la forme de **1,2,3,5** à **1-3,5**.

2.12. CRÉATION DE NOUVEAUX PROFILS TUNED

Cette procédure crée un nouveau profil **TuneD** avec des règles de performance personnalisées.

Conditions préalables

- Le service **TuneD** est en cours d'exécution. Voir Installation et activation de TuneD pour plus de détails.

Procédure

1. Dans le répertoire /**etc**/**tuned**/, créez un nouveau répertoire portant le même nom que le profil que vous souhaitez créer :

```
# mkdir /etc/tuned/my-profile
```

2. Dans le nouveau répertoire, créez un fichier nommé **tuned.conf**. Ajoutez-y une section **[main]** et des définitions de plug-ins, en fonction de vos besoins.
Par exemple, voir la configuration du profil **balanced**:

```
[main]
summary=General non-specialized TuneD profile

[cpu]
governor=conservative
energy_perf_bias=normal

[audio]
timeout=10
```

```
[video]
radeon_powersave=dpm-balanced, auto

[scsi_host]
alpm=medium_power
```

3. Pour activer le profil, utilisez :

```
# tuned-adm profile my-profile
```

4. Vérifiez que le profil **TuneD** est actif et que les paramètres du système sont appliqués :

```
$ tuned-adm active

Current active profile: my-profile
```

```
$ tuned-adm verify

Verification succeeded, current system settings match the preset profile.
See tuned log file ('/var/log/tuned/tuned.log') for details.
```

Ressources supplémentaires

- **tuned.conf(5)** page de manuel

2.13. MODIFIER LES PROFILS TUNED EXISTANTS

Cette procédure permet de créer un profil enfant modifié sur la base d'un profil **TuneD** existant.

Conditions préalables

- Le service **TuneD** est en cours d'exécution. Voir Installation et activation de TuneD pour plus de détails.

Procédure

1. Dans le répertoire **/etc/tuned/**, créez un nouveau répertoire portant le même nom que le profil que vous souhaitez créer :

```
# mkdir /etc/tuned/modified-profile
```

2. Dans le nouveau répertoire, créez un fichier nommé **tuned.conf**, et définissez la section **[main]** comme suit :

```
[main]
include=parent-profile
```

   Remplacer *parent-profile* par le nom du profil que vous modifiez.

3. Inclure les modifications de votre profil.

Exemple 2.10. Diminution de la permutation dans le profil débit-performance

Pour utiliser les paramètres du profil **throughput-performance** et modifier la valeur de **vm.swappiness** à 5, au lieu de la valeur par défaut de 10, utilisez :

```
[main]
include=throughput-performance

[sysctl]
vm.swappiness=5
```

4. Pour activer le profil, utilisez :

```
# tuned-adm profile modified-profile
```

5. Vérifiez que le profil **TuneD** est actif et que les paramètres du système sont appliqués :

```
$ tuned-adm active

Current active profile: my-profile
```

```
$ tuned-adm verify

Verification succeeded, current system settings match the preset profile.
See tuned log file ('/var/log/tuned/tuned.log') for details.
```

Ressources supplémentaires

- **tuned.conf(5)** page de manuel

2.14. PARAMÉTRAGE DU PLANIFICATEUR DE DISQUE À L'AIDE DE TUNED

Cette procédure permet de créer et d'activer un profil **TuneD** qui définit un planificateur de disque donné pour les périphériques de bloc sélectionnés. Le paramètre persiste lors des redémarrages du système.

Dans les commandes et la configuration suivantes, remplacer :

- *device* avec le nom du dispositif de blocage, par exemple **sdf**
- *selected-scheduler* avec le planificateur de disque que vous souhaitez définir pour le périphérique, par exemple **bfq**

Conditions préalables

- Le service **TuneD** est installé et activé. Pour plus de détails, voir Installation et activation de TuneD.

Procédure

1. Facultatif : Sélectionnez un profil **TuneD** existant sur lequel votre profil sera basé. Pour obtenir une liste des profils disponibles, voir les profils TuneD distribués avec RHEL .

Pour savoir quel profil est actuellement actif, utilisez :

```
$ tuned-adm active
```

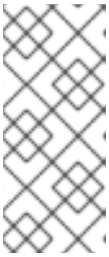
2. Créez un nouveau répertoire qui contiendra votre profil **TuneD**:

```
# mkdir /etc/tuned/my-profile
```

3. Recherchez l'identifiant unique du système du bloc sélectionné :

```
$ udevadm info --query=property --name=/dev/device | grep -E '(WWN|SERIAL)'

ID_WWN=0x5002538d00000000_
ID_SERIAL=Generic-_SD_MMC_20120501030900000-0:0
ID_SERIAL_SHORT=20120501030900000
```



NOTE

La commande de cet exemple renverra toutes les valeurs identifiées par un World Wide Name (WWN) ou un numéro de série associé au dispositif de bloc spécifié. Bien qu'il soit préférable d'utiliser un WWN, celui-ci n'est pas toujours disponible pour un dispositif donné et toutes les valeurs renvoyées par la commande de l'exemple peuvent être utilisées comme *device system unique ID*.

4. Créer le fichier de **/etc/tuned/*my-profile*/tuned.conf** fichier de configuration. Dans le fichier, définissez les options suivantes :

   a. Facultatif : Inclure un profil existant :

   ```
   [main]
   include=existing-profile
   ```

   b. Définir le planificateur de disque sélectionné pour le périphérique qui correspond à l'identifiant WWN :

   ```
   [disk]
   devices_udev_regex=IDNAME=device system unique id
   elevator=selected-scheduler
   ```

   Ici :

   - Remplacer *IDNAME* par le nom de l'identifiant utilisé (par exemple, **ID_WWN**).
   - Remplacer *device system unique id* par la valeur de l'identifiant choisi (par exemple, **0x5002538d00000000**).
     Pour faire correspondre plusieurs appareils dans l'option **devices_udev_regex**, mettez les identifiants entre parenthèses et séparez-les par des barres verticales :

     ```
     devices_udev_regex=(ID_WWN=0x5002538d00000000)|
     (ID_WWN=0x1234567800000000)
     ```

5. Activez votre profil :

```
# tuned-adm profile my-profile
```

Verification steps

1. Vérifiez que le profil TuneD est actif et appliqué :

```
$ tuned-adm active

Current active profile: my-profile
```

```
$ tuned-adm verify

Verification succeeded, current system settings match the preset profile.
See TuneD log file ('/var/log/tuned/tuned.log') for details.
```

2. Lire le contenu du **/sys/block/*device*/queue/scheduler** fichier :

```
# cat /sys/block/device/queue/scheduler

[mq-deadline] kyber bfq none
```

Dans le nom du fichier, remplacez *device* par le nom du bloc, par exemple **sdc**.

Le planificateur actif est indiqué entre crochets (**[]**).

Ressources supplémentaires

- Personnalisation des profils TuneD.

CHAPITRE 3. RÉVISION D'UN SYSTÈME À L'AIDE DE L'INTERFACE TUNA

Utilisez l'outil **tuna** pour ajuster les paramètres de l'ordonnanceur, la priorité des threads, les gestionnaires d'IRQ et pour isoler les cœurs et les sockets de l'unité centrale. Tuna réduit la complexité des tâches de réglage.

L'outil **tuna** effectue les opérations suivantes :

- Liste des unités centrales d'un système
- Répertorie les demandes d'interruption (IRQ) en cours sur un système
- Modifie les informations relatives à la politique et à la priorité sur les fils de discussion
- Affiche les politiques et priorités actuelles d'un système

3.1. INSTALLATION DE L'OUTIL THON

L'outil **tuna** est conçu pour être utilisé sur un système en fonctionnement. Cela permet aux outils de mesure spécifiques à une application de voir et d'analyser les performances du système immédiatement après que des modifications ont été apportées.

Cette procédure décrit comment installer l'outil **tuna**.

Procédure

- Installer l'outil **tuna**:

```
# dnf install tuna
```

Verification steps

- Voir les options CLI disponibles sur **tuna**:

```
# tuna -h
```

Ressources supplémentaires

- **tuna(8)** page de manuel

3.2. VISUALISATION DE L'ÉTAT DU SYSTÈME À L'AIDE DE L'OUTIL TUNA

Cette procédure décrit comment visualiser l'état du système à l'aide de l'outil d'interface de ligne de commande (CLI) **tuna**.

Conditions préalables

- L'outil tuna est installé. Pour plus d'informations, voir Installation de l'outil tuna.

Procédure

- Pour consulter les politiques et priorités actuelles :

```
# tuna --show_threads
            thread
pid   SCHED_ rtpri affinity            cmd
1     OTHER     0      0,1            init
2      FIFO    99        0     migration/0
3     OTHER     0        0     ksoftirqd/0
4      FIFO    99        0      watchdog/0
```

- Pour visualiser un fil spécifique correspondant à un PID ou à un nom de commande :

```
# tuna --threads=pid_or_cmd_list --show_threads
```

L'argument *pid_or_cmd_list* est une liste de PID ou de noms de commande séparés par des virgules.

- Pour régler les processeurs à l'aide de l'interface CLI de **tuna**, voir Réglage des processeurs à l'aide de l'outil tuna.
- Pour régler les IRQ à l'aide de l'outil **tuna**, voir Réglage des IRQ à l'aide de l'outil tuna .
- Pour enregistrer la configuration modifiée :

```
# tuna --save=filename
```

Cette commande ne sauvegarde que les threads du noyau en cours d'exécution. Les processus qui ne sont pas en cours d'exécution ne sont pas sauvegardés.

Ressources supplémentaires

- **tuna(8)** page de manuel

3.3. OPTIMISATION DES PROCESSEURS À L'AIDE DE L'OUTIL TUNA

Les commandes de l'outil **tuna** peuvent cibler des unités centrales individuelles.

En utilisant l'outil thon, vous pouvez

Isolate CPUs

Toutes les tâches exécutées sur l'unité centrale spécifiée se déplacent vers la prochaine unité centrale disponible. L'isolation d'une unité centrale la rend indisponible en la supprimant du masque d'affinité de tous les threads.

Include CPUs

Permet aux tâches de s'exécuter sur l'unité centrale spécifiée

Restore CPUs

Rétablit la configuration précédente de l'unité centrale spécifiée.

Cette procédure décrit comment régler les CPU à l'aide de l'interface CLI de **tuna**.

Conditions préalables

- L'outil tuna est installé. Pour plus d'informations, voir Installation de l'outil tuna.

Procédure

- Pour spécifier la liste des unités centrales devant être affectées par une commande :

```
# tuna --cpus=cpu_list [command]
```

L'argument *cpu_list* est une liste de numéros de CPU séparés par des virgules. Par exemple, **--cpus=*0,2***. Les listes d'unités centrales peuvent également être spécifiées dans une plage, par exemple **--cpus="*1-3*"**qui sélectionnerait les unités centrales 1, 2 et 3.

Pour ajouter une unité centrale spécifique à l'actuelle *cpu_list*, par exemple, utilisez **--cpus= 0**.

Remplacer [*command*] par, par exemple, **--isolate**.

- Pour isoler une unité centrale :

```
# tuna --cpus=cpu_list --isolate
```

- Pour inclure une unité centrale :

```
# tuna --cpus=cpu_list --include
```

- Pour utiliser un système à quatre processeurs ou plus, montrez comment faire en sorte que tous les threads ssh s'exécutent sur les unités centrales *0* et *1*, et tous les threads **http** sur les unités centrales *2* et *3*:

```
# tuna --cpus=0,1 --threads=ssh\* \
--move --cpus=2,3 --threads=http\* --move
```

Cette commande permet d'effectuer les opérations suivantes de manière séquentielle :

1. Sélectionne les unités centrales *0* et *1*.
2. Sélectionne tous les fils qui commencent par **ssh**.
3. Déplace les threads sélectionnés vers les unités centrales sélectionnées. Tuna définit le masque d'affinité des threads commençant par **ssh** vers les CPU appropriés. Les CPU peuvent être exprimés numériquement par *0* et *1*, en masque hexagonal par 0x3, ou en binaire par 11.
4. Réinitialise la liste des unités centrales à *2* et *3*.
5. Sélectionne tous les fils qui commencent par **http**.
6. Déplace les threads sélectionnés vers les unités centrales spécifiées. Tuna définit le masque d'affinité des threads commençant par **http** vers les unités centrales spécifiées. Les unités centrales peuvent être exprimées numériquement par *2* et *3*, en masque hexagonal par 0xC, ou en binaire par 1100.

Verification steps

- Affichez la configuration actuelle et vérifiez que les modifications ont été effectuées comme prévu :

```
# tuna --threads=gnome-sc\* --show_threads \
```

```
--cpus=0 --move --show_threads --cpus=1 \
--move --show_threads --cpus=+0 --move --show_threads

                  thread       ctxt_switches
    pid SCHED_ rtpri affinity voluntary nonvoluntary            cmd
   3861   OTHER     0      0,1     33997           58 gnome-screensav
                  thread       ctxt_switches
    pid SCHED_ rtpri affinity voluntary nonvoluntary            cmd
   3861   OTHER     0        0     33997           58 gnome-screensav
                  thread       ctxt_switches
    pid SCHED_ rtpri affinity voluntary nonvoluntary            cmd
   3861   OTHER     0        1     33997           58 gnome-screensav
                  thread       ctxt_switches
    pid SCHED_ rtpri affinity voluntary nonvoluntary            cmd
   3861   OTHER     0      0,1     33997           58 gnome-screensav
```

Cette commande permet d'effectuer les opérations suivantes de manière séquentielle :

1. Sélectionne tous les fils qui commencent par les fils **gnome-sc**.
2. Affiche les threads sélectionnés pour permettre à l'utilisateur de vérifier leur masque d'affinité et leur priorité RT.
3. Sélectionne l'unité centrale *0*.
4. Déplace les threads **gnome-sc** vers l'unité centrale spécifiée, l'unité centrale *0*.
5. Affiche le résultat du déplacement.
6. Réinitialise la liste des CPU à CPU *1*.
7. Déplace les threads **gnome-sc** vers l'unité centrale spécifiée, l'unité centrale *1*.
8. Affiche le résultat du déplacement.
9. Ajoute l'unité centrale *0* à la liste des unités centrales.
10. Déplace les threads **gnome-sc** vers les unités centrales spécifiées, les unités centrales *0* et *1*.
11. Affiche le résultat du déplacement.

Ressources supplémentaires

- **/proc/cpuinfo** fichier
- **tuna(8)** page de manuel

3.4. RÉGLAGE DES IRQ À L'AIDE DE L'OUTIL TUNA

Le fichier **/proc/interrupts** enregistre le nombre d'interruptions par IRQ, le type d'interruption et le nom du périphérique situé à cette IRQ.

Cette procédure décrit comment régler les IRQ à l'aide de l'outil **tuna**.

Conditions préalables

- L'outil tuna est installé. Pour plus d'informations, voir Installation de l'outil tuna.

Procédure

- Pour afficher les IRQ en cours et leur affinité :

```
# tuna --show_irqs
# users          affinity
0 timer              0
1 i8042              0
7 parport0           0
```

- Pour spécifier la liste des IRQ à affecter par une commande :

```
# tuna --irqs=irq_list [command]
```

L'argument *irq_list* est une liste de numéros d'IRQ ou de noms d'utilisateurs séparés par des virgules.

Remplacer [*command*] par, par exemple, **--spread**.

- Pour déplacer une interruption vers une unité centrale spécifiée :

```
# tuna --irqs=128 --show_irqs
  # users          affinity
128 iwlwifi          0,1,2,3

# tuna --irqs=128 --cpus=3 --move
```

Remplacez *128* par l'argument irq_list et *3* par l'argument cpu_list.

L'argument *cpu_list* est une liste de numéros de CPU séparés par des virgules, par exemple, **--cpus=*0,2***. Pour plus d'informations, voir Tuning CPUs using tuna tool.

Verification steps

- Comparer l'état des IRQ sélectionnées avant et après le déplacement d'une interruption vers une unité centrale spécifiée :

```
# tuna --irqs=128 --show_irqs
  # users          affinity
128 iwlwifi              3
```

Ressources supplémentaires

- **/procs/interrupts** fichier
- **tuna(8)** page de manuel

CHAPITRE 4. SURVEILLANCE DES PERFORMANCES À L'AIDE DES RÔLES SYSTÈME RHEL

En tant qu'administrateur système, vous pouvez utiliser le rôle système **metrics** RHEL pour surveiller les performances d'un système.

4.1. PRÉPARATION D'UN NŒUD DE CONTRÔLE ET DE NŒUDS GÉRÉS À L'UTILISATION DES RÔLES SYSTÈME RHEL

Avant de pouvoir utiliser des rôles système RHEL individuels pour gérer des services et des paramètres, vous devez préparer le nœud de contrôle et les nœuds gérés.

4.1.1. Préparation d'un nœud de contrôle sur RHEL 9

Avant d'utiliser RHEL System Roles, vous devez configurer un nœud de contrôle. Ce système configure ensuite les hôtes gérés à partir de l'inventaire conformément aux playbooks.

Conditions préalables

- RHEL 8.6 ou une version ultérieure est installée. Pour plus d'informations sur l'installation de RHEL, voir Effectuer une installation standard de RHEL 9 .
- Le système est enregistré sur le portail client.
- Un abonnement **Red Hat Enterprise Linux Server** est attaché au système.
- S'il est disponible dans votre compte Portail Client, un abonnement **Ansible Automation Platform** est attaché au système.

Procédure

1. Installez le paquetage **rhel-system-roles**:

```
[root@control-node]# dnf install rhel-system-roles
```

   Cette commande installe le paquet **ansible-core** en tant que dépendance.

NOTE

Dans RHEL 8.5 et les versions antérieures, les packages Ansible étaient fournis via Ansible Engine au lieu d'Ansible Core, et avec un niveau de support différent. N'utilisez pas Ansible Engine car les packages peuvent ne pas être compatibles avec le contenu d'automatisation Ansible dans RHEL 8.6 et les versions ultérieures. Pour plus d'informations, voir Étendue de la prise en charge du package Ansible Core inclus dans les référentiels AppStream RHEL 9 et RHEL 8.6 et versions ultérieures.

2. Créez un utilisateur nommé **ansible** pour gérer et exécuter les playbooks :

```
[root@control-node]# useradd ansible
```

3. Passez à l'utilisateur nouvellement créé **ansible**:

```
[root@control-node]# su - ansible
```

Effectuez le reste de la procédure en tant qu'utilisateur.

4. Créez une clé publique et une clé privée SSH :

```
[ansible@control-node]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ansible/.ssh/id_rsa): <password>
...
```

Utilisez l'emplacement par défaut proposé pour le fichier clé.

5. Facultatif : Pour éviter qu'Ansible ne vous demande le mot de passe de la clé SSH à chaque fois que vous établissez une connexion, configurez un agent SSH.

6. Créez le fichier ~/**.ansible.cfg** avec le contenu suivant :

```
[defaults]
inventory = /home/ansible/inventory
remote_user = ansible

[privilege_escalation]
become = True
become_method = sudo
become_user = root
become_ask_pass = True
```

NOTE

Les paramètres du fichier ~/**.ansible.cfg** ont une priorité plus élevée et remplacent les paramètres du fichier global /**etc**/**ansible**/**ansible.cfg**.

Avec ces paramètres, Ansible effectue les actions suivantes :

- Gère les hôtes dans le fichier d'inventaire spécifié.
- Utilise le compte défini dans le paramètre **remote_user** lorsqu'il établit des connexions SSH avec les nœuds gérés.
- Utilise l'utilitaire **sudo** pour exécuter des tâches sur les nœuds gérés en tant qu'utilisateur **root**.
- Demande le mot de passe root de l'utilisateur distant à chaque fois que vous appliquez un playbook. Ceci est recommandé pour des raisons de sécurité.

7. Créez un fichier ~/**inventory** au format INI ou YAML qui répertorie les noms d'hôtes gérés. Vous pouvez également définir des groupes d'hôtes dans le fichier d'inventaire. Par exemple, voici un fichier d'inventaire au format INI avec trois hôtes et un groupe d'hôtes nommé **US**:

```
managed-node-01.example.com

[US]
managed-node-02.example.com ansible_host=192.0.2.100
managed-node-03.example.com
```

- Notez que le nœud de contrôle doit être en mesure de résoudre les noms d'hôte. Si le serveur DNS ne peut pas résoudre certains noms d'hôtes, ajoutez le paramètre **ansible_host** à côté de l'entrée de l'hôte pour spécifier son adresse IP.

Prochaines étapes

- Préparez les nœuds gérés. Pour plus d'informations, voir Préparation d'un nœud géré .

Ressources supplémentaires

- Étendue de la prise en charge du package Ansible Core inclus dans les référentiels AppStream RHEL 9 et RHEL 8.6 et versions ultérieures
- Comment enregistrer et abonner un système au portail client de Red Hat à l'aide du gestionnaire d'abonnements ?
- La page de manuel **ssh-keygen(1)**
- Se connecter à des machines distantes avec des clés SSH en utilisant ssh-agent
- Paramètres de configuration d'Ansible
- Comment constituer votre inventaire

4.1.2. Préparation d'un nœud géré

Les nœuds gérés sont les systèmes répertoriés dans l'inventaire et qui seront configurés par le nœud de contrôle conformément au cahier de jeu. Il n'est pas nécessaire d'installer Ansible sur les hôtes gérés.

Conditions préalables

- Vous avez préparé le nœud de contrôle. Pour plus d'informations, voir Préparation d'un nœud de contrôle sur RHEL 9.
- Vous disposez d'un accès SSH à partir du nœud de contrôle.

IMPORTANT

L'accès SSH direct en tant qu'utilisateur **root** présente un risque pour la sécurité. Pour réduire ce risque, vous créerez un utilisateur local sur ce nœud et configurerez une politique **sudo** lors de la préparation d'un nœud géré. Ansible sur le nœud de contrôle peut alors utiliser le compte d'utilisateur local pour se connecter au nœud géré et exécuter des playbooks en tant qu'utilisateurs différents, tels que **root**.

Procédure

1. Créez un utilisateur nommé **ansible**:

```
[root@managed-node-01]# useradd ansible
```

Le nœud de contrôle utilise ensuite cet utilisateur pour établir une connexion SSH avec cet hôte.

2. Définir un mot de passe pour l'utilisateur **ansible**:

```
[root@managed-node-01]# passwd ansible
Changing password for user ansible.
New password: <password>
Retype new password: <password>
passwd: all authentication tokens updated successfully.
```

Vous devez saisir ce mot de passe lorsque Ansible utilise **sudo** pour effectuer des tâches en tant qu'utilisateur **root**.

3. Installez la clé publique SSH de l'utilisateur **ansible** sur le nœud géré :

   a. Connectez-vous au nœud de contrôle en tant qu'utilisateur **ansible** et copiez la clé publique SSH sur le nœud géré :

```
[ansible@control-node]$ ssh-copy-id managed-node-01.example.com
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed:
"/home/ansible/.ssh/id_rsa.pub"
The authenticity of host 'managed-node-01.example.com (192.0.2.100)' can't be
established.
ECDSA key fingerprint is
SHA256:9bZ33GJNODK3zbNhybokN/6Mq7hu3vpBXDrCxe7NAvo.
```

   b. Lorsque vous y êtes invité, connectez-vous en entrant **yes**:

```
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that
are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is
to install the new keys
```

   c. Lorsque vous y êtes invité, saisissez le mot de passe :

```
ansible@managed-node-01.example.com's password: <password>

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh '<managed-node-01.example.com>'"
and check to make sure that only the key(s) you wanted were added.
```

   d. Vérifiez la connexion SSH en exécutant à distance une commande sur le nœud de contrôle :

```
[ansible@control-node]$ ssh <managed-node-01.example.com> whoami
ansible
```

4. Créer une configuration **sudo** pour l'utilisateur **ansible**:

   a. Créez et modifiez le fichier **/etc/sudoers.d/ansible** à l'aide de la commande **visudo**:

```
[root@managed-node-01]# visudo /etc/sudoers.d/ansible
```

L'avantage d'utiliser **visudo** plutôt qu'un éditeur normal est que cet utilitaire fournit des contrôles de base et vérifie les erreurs d'analyse avant d'installer le fichier.

b. Configurez une politique **sudoers** dans le fichier **/etc/sudoers.d/ansible** qui réponde à vos besoins, par exemple :

- Pour autoriser l'utilisateur **ansible** à exécuter toutes les commandes en tant qu'utilisateur et groupe sur cet hôte après avoir saisi le mot de passe de l'utilisateur **ansible**, utilisez l'option suivante :

```
ansible ALL=(ALL) ALL
```

- Pour autoriser l'utilisateur **ansible** à exécuter toutes les commandes en tant qu'utilisateur et groupe sur cet hôte sans saisir le mot de passe de l'utilisateur **ansible**, utilisez la commande suivante

```
ansible ALL=(ALL) NOPASSWD: ALL
```

Vous pouvez également configurer une politique plus fine qui correspond à vos exigences en matière de sécurité. Pour plus d'informations sur les politiques de **sudoers**, voir la page de manuel **sudoers(5)**.

Vérification

1. Vérifiez que vous pouvez exécuter des commandes à partir du nœud de contrôle sur tous les nœuds gérés :

```
[ansible@control-node]$ ansible all -m ping
BECOME password: <password>
managed-node-01.example.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
...
```

Le groupe all codé en dur contient dynamiquement tous les hôtes répertoriés dans le fichier d'inventaire.

2. Vérifiez que l'escalade des privilèges fonctionne correctement en exécutant l'utilitaire **whoami** sur un hôte géré à l'aide du module Ansible **command**:

```
[ansible@control-node]$ ansible managed-node-01.example.com -m command -a whoami
BECOME password: <password>
managed-node-01.example.com | CHANGED | rc=0 >>
root
```

Si la commande renvoie la valeur root, vous avez configuré correctement **sudo** sur les nœuds gérés.

Ressources supplémentaires

- Préparation d'un nœud de contrôle sur RHEL 9 .
- La page de manuel **sudoers(5)**

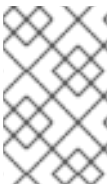
4.2. INTRODUCTION AU RÔLE DU SYSTÈME METRICS

RHEL System Roles est une collection de rôles et de modules Ansible qui fournissent une interface de configuration cohérente pour gérer à distance plusieurs systèmes RHEL. Le rôle système **metrics** configure les services d'analyse des performances pour le système local et, en option, inclut une liste de systèmes distants à surveiller par le système local. Le rôle système **metrics** vous permet d'utiliser **pcp** pour surveiller les performances de vos systèmes sans avoir à configurer **pcp** séparément, car la configuration et le déploiement de **pcp** sont gérés par le playbook.

Tableau 4.1. metrics variables de rôle du système

Variable de rôle	Description	Exemple d'utilisation
métriques_hôtes_surveillés	Liste des hôtes distants à analyser par l'hôte cible. Ces hôtes auront des métriques enregistrées sur l'hôte cible, il faut donc s'assurer qu'il y a suffisamment d'espace disque sous **/var/log** pour chaque hôte.	**metrics_monitored_hosts: ["*webserver.example.com*", "*database.example.com*"]**
jours_de_métriques_de_rétention	Configure le nombre de jours de rétention des données de performance avant leur suppression.	**metrics_retention_days: 14**
service_graphique_métriques	Indicateur booléen qui permet à l'hôte d'être configuré avec des services de visualisation de données de performance via **pcp** et **grafana**. La valeur par défaut est false.	**metrics_graph_service: no**
metrics_query_service	Indicateur booléen qui permet à l'hôte d'être configuré avec des services d'interrogation de séries temporelles pour l'interrogation des mesures enregistrées sur **pcp** via **redis**. La valeur par défaut est false.	**metrics_query_service: no**
fournisseur de métriques	Spécifie le collecteur de métriques à utiliser pour fournir des métriques. Actuellement, **pcp** est le seul fournisseur de métriques pris en charge.	**metrics_provider: "pcp"**
metrics_manage_firewall	Utilise le rôle **firewall** pour gérer l'accès aux ports directement à partir du rôle **metrics**. La valeur par défaut est false.	**metrics_manage_firewall: true**

Variable de rôle	Description	Exemple d'utilisation
metrics_manage_selinux	Utilise le rôle **selinux** pour gérer l'accès aux ports directement à partir du rôle **metrics**. La valeur par défaut est false.	**metrics_manage_selinux: true**



NOTE

Pour plus de détails sur les paramètres utilisés dans **metrics_connections** et des informations supplémentaires sur le rôle de système **metrics**, voir le fichier **/usr/share/ansible/roles/rhel-system-roles.metrics/README.md**.

4.3. UTILISATION DU RÔLE DE SYSTÈME METRICS POUR SURVEILLER VOTRE SYSTÈME LOCAL AVEC VISUALISATION

Cette procédure décrit comment utiliser le rôle de système RHEL **metrics** pour surveiller votre système local tout en fournissant une visualisation des données via **Grafana**.

Conditions préalables

- Le paquetage Ansible Core est installé sur la machine de contrôle.
- Le paquetage **rhel-system-roles** est installé sur la machine que vous voulez surveiller.

Procédure

1. Configurez **localhost** dans l'inventaire Ansible **/etc/ansible/hosts** en ajoutant le contenu suivant à l'inventaire :

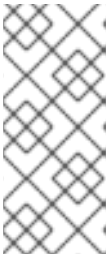
```
localhost ansible_connection=local
```

2. Créez un playbook Ansible avec le contenu suivant :

```
---
- name: Manage metrics
  hosts: localhost
  vars:
    metrics_graph_service: yes
    metrics_manage_firewall: true
    metrics_manage_selinux: true
  roles:
    - rhel-system-roles.metrics
```

3. Exécutez le playbook Ansible :

```
# ansible-playbook name_of_your_playbook.yml
```



NOTE

Comme le booléen **metrics_graph_service** est défini sur la valeur="yes", **Grafana** est automatiquement installé et provisionné avec **pcp** ajouté en tant que source de données. Comme metrics_manage_firewall et metrics_manage_selinux sont tous deux définis sur true, le rôle metrics utilisera les rôles système firewall et selinux pour gérer les ports utilisés par le rôle metrics.

4. Pour visualiser les métriques collectées sur votre machine, accédez à l'interface web **grafana** comme décrit dans Accéder à l'interface web Grafana .

4.4. L'UTILISATION DU RÔLE DE SYSTÈME METRICS PERMET DE CONFIGURER UN PARC DE SYSTÈMES INDIVIDUELS POUR QU'ILS SE SURVEILLENT EUX-MÊMES

Cette procédure décrit comment utiliser le rôle de système **metrics** pour configurer une flotte de machines afin qu'elles se surveillent elles-mêmes.

Conditions préalables

- Le paquetage Ansible Core est installé sur la machine de contrôle.
- Le paquetage **rhel-system-roles** est installé sur la machine que vous souhaitez utiliser pour exécuter le playbook.
- La connexion SSH est établie.

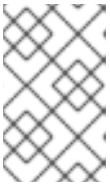
Procédure

1. Ajoutez le nom ou l'IP des machines que vous souhaitez surveiller via le playbook au fichier d'inventaire Ansible **/etc/ansible/hosts** sous un nom de groupe d'identification entre parenthèses :

```
[remotes]
webserver.example.com
database.example.com
```

2. Créez un playbook Ansible avec le contenu suivant :

```
---
- hosts: remotes
  vars:
    metrics_retention_days: 0
    metrics_manage_firewall: true
    metrics_manage_selinux: true
  roles:
    - rhel-system-roles.metrics
```



NOTE

Puisque **metrics_manage_firewall** et **metrics_manage_selinux** sont tous deux définis sur true, le rôle metrics utilisera les rôles **firewall** et **selinux** pour gérer les ports utilisés par le rôle **metrics**.

3. Exécutez le playbook Ansible :

```
# ansible-playbook name_of_your_playbook.yml -k
```

Lorsque le site **-k** demande un mot de passe pour se connecter au système distant.

4.5. UTILISATION DU RÔLE DE SYSTÈME METRICS POUR SURVEILLER UN PARC DE MACHINES DE MANIÈRE CENTRALISÉE VIA VOTRE MACHINE LOCALE

Cette procédure décrit comment utiliser le rôle de système **metrics** pour configurer votre machine locale afin de surveiller de manière centralisée un parc de machines, tout en prévoyant la visualisation des données via **grafana** et l'interrogation des données via **redis**.

Conditions préalables

- Le paquetage Ansible Core est installé sur la machine de contrôle.
- Le paquetage **rhel-system-roles** est installé sur la machine que vous souhaitez utiliser pour exécuter le playbook.

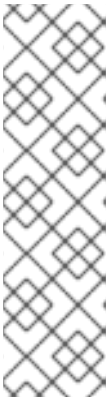
Procédure

1. Créez un playbook Ansible avec le contenu suivant :

```
---
- hosts: localhost
  vars:
    metrics_graph_service: yes
    metrics_query_service: yes
    metrics_retention_days: 10
    metrics_monitored_hosts: ["database.example.com", "webserver.example.com"]
    metrics_manage_firewall: yes
    metrics_manage_selinux: yes
  roles:
    - rhel-system-roles.metrics
```

2. Exécutez le playbook Ansible :

```
# ansible-playbook name_of_your_playbook.yml
```



NOTE

Étant donné que les booléens **metrics_graph_service** et **metrics_query_service** ont la valeur "yes", **grafana** est automatiquement installé et approvisionné avec **pcp** ajouté en tant que source de données, l'enregistrement des données **pcp** étant indexé dans **redis**, ce qui permet d'utiliser le langage d'interrogation **pcp** pour effectuer des requêtes complexes sur les données. Étant donné que **metrics_manage_firewall** et **metrics_manage_selinux** sont tous deux définis comme vrais, le rôle **metrics** utilisera les rôles **firewall** et **selinux** pour gérer les ports utilisés par le rôle **metrics**.

3. Pour afficher une représentation graphique des métriques collectées de manière centralisée par votre machine et pour interroger les données, accédez à l'interface web **grafana** comme décrit dans Accéder à l'interface web Grafana .

4.6. CONFIGURATION DE L'AUTHENTIFICATION LORS DE LA SURVEILLANCE D'UN SYSTÈME À L'AIDE DU RÔLE DE SYSTÈME METRICS

PCP prend en charge le mécanisme d'authentification **scram-sha-256** par le biais du cadre SASL (Simple Authentication Security Layer). Le rôle système **metrics** RHEL automatise les étapes de configuration de l'authentification à l'aide du mécanisme d'authentification **scram-sha-256**. Cette procédure décrit comment configurer l'authentification à l'aide du rôle système **metrics** RHEL.

Conditions préalables

- Le paquetage Ansible Core est installé sur la machine de contrôle.
- Le paquetage **rhel-system-roles** est installé sur la machine que vous souhaitez utiliser pour exécuter le playbook.

Procédure

1. Incluez les variables suivantes dans le playbook Ansible pour lequel vous souhaitez configurer l'authentification :

```
---
 vars:
   metrics_username: your_username
   metrics_password: your_password
   metrics_manage_firewall: true
   metrics_manage_selinux: true
```

NOTE

Puisque **metrics_manage_firewall** et **metrics_manage_selinux** sont tous deux définis sur true, le rôle **metrics** utilisera les rôles **firewall** et **selinux** pour gérer les ports utilisés par le rôle **metrics**.

2. Exécutez le playbook Ansible :

```
# ansible-playbook name_of_your_playbook.yml
```

Verification steps

- Vérifiez la configuration de **sasl**:

```
# pminfo -f -h "pcp://ip_adress?username=your_username" disk.dev.read
Password:
disk.dev.read
inst [0 or "sda"] value 19540
```

ip_adress doit être remplacé par l'adresse IP de l'hôte.

4.7. UTILISATION DU RÔLE DE SYSTÈME METRICS POUR CONFIGURER ET ACTIVER LA COLLECTE DE MÉTRIQUES POUR SQL SERVER

Cette procédure décrit comment utiliser le rôle système **metrics** RHEL pour automatiser la configuration et l'activation de la collecte de métriques pour Microsoft SQL Server via **pcp** sur votre système local.

Conditions préalables

- Le paquetage Ansible Core est installé sur la machine de contrôle.
- Le paquetage **rhel-system-roles** est installé sur la machine que vous voulez surveiller.
- Vous avez installé Microsoft SQL Server pour Red Hat Enterprise Linux et établi une connexion "fiable" à un serveur SQL. Voir Installer SQL Server et créer une base de données sur Red Hat .
- Vous avez installé le pilote Microsoft ODBC pour SQL Server pour Red Hat Enterprise Linux. Voir Red Hat Enterprise Server et Oracle Linux .

Procédure

1. Configurez **localhost** dans l'inventaire Ansible **/etc/ansible/hosts** en ajoutant le contenu suivant à l'inventaire :

```
localhost ansible_connection=local
```

2. Créez un playbook Ansible contenant les éléments suivants :

```
---
- hosts: localhost
  vars:
   metrics_from_mssql: true
   metrics_manage_firewall: true
   metrics_manage_selinux: true
  roles:
   - role: rhel-system-roles.metrics
```

NOTE

Puisque **metrics_manage_firewall** et **metrics_manage_selinux** sont tous deux définis sur true, le rôle **metrics** utilisera les rôles **firewall** et **selinux** pour gérer les ports utilisés par le rôle **metrics**.

3. Exécutez le playbook Ansible :

```
# ansible-playbook name_of_your_playbook.yml
```

Verification steps

- Utilisez la commande **pcp** pour vérifier que l'agent PMDA du serveur SQL (mssql) est chargé et en cours d'exécution :

```
# pcp
```

```
platform: Linux rhel82-2.local 4.18.0-167.el8.x86_64 #1 SMP Sun Dec 15 01:24:23 UTC
2019 x86_64
 hardware: 2 cpus, 1 disk, 1 node, 2770MB RAM
 timezone: PDT+7
 services: pmcd pmproxy
    pmcd: Version 5.0.2-1, 12 agents, 4 clients
    pmda: root pmcd proc pmproxy xfs linux nfsclient mmv kvm mssql
          jbd2 dm
 pmlogger: primary logger: /var/log/pcp/pmlogger/rhel82-2.local/20200326.16.31
    pmie: primary engine: /var/log/pcp/pmie/rhel82-2.local/pmie.log
```

Ressources supplémentaires

- Pour plus d'informations sur l'utilisation de Performance Co-Pilot pour Microsoft SQL Server, consultez ce billet du Red Hat Developers Blog.

CHAPITRE 5. MISE EN PLACE DU PCP

Performance Co-Pilot (PCP) est une suite d'outils, de services et de bibliothèques permettant de surveiller, de visualiser, de stocker et d'analyser les mesures de performance au niveau du système.

5.1. VUE D'ENSEMBLE DU PCP

Vous pouvez ajouter des mesures de performance à l'aide des interfaces Python, Perl, C et C. Les outils d'analyse peuvent utiliser directement les API client Python, C , C, et les applications web riches peuvent explorer toutes les données de performance disponibles à l'aide d'une interface JSON.

Vous pouvez analyser les modèles de données en comparant les résultats en temps réel avec les données archivées.

Caractéristiques du PCP :

- Architecture distribuée légère, utile pour l'analyse centralisée de systèmes complexes.
- Il permet le suivi et la gestion des données en temps réel.
- Il permet d'enregistrer et d'extraire des données historiques.

Le PCP comprend les éléments suivants :

- Le Performance Metric Collector Daemon (**pmcd**) collecte les données de performance à partir des Performance Metric Domain Agents (**pmda**) installés. **PMDAs** peut être chargé ou déchargé individuellement sur le système et est contrôlé par **PMCD** sur le même hôte.
- Divers outils clients, tels que **pminfo** ou **pmstat**, peuvent récupérer, afficher, archiver et traiter ces données sur le même hôte ou sur le réseau.
- Le paquetage **pcp** fournit les outils en ligne de commande et les fonctionnalités sous-jacentes.
- Le paquetage **pcp-gui** fournit l'application graphique. Installez le paquetage **pcp-gui** en exécutant la commande **dnf install pcp-gui**. Pour plus d'informations, voir Tracer visuellement les archives de journaux PCP avec l'application PCP Charts.

Ressources supplémentaires

- **pcp(1)** page de manuel
- **/usr/share/doc/pcp-doc/** répertoire
- Outils distribués avec le PCP
- Index des articles, solutions, tutoriels et livres blancs de Performance Co-Pilot (PCP) sur le portail client de Red Hat
- Comparaison côte à côte des outils PCP avec les outils existants Article de la base de connaissances de Red Hat
- Documentation en amont du PCP

5.2. INSTALLATION ET ACTIVATION DE PCP

Pour commencer à utiliser PCP, installez tous les paquets nécessaires et activez les services de surveillance PCP.

Cette procédure décrit comment installer PCP à l'aide du paquetage **pcp**. Si vous souhaitez automatiser l'installation de PCP, installez-le à l'aide du paquetage **pcp-zeroconf**. Pour plus d'informations sur l'installation de PCP à l'aide de **pcp-zeroconf**, voir Configuration de PCP avec pcp-zeroconf .

Procédure

1. Installez le paquetage **pcp**:

```
# dnf install pcp
```

2. Activez et démarrez le service **pmcd** sur la machine hôte :

```
# systemctl enable pmcd

# systemctl start pmcd
```

Verification steps

- Vérifiez si le processus **pmcd** est en cours d'exécution sur l'hôte :

```
# pcp

Performance Co-Pilot configuration on workstation:

platform: Linux workstation 4.18.0-80.el8.x86_64 #1 SMP Wed Mar 13 12:02:46 UTC 2019
x86_64
hardware: 12 cpus, 2 disks, 1 node, 36023MB RAM
timezone: CEST-2
services: pmcd
pmcd: Version 4.3.0-1, 8 agents
pmda: root pmcd proc xfs linux mmv kvm jbd2
```

Ressources supplémentaires

- **pmcd(1)** page de manuel
- Outils distribués avec le PCP

5.3. DÉPLOYER UNE CONFIGURATION PCP MINIMALE

L'installation minimale de PCP permet de collecter des statistiques de performance sur Red Hat Enterprise Linux. L'installation consiste à ajouter le nombre minimum de paquets sur un système de production nécessaire pour collecter des données en vue d'une analyse plus approfondie.

Vous pouvez analyser le fichier **tar.gz** résultant et l'archive de la sortie **pmlogger** à l'aide de divers outils PCP et les comparer à d'autres sources d'informations sur les performances.

Conditions préalables

- PCP est installé. Pour plus d'informations, voir Installation et activation de PCP.

Procédure

1. Mettre à jour la configuration de **pmlogger**:

```
# pmlogconf -r /var/lib/pcp/config/pmlogger/config.default
```

2. Démarrez les services **pmcd** et **pmlogger**:

```
# systemctl start pmcd.service

# systemctl start pmlogger.service
```

3. Exécutez les opérations nécessaires pour enregistrer les données de performance.

4. Arrêtez les services **pmcd** et **pmlogger**:

```
# systemctl stop pmcd.service

# systemctl stop pmlogger.service
```

5. Enregistrez la sortie et sauvegardez-la dans un fichier **tar.gz** dont le nom est basé sur le nom de l'hôte et sur la date et l'heure actuelles :

```
# cd /var/log/pcp/pmlogger/

# tar -czf $(hostname).$(date +%F-%Hh%M).pcp.tar.gz $(hostname)
```

Extraire ce fichier et analyser les données à l'aide des outils PCP.

Ressources supplémentaires

- **pmlogconf(1)**, **pmlogger(1)**, et **pmcd(1)** pages de manuel
- Outils distribués avec le PCP
- Services système distribués avec PCP

5.4. SERVICES SYSTÈME DISTRIBUÉS AVEC PCP

Le tableau suivant décrit les rôles des différents services du système, qui sont distribués avec PCP.

Tableau 5.1. Rôles des services du système distribués avec le PCP

Nom	Description
pmcd	Le Performance Metric Collector Daemon (PMCD).
pmie	Le moteur d'inférence des mesures de performance.
pmlogger	L'enregistreur de mesures de performance.

pmproxy	Le proxy de mesures de performance en temps réel et historique, la requête de séries temporelles et le service API REST.

5.5. OUTILS DISTRIBUÉS AVEC LE PCP

Le tableau suivant décrit l'utilisation des différents outils fournis avec PCP.

Tableau 5.2. Utilisation des outils distribués avec le PCP

Nom	Description
pcp	Affiche l'état actuel de l'installation de Performance Co-Pilot.
pcp-atop	Montre l'occupation au niveau du système des ressources matérielles les plus critiques du point de vue des performances : CPU, mémoire, disque et réseau.
pcp-atopsar	Génère un rapport d'activité au niveau du système sur une variété d'utilisation des ressources du système. Le rapport est généré à partir d'un fichier journal brut préalablement enregistré à l'aide de pmlogger ou de l'option -w de pcp-atop.
pcp-dmcache	Affiche des informations sur les cibles configurées de Device Mapper Cache, telles que : les IOP des périphériques, l'utilisation des périphériques de cache et de métadonnées, ainsi que les taux de réussite et d'échec et les ratios pour les lectures et les écritures pour chaque périphérique de cache.
pcp-dstat	Affiche les métriques d'un seul système à la fois. Pour afficher les mesures de plusieurs systèmes, utilisez l'option **--host**.
pcp-free	Indique la mémoire libre et la mémoire utilisée dans un système.
pcp-htop	Affiche tous les processus en cours d'exécution sur un système ainsi que leurs arguments de ligne de commande d'une manière similaire à la commande **top**, mais vous permet de faire défiler verticalement et horizontalement et d'interagir à l'aide d'une souris. Vous pouvez également visualiser les processus sous forme d'arbre et sélectionner et agir sur plusieurs processus à la fois.

pcp-ipcs	Affiche des informations sur les installations de communication interprocessus (IPC) pour lesquelles le processus appelant dispose d'un accès en lecture.
pcp-numastat	Affiche les statistiques d'allocation NUMA de l'allocateur de mémoire du noyau.
pcp-pidstat	Affiche des informations sur les tâches individuelles ou les processus en cours d'exécution sur le système, telles que le pourcentage de CPU, l'utilisation de la mémoire et de la pile, la planification et la priorité : Le pourcentage de CPU, l'utilisation de la mémoire et de la pile, la planification et la priorité. Affiche par défaut les données en direct de l'hôte local.
pcp-ss	Affiche les statistiques des sockets collectées par l'agent PMDA (Performance Metrics Domain Agent).
pcp-uptime	Affiche la durée de fonctionnement du système, le nombre d'utilisateurs actuellement connectés et les moyennes de charge du système pour les 1, 5 et 15 dernières minutes.
pcp-vmstat	Fournit une vue d'ensemble des performances du système toutes les 5 secondes. Affiche des informations sur les processus, la mémoire, la pagination, les entrées-sorties par bloc, les pièges et l'activité de l'unité centrale.
pmchart	Trace les valeurs des mesures de performance disponibles par le biais des installations du copilote de performance.
pmclient	Affiche les mesures de performance du système de haut niveau en utilisant l'interface de programmation d'applications de mesures de performance (PMAPI).
pmconfig	Affiche les valeurs des paramètres de configuration.
pmdbg	Affiche les drapeaux de contrôle de débogage de Performance Co-Pilot disponibles et leurs valeurs.
pmdiff	Compare les valeurs moyennes de chaque paramètre dans une ou deux archives, dans une fenêtre temporelle donnée, pour détecter les changements susceptibles d'être intéressants lors de la recherche de régressions des performances.
pmdumplog	Affiche les informations relatives au contrôle, aux métadonnées, à l'index et à l'état d'un fichier d'archive de Performance Co-Pilot.

pmdumptext	Produit les valeurs des mesures de performance collectées en direct ou à partir d'une archive Performance Co-Pilot.
pmerr	Affiche les codes d'erreur Performance Co-Pilot disponibles et les messages d'erreur correspondants.
pmfind	Recherche des services de PCP sur le réseau.
pmie	Un moteur d'inférence qui évalue périodiquement un ensemble d'expressions arithmétiques, logiques et de règles. Les mesures sont collectées soit à partir d'un système réel, soit à partir d'un fichier d'archive de Performance Co-Pilot.
pmieconf	Affiche ou définit les variables pmie configurables.
pmiectl	Gère les instances non primaires de pmie.
pminfo	Affiche des informations sur les mesures de performance. Les mesures sont collectées soit à partir d'un système réel, soit à partir d'un fichier d'archive de Performance Co-Pilot.
pmiostat	Affiche les statistiques d'E/S pour les périphériques SCSI (par défaut) ou les périphériques de mappage de périphériques (avec l'option -x dm).
pmlc	Configure de manière interactive les instances actives de pmlogger.
pmlogcheck	Identifie les données non valides dans un fichier d'archive de Performance Co-Pilot.
pmlogconf	Crée et modifie un fichier de configuration pmlogger.
pmlogctl	Gère les instances non primaires de pmlogger.
pmloglabel	Vérifie, modifie ou répare l'étiquette d'un fichier d'archive de Performance Co-Pilot.
pmlogsummary	Calcule des informations statistiques sur les mesures de performance stockées dans un fichier d'archive de Performance Co-Pilot.
pmprobe	Détermine la disponibilité des mesures de performance.

pmrep	Rapports sur des valeurs de mesure de la performance sélectionnées et facilement personnalisables.
pmsocks	Permet d'accéder à un hôte Performance Co-Pilot à travers un pare-feu.
pmstat	Affiche périodiquement un bref résumé des performances du système.
pmstore	Modifie les valeurs des mesures de performance.
pmtrace	Fournit une interface de ligne de commande à la trace PMDA.
pmval	Affiche la valeur actuelle d'une mesure de performance.

5.6. ARCHITECTURES DE DÉPLOIEMENT PCP

Performance Co-Pilot (PCP) prend en charge plusieurs architectures de déploiement, en fonction de l'échelle de déploiement de PCP, et offre de nombreuses options pour réaliser des configurations avancées.

Les variantes de configuration de déploiement de mise à l'échelle disponibles, basées sur la configuration de déploiement recommandée par Red Hat, les facteurs de dimensionnement et les options de configuration, sont les suivantes :

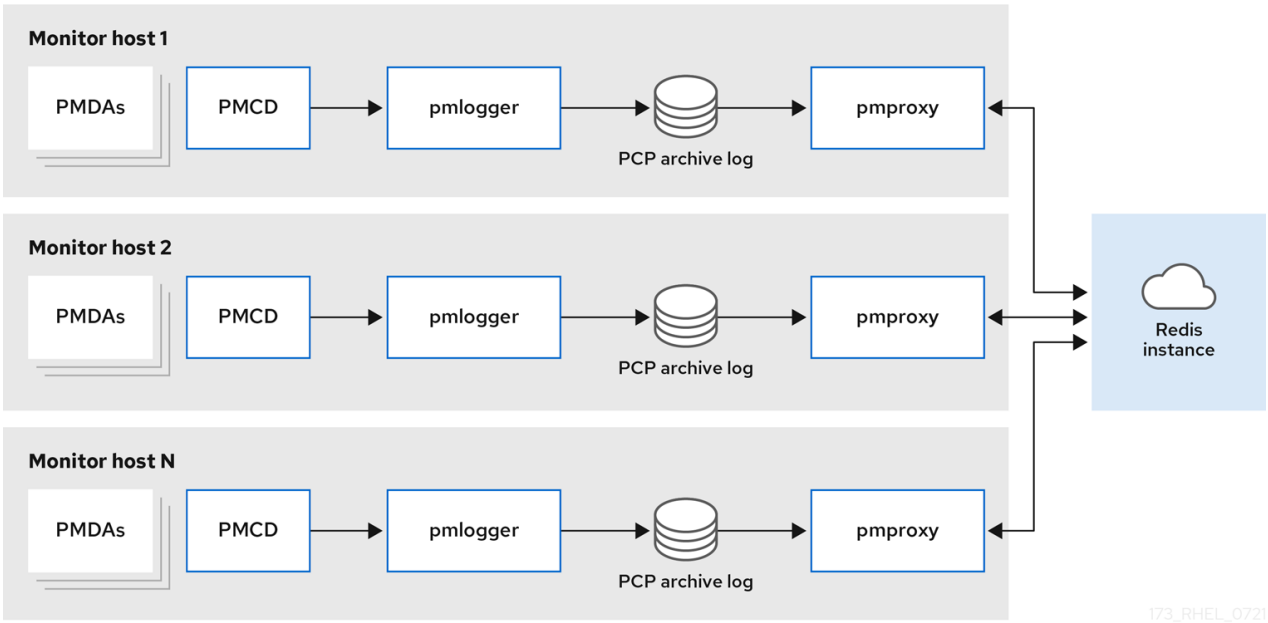
Localhost

Chaque service s'exécute localement sur la machine surveillée. Lorsque vous démarrez un service sans aucun changement de configuration, il s'agit du déploiement par défaut. Dans ce cas, il n'est pas possible de passer à une échelle supérieure à celle d'un nœud individuel.
Par défaut, le déploiement de Redis se fait de manière autonome, sur l'hôte local. Cependant, Redis peut optionnellement fonctionner en grappe hautement disponible et hautement évolutive, où les données sont partagées entre plusieurs hôtes. Une autre option viable consiste à déployer un cluster Redis dans le nuage ou à utiliser un cluster Redis géré par un fournisseur de nuage.

Decentralized

La seule différence entre l'hôte local et la configuration décentralisée est le service Redis centralisé. Dans ce modèle, l'hôte exécute le service **pmlogger** sur chaque hôte surveillé et récupère les mesures à partir d'une instance locale **pmcd**. Un service local **pmproxy** exporte ensuite les mesures de performance vers une instance Redis centrale.

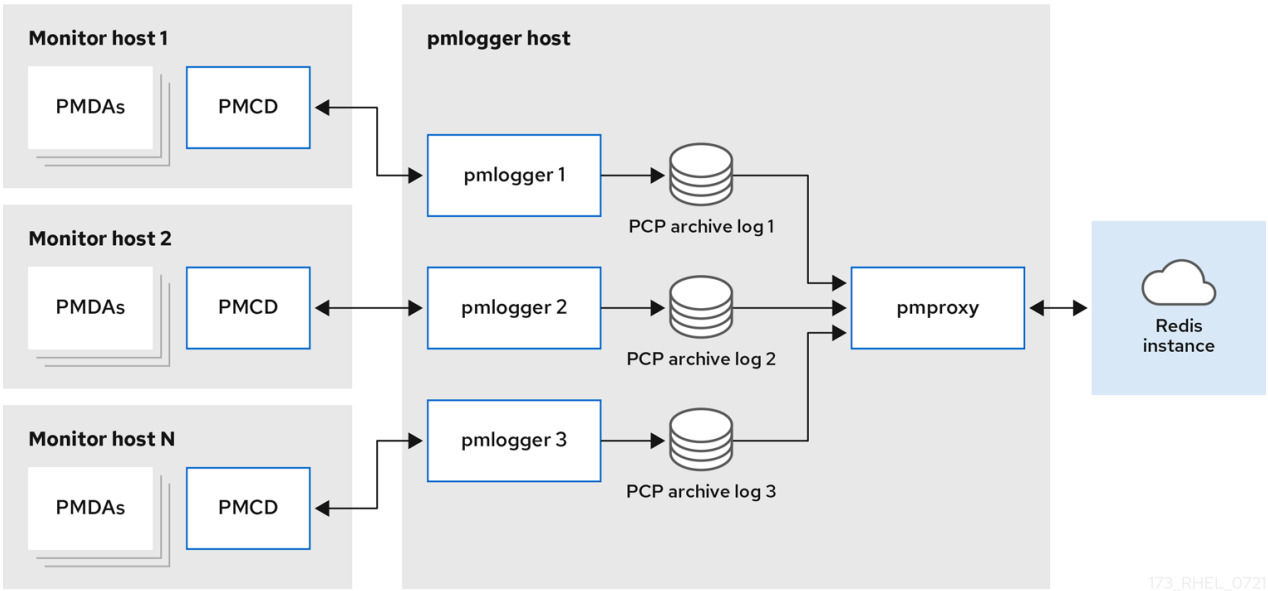
Figure 5.1. Exploitation décentralisée des données



Centralized logging - pmlogger farm

Lorsque l'utilisation des ressources sur les hôtes surveillés est limitée, une autre option de déploiement est une ferme **pmlogger**, également connue sous le nom de journalisation centralisée. Dans cette configuration, un seul hôte enregistreur exécute plusieurs processus **pmlogger**, et chacun est configuré pour récupérer des mesures de performance à partir d'un hôte **pmcd** distant différent. L'hôte de l'enregistreur centralisé est également configuré pour exécuter le service **pmproxy**, qui découvre les archives PCP résultantes et charge les données métriques dans une instance Redis.

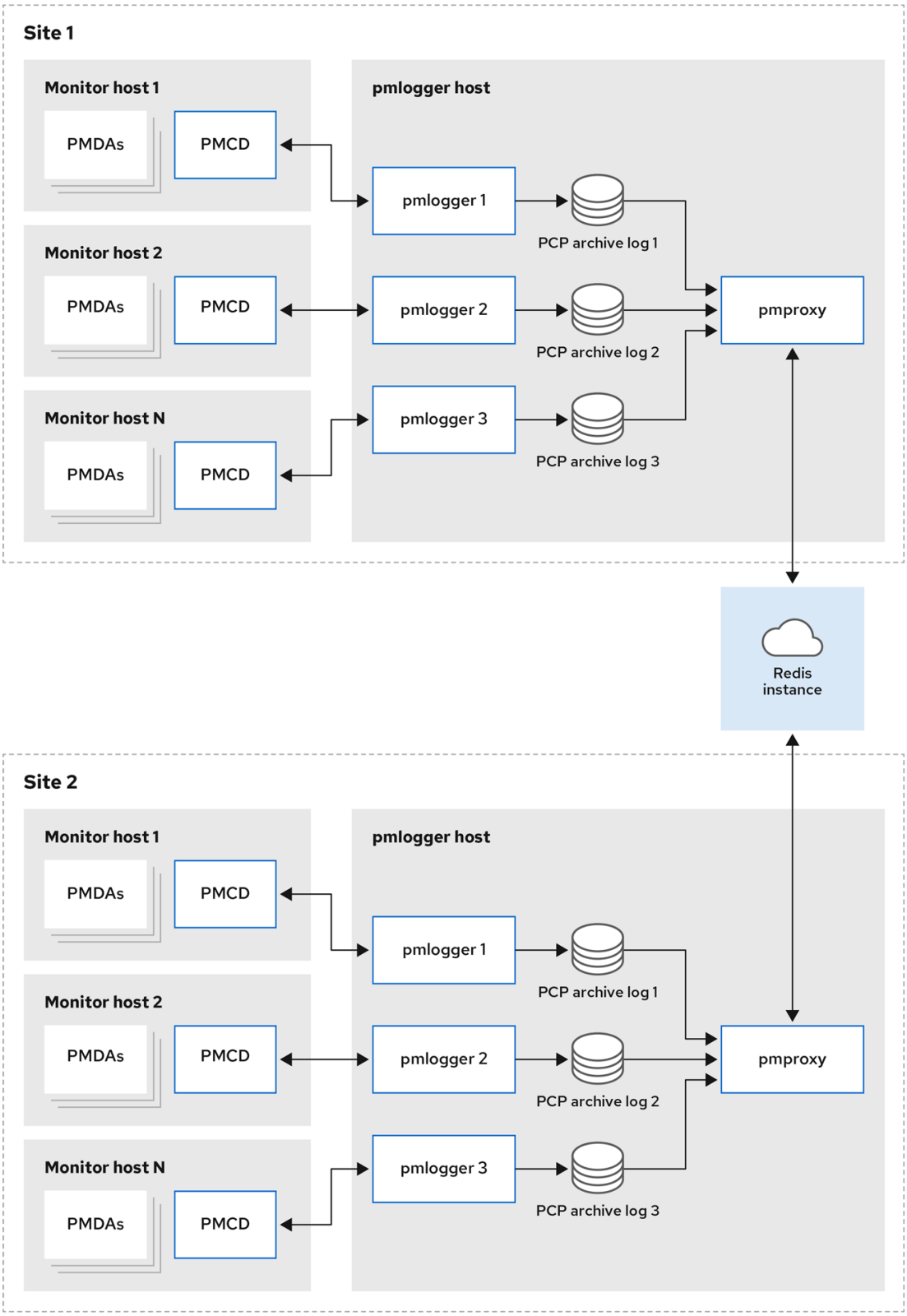
Figure 5.2. Journalisation centralisée - pmlogger farm

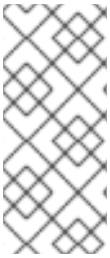


Federated - multiple pmlogger farms

Pour les déploiements à grande échelle, Red Hat recommande de déployer plusieurs fermes **pmlogger** de manière fédérée. Par exemple, une ferme **pmlogger** par rack ou centre de données. Chaque ferme **pmlogger** charge les métriques dans une instance Redis centrale.

Figure 5.3. Federated - de multiples fermes de pmlogger





NOTE

Par défaut, le déploiement de Redis se fait de manière autonome, sur l'hôte local. Cependant, Redis peut optionnellement fonctionner en grappe hautement disponible et hautement évolutive, où les données sont partagées entre plusieurs hôtes. Une autre option viable consiste à déployer un cluster Redis dans le nuage ou à utiliser un cluster Redis géré par un fournisseur de nuage.

Ressources supplémentaires

- **pcp(1)**, **pmlogger(1)**, **pmproxy(1)**, et **pmcd(1)** pages de manuel
- Architecture de déploiement recommandée

5.7. ARCHITECTURE DE DÉPLOIEMENT RECOMMANDÉE

Le tableau suivant décrit les architectures de déploiement recommandées en fonction du nombre d'hôtes surveillés.

Tableau 5.3. Architecture de déploiement recommandée

Nombre d'hôtes (N)	1-10	10-100	100-1000
pmcd serveurs	N	N	N
pmlogger serveurs	1 à N	N/10 à N	N/100 à N
pmproxy serveurs	1 à N	1 à N	N/100 à N
Serveurs Redis	1 à N	1 à N/10	N/100 à N/10
Cluster Redis	Non	Peut-être	Oui
Configuration de déploiement recommandée	Journalisation locale, décentralisée ou centralisée	Décentralisé, enregistrement centralisé ou fédéré	Décentralisé ou fédéré

5.8. FACTEURS DE DIMENSIONNEMENT

Les facteurs de dimensionnement requis pour la mise à l'échelle sont les suivants :

Remote system size

Le nombre d'unités centrales, de disques, d'interfaces réseau et d'autres ressources matérielles influe sur la quantité de données collectées par chaque site **pmlogger** sur l'hôte d'enregistrement centralisé.

Logged Metrics

Le nombre et les types de mesures enregistrées jouent un rôle important. En particulier, les mesures **per-process proc.*** nécessitent un espace disque important. Par exemple, avec la configuration standard **pcp-zeroconf**, un intervalle de journalisation de 10 secondes, 11 Mo sans les mesures proc

contre 155 Mo avec les mesures proc, soit un facteur de 10 fois supérieur. En outre, le nombre d'instances pour chaque mesure, par exemple le nombre de CPU, de périphériques de bloc et d'interfaces réseau, a également une incidence sur la capacité de stockage requise.

Logging Interval

L'intervalle de temps pendant lequel les métriques sont enregistrées affecte les besoins en stockage. Les tailles prévues des fichiers d'archive PCP quotidiens sont écrites dans le fichier **pmlogger.log** pour chaque instance de **pmlogger**. Ces valeurs sont des estimations non compressées. Étant donné que les archives PCP sont très bien compressées (environ 10:1), les besoins réels en espace disque à long terme peuvent être déterminés pour un site particulier.

pmlogrewrite

Après chaque mise à jour de PCP, l'outil **pmlogrewrite** est exécuté et réécrit les anciennes archives s'il y a eu des changements dans les métadonnées métriques entre la version précédente et la nouvelle version de PCP. La durée de ce processus est linéaire en fonction du nombre d'archives stockées.

Ressources supplémentaires

- **pmlogrewrite(1)** et **pmlogger(1)** pages de manuel

5.9. OPTIONS DE CONFIGURATION POUR LA MISE À L'ÉCHELLE DU PCP

Les options de configuration suivantes sont nécessaires pour la mise à l'échelle :

sysctl and rlimit settings

Lorsque la découverte d'archives est activée, **pmproxy** a besoin de quatre descripteurs pour chaque **pmlogger** qu'il surveille ou enregistre, ainsi que des descripteurs de fichiers supplémentaires pour les journaux de service et les sockets des clients **pmproxy**, le cas échéant. Chaque processus **pmlogger** utilise environ 20 descripteurs de fichiers pour la socket **pmcd** distante, les fichiers d'archive, les journaux de service et autres. Au total, cela peut dépasser la limite de 1024 soft par défaut sur un système exécutant environ 200 processus **pmlogger**. Le service **pmproxy** dans **pcp-5.3.0** et les versions ultérieures augmente automatiquement la limite douce à la limite dure. Sur les versions antérieures de PCP, un réglage est nécessaire si un nombre élevé de processus **pmlogger** doit être déployé, ce qui peut être réalisé en augmentant les limites logicielles ou matérielles pour **pmlogger**. Pour plus d'informations, voir Comment définir des limites (ulimit) pour les services exécutés par systemd.

Local Archives

Le service **pmlogger** stocke les mesures des systèmes locaux et distants **pmcds** dans le répertoire **/var/log/pcp/pmlogger/**. Pour contrôler l'intervalle de journalisation du système local, mettez à jour le fichier **/etc/pcp/pmlogger/control.d/*configfile*** et ajoutez **-t *X*** dans les arguments, où *X* est l'intervalle de journalisation en secondes. Pour configurer les mesures qui doivent être enregistrées, exécutez **pmlogconf /var/lib/pcp/config/pmlogger/config.*clienthostname***. Cette commande déploie un fichier de configuration avec un ensemble de mesures par défaut, qui peut éventuellement être personnalisé. Pour spécifier les paramètres de rétention, c'est-à-dire quand purger les anciennes archives PCP, mettez à jour le fichier **/etc/sysconfig/pmlogger_timers** et spécifiez **PMLOGGER_DAILY_PARAMS="-E -k *X*"**où *X* est le nombre de jours pendant lesquels les archives PCP doivent être conservées.

Redis

Le service **pmproxy** envoie les mesures consignées de **pmlogger** à une instance Redis. Voici les deux options disponibles pour spécifier les paramètres de rétention dans le fichier de configuration **/etc/pcp/pmproxy/pmproxy.conf**:

- **stream.expire** spécifie la durée pendant laquelle les métriques périmées doivent être supprimées, c'est-à-dire les métriques qui n'ont pas été mises à jour pendant un laps de temps spécifié en secondes.
- **stream.maxlen** spécifie le nombre maximum de valeurs métriques pour une métrique par hôte. Ce paramètre doit correspondre à la durée de conservation divisée par l'intervalle d'enregistrement, par exemple 20160 pour une conservation de 14 jours et un intervalle d'enregistrement de 60s (60*60*24*14/60)

Ressources supplémentaires

- **pmproxy(1)**, **pmlogger(1)**, et **sysctl(8)** pages de manuel

5.10. EXEMPLE : ANALYSE DU DÉPLOIEMENT DE LA JOURNALISATION CENTRALISÉE

Les résultats suivants ont été recueillis sur une configuration de journalisation centralisée, également connue sous le nom de déploiement de ferme pmlogger, avec une installation par défaut de **pcp-zeroconf 5.3.0**, où chaque hôte distant est une instance de conteneur identique exécutant **pmcd** sur un serveur avec 64 cœurs de CPU, 376 Go de RAM, et un disque attaché.

L'intervalle de journalisation est de 10 secondes, les métriques proc des nœuds distants ne sont pas incluses et les valeurs de mémoire se réfèrent à la valeur RSS (Resident Set Size).

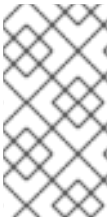
Tableau 5.4. Statistiques d'utilisation détaillées pour un intervalle d'enregistrement de 10 secondes

Nombre d'hôtes	10	50
Stockage des archives PCP par jour	91 MB	522 MB
pmlogger Mémoire	160 MO	580 MB
pmlogger Réseau par jour (en)	2 MB	9 MB
pmproxy Mémoire	1.4 GB	6.3 GB
Mémoire Redis par jour	2.6 GB	12 GB

Tableau 5.5. Ressources utilisées en fonction des hôtes surveillés pour l'intervalle d'enregistrement de 60s

Nombre d'hôtes	10	50	100
Stockage des archives PCP par jour	20 MB	120 MB	271 MB
pmlogger Mémoire	104 MB	524 MB	1049 MB

Nombre d'hôtes	10	50	100
pmlogger Réseau par jour (en)	0.38 MB	1.75 MB	3.48 MB
pmproxy Mémoire	2.67 GB	5.5GB	9 GB
Mémoire Redis par jour	0.54 GB	2.65 GB	5.3 GB



NOTE

Le site **pmproxy** met en file d'attente les requêtes Redis et utilise le pipelining Redis pour accélérer les requêtes Redis. Cela peut entraîner une utilisation élevée de la mémoire. Pour résoudre ce problème, voir Résolution des problèmes liés à une utilisation élevée de la mémoire.

5.11. EXEMPLE : ANALYSE DU DÉPLOIEMENT DE L'INSTALLATION FÉDÉRÉE

Les résultats suivants ont été observés sur une installation fédérée, également connue sous le nom de fermes **pmlogger** multiples, composée de trois installations de journalisation centralisée (fermes**pmlogger**), où chaque ferme **pmlogger** surveillait 100 hôtes distants, soit 300 hôtes au total.

Cette configuration des fermes **pmlogger** est identique à celle mentionnée dans le document

Exemple : Analyse du déploiement de la journalisation centralisée pour l'intervalle de journalisation de 60s, sauf que les serveurs Redis fonctionnaient en mode cluster.

Tableau 5.6. Ressources utilisées en fonction des hôtes fédérés pour un intervalle de journalisation de 60s

Stockage des archives PCP par jour	**pmlogger** Mémoire	Réseau par jour (entrée/sortie)	**pmproxy** Mémoire	Mémoire Redis par jour
277 MB	1058 MB	15.6 MB / 12.3 MB	6-8 GB	5.5 GB

Ici, toutes les valeurs sont par hôte. La bande passante du réseau est plus élevée en raison de la communication entre les nœuds du cluster Redis.

5.12. ÉTABLIR DES CONNEXIONS PCP SÉCURISÉES

Vous pouvez configurer les composants de collecte et de surveillance PCP pour qu'ils participent aux échanges sécurisés du protocole PCP.

5.12.1. Connexions PCP sécurisées

Vous pouvez établir des connexions sécurisées entre les composants de collecte et de surveillance de Performance Co-Pilot (PCP). Les composants de collecte PCP sont les parties de PCP qui collectent et extraient les données de performance à partir de différentes sources. Les composants de surveillance

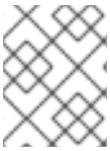
PCP sont les parties de PCP qui affichent les données collectées à partir d'hôtes ou d'archives sur lesquels sont installés les composants de collecte PCP. L'établissement de connexions sécurisées entre ces composants permet d'éviter que des personnes non autorisées n'accèdent aux données collectées et surveillées ou ne les modifient.

Toutes les connexions avec le démon Performance Metrics Collector (**pmcd**) sont effectuées à l'aide du protocole PCP basé sur TCP/IP. Le protocole proxy et les API REST de PCP sont servis par le démon **pmproxy** - l'API REST est accessible via HTTPS, ce qui garantit une connexion sécurisée.

Les démons **pmcd** et **pmproxy** sont tous deux capables d'établir des communications TLS et non TLS simultanées sur un seul port. Le port par défaut pour **pmcd** est 44321 et 44322 pour **pmproxy**. Cela signifie que vous n'avez pas à choisir entre les communications TLS ou non TLS pour vos systèmes collecteurs PCP et que vous pouvez utiliser les deux en même temps.

5.12.2. Configuration de connexions sécurisées pour les composants du collecteur PCP

Tous les systèmes collecteurs PCP doivent disposer de certificats valides afin de participer aux échanges sécurisés du protocole PCP.



NOTE

le démon **pmproxy** fonctionne à la fois comme un client et un serveur du point de vue de TLS.

Conditions préalables

- PCP est installé. Pour plus d'informations, voir Installation et activation de PCP.
- La clé privée du client est stockée dans le fichier **/etc/pcp/tls/client.key**. Si vous utilisez un chemin différent, adaptez les étapes correspondantes de la procédure.
  Pour plus de détails sur la création d'une clé privée et d'une demande de signature de certificat (CSR), ainsi que sur la manière de demander un certificat à une autorité de certification (AC), consultez la documentation de votre AC.
- Le certificat client TLS est stocké dans le fichier **/etc/pcp/tls/client.crt**. Si vous utilisez un chemin différent, adaptez les étapes correspondantes de la procédure.
- Le certificat CA est stocké dans le fichier **/etc/pcp/tls/ca.crt**. Si vous utilisez un chemin différent, adaptez les étapes correspondantes de la procédure. En outre, pour le démon **pmproxy**:
- La clé privée du serveur est stockée dans le fichier **/etc/pcp/tls/server.key**. Si vous utilisez un chemin différent, adaptez les étapes correspondantes de la procédure
- Le certificat du serveur TLS est stocké dans le fichier **/etc/pcp/tls/server.crt**. Si vous utilisez un chemin différent, adaptez les étapes correspondantes de la procédure.

Procédure

1. Mettez à jour le fichier de configuration PCP TLS sur les systèmes collecteurs afin d'utiliser les certificats émis par l'autorité de certification pour établir une connexion sécurisée :

```
# cat > /etc/pcp/tls.conf << END
tls-ca-cert-file = /etc/pcp/tls/ca.crt
```

```
tls-key-file = /etc/pcp/tls/server.key
tls-cert-file = /etc/pcp/tls/server.crt
tls-client-key-file = /etc/pcp/tls/client.key
tls-client-cert-file = /etc/pcp/tls/client.crt
END
```

2. Redémarrer l'infrastructure du collecteur PCP :

```
# systemctl restart pmcd.service
# systemctl restart pmproxy.service
```

Vérification

- Vérifiez la configuration TLS :
  - Sur le service **pmcd**:

```
# grep 'Info:' /var/log/pcp/pmcd/pmcd.log
[Tue Feb 07 11:47:33] pmcd(6558) Info: OpenSSL 3.0.7 setup
```

  - Sur le service **pmproxy**:

```
# grep 'Info:' /var/log/pcp/pmproxy/pmproxy.log
[Tue Feb 07 11:44:13] pmproxy(6014) Info: OpenSSL 3.0.7 setup
```

5.12.3. Configuration de connexions sécurisées pour les composants de surveillance PCP

Configurez vos composants de surveillance PCP pour qu'ils participent aux échanges sécurisés du protocole PCP.

Conditions préalables

- PCP est installé. Pour plus d'informations, voir Installation et activation de PCP.
- La clé privée du client est stockée dans le fichier **~/.pcp/tls/client.key**. Si vous utilisez un chemin différent, adaptez les étapes correspondantes de la procédure.
  Pour plus de détails sur la création d'une clé privée et d'une demande de signature de certificat (CSR), ainsi que sur la manière de demander un certificat à une autorité de certification (AC), consultez la documentation de votre AC.
- Le certificat client TLS est stocké dans le fichier **~/.pcp/tls/client.crt**. Si vous utilisez un chemin différent, adaptez les étapes correspondantes de la procédure.
- Le certificat CA est stocké dans le fichier **/etc/pcp/tls/ca.crt**. Si vous utilisez un chemin différent, adaptez les étapes correspondantes de la procédure.

Procédure

1. Créez un fichier de configuration TLS avec les informations suivantes :

```
$ home=echo ~
$ cat > ~/.pcp/tls.conf << END
tls-ca-cert-file = /etc/pcp/tls/ca.crt
```

```
tls-key-file = $home/.pcp/tls/client.key
tls-cert-file = $home/.pcp/tls/client.crt
END
```

2. Établir la connexion sécurisée :

```
$ export PCP_SECURE_SOCKETS=enforce
$ export PCP_TLSCONF_PATH=~/.pcp/tls.conf
```

Vérification

- Vérifiez que la connexion sécurisée est configurée :

```
$ pminfo --fetch --host pcps://localhost kernel.all.load

kernel.all.load
    inst [1 or "1 minute"] value 1.26
    inst [5 or "5 minute"] value 1.29
    inst [15 or "15 minute"] value 1.28
```

5.13. DÉPANNAGE EN CAS D'UTILISATION ÉLEVÉE DE LA MÉMOIRE

Les scénarios suivants peuvent entraîner une utilisation élevée de la mémoire :

- Le processus **pmproxy** est occupé à traiter de nouvelles archives PCP et n'a pas de cycles de CPU disponibles pour traiter les requêtes et les réponses Redis.
- Le nœud ou le cluster Redis est surchargé et ne peut pas traiter les demandes entrantes à temps.

Le démon de service **pmproxy** utilise les flux Redis et prend en charge les paramètres de configuration, qui sont des paramètres de réglage PCP et affectent l'utilisation de la mémoire Redis et la conservation des clés. Le fichier **/etc/pcp/pmproxy/pmproxy.conf** répertorie les options de configuration disponibles pour **pmproxy** et les API associées.

La procédure suivante décrit comment résoudre un problème d'utilisation élevée de la mémoire.

Conditions préalables

1. Installez le paquetage **pcp-pmda-redis**:

```
# dnf install pcp-pmda-redis
```

2. Installez le PMDA redis :

```
# cd /var/lib/pcp/pmdas/redis && ./Install
```

Procédure

- Pour résoudre un problème d'utilisation élevée de la mémoire, exécutez la commande suivante et observez la colonne **inflight**:

```
$ pmrep :pmproxy
```

```
        backlog  inflight  reqs/s  resp/s   wait req err  resp err  changed  throttled
           byte     count   count/s count/s  s/s  count/s   count/s  count/s   count/s
14:59:08   0        0       N/A      N/A     N/A  N/A       N/A      N/A       N/A
14:59:09   0        0       2268.9   2268.9  28   0         0        2.0       4.0
14:59:10   0        0       0.0      0.0     0    0         0        0.0       0.0
14:59:11   0        0       0.0      0.0     0    0         0        0.0       0.0
```

Cette colonne indique le nombre de requêtes Redis en cours, ce qui signifie qu'elles sont en file d'attente ou envoyées et qu'aucune réponse n'a été reçue jusqu'à présent.

Un nombre élevé indique l'une des conditions suivantes :

- Le processus **pmproxy** est occupé à traiter de nouvelles archives PCP et n'a pas de cycles de CPU disponibles pour traiter les requêtes et les réponses Redis.
- Le nœud ou le cluster Redis est surchargé et ne peut pas traiter les demandes entrantes à temps.

- Pour résoudre le problème de l'utilisation élevée de la mémoire, réduisez le nombre de processus **pmlogger** pour cette ferme et ajoutez une autre ferme pmlogger. Utilisez la configuration fédérée - plusieurs fermes pmlogger.
  Si le nœud Redis utilise 100 PU pendant une période prolongée, déplacez-le vers un hôte plus performant ou utilisez plutôt une configuration Redis en grappe.

- Pour afficher les paramètres de **pmproxy.redis.***, utilisez la commande suivante :

```
$ pminfo -ftd pmproxy.redis
pmproxy.redis.responses.wait [wait time for responses]
    Data Type: 64-bit unsigned int  InDom: PM_INDOM_NULL 0xffffffff
    Semantics: counter  Units: microsec
    value 546028367374
pmproxy.redis.responses.error [number of error responses]
    Data Type: 64-bit unsigned int  InDom: PM_INDOM_NULL 0xffffffff
    Semantics: counter  Units: count
    value 1164
[...]
pmproxy.redis.requests.inflight.bytes [bytes allocated for inflight requests]
    Data Type: 64-bit int  InDom: PM_INDOM_NULL 0xffffffff
    Semantics: discrete  Units: byte
    value 0

pmproxy.redis.requests.inflight.total [inflight requests]
    Data Type: 64-bit unsigned int  InDom: PM_INDOM_NULL 0xffffffff
    Semantics: discrete  Units: count
    value 0
[...]
```

Pour connaître le nombre de requêtes Redis en cours, consultez les métriques **pmproxy.redis.requests.inflight.total** et **pmproxy.redis.requests.inflight.bytes** pour savoir combien d'octets sont occupés par toutes les requêtes Redis en cours.

En général, la file d'attente des requêtes redis est nulle, mais elle peut s'allonger en fonction de l'utilisation des grandes fermes pmlogger, ce qui limite l'évolutivité et peut entraîner des temps de latence élevés pour les clients **pmproxy**.

- Utilisez la commande **pminfo** pour afficher des informations sur les mesures de performance. Par exemple, pour afficher les mesures de **redis.***, utilisez la commande suivante :

```
$ pminfo -ftd redis
redis.redis_build_id [Build ID]
    Data Type: string  InDom: 24.0 0x6000000
    Semantics: discrete  Units: count
    inst [0 or "localhost:6379"] value "87e335e57cffa755"
redis.total_commands_processed [Total number of commands processed by the server]
    Data Type: 64-bit unsigned int  InDom: 24.0 0x6000000
    Semantics: counter  Units: count
    inst [0 or "localhost:6379"] value 595627069
[...]

redis.used_memory_peak [Peak memory consumed by Redis (in bytes)]
    Data Type: 32-bit unsigned int  InDom: 24.0 0x6000000
    Semantics: instant  Units: count
    inst [0 or "localhost:6379"] value 572234920
[...]
```

  Pour connaître l'utilisation maximale de la mémoire, consultez la métrique **redis.used_memory_peak**.

Ressources supplémentaires

- **pmdaredis(1)**, **pmproxy(1)**, et **pminfo(1)** pages de manuel
- Architectures de déploiement PCP

CHAPITRE 6. ENREGISTREMENT DES DONNÉES DE PERFORMANCE AVEC PMLOGGER

L'outil PCP vous permet d'enregistrer les valeurs des mesures de performance et de les rejouer ultérieurement. Cela vous permet d'effectuer une analyse rétrospective des performances.

En utilisant l'outil **pmlogger**, vous pouvez

- Créer les journaux archivés des mesures sélectionnées sur le système
- Spécifier quelles mesures sont enregistrées sur le système et à quelle fréquence

6.1. MODIFIER LE FICHIER DE CONFIGURATION DE PMLOGGER AVEC PMLOGCONF

Lorsque le service **pmlogger** est en cours d'exécution, PCP enregistre un ensemble de mesures par défaut sur l'hôte.

Utilisez l'utilitaire **pmlogconf** pour vérifier la configuration par défaut. Si le fichier de configuration **pmlogger** n'existe pas, **pmlogconf** le crée avec des valeurs métriques par défaut.

Conditions préalables

- PCP est installé. Pour plus d'informations, voir Installation et activation de PCP.

Procédure

1. Créer ou modifier le fichier de configuration de **pmlogger**:

```
# pmlogconf -r /var/lib/pcp/config/pmlogger/config.default
```

2. Suivez les invites de **pmlogconf** pour activer ou désactiver des groupes de mesures de performance connexes et pour contrôler l'intervalle de journalisation pour chaque groupe activé.

Ressources supplémentaires

- **pmlogconf(1)** et **pmlogger(1)** pages de manuel
- Outils distribués avec le PCP
- Services système distribués avec PCP

6.2. MODIFIER MANUELLEMENT LE FICHIER DE CONFIGURATION DE PMLOGGER

Pour créer une configuration de journalisation personnalisée avec des mesures spécifiques et des intervalles donnés, modifiez manuellement le fichier de configuration **pmlogger**. Le fichier de configuration par défaut de **pmlogger** est **/var/lib/pcp/config/pmlogger/config.default**. Le fichier de configuration spécifie les mesures qui sont enregistrées par l'instance de journalisation principale.

En configuration manuelle, vous pouvez

- Enregistrer les mesures qui ne sont pas répertoriées dans la configuration automatique.
- Choisissez des fréquences d'enregistrement personnalisées.
- Ajouter **PMDA** avec les métriques de l'application.

Conditions préalables

- PCP est installé. Pour plus d'informations, voir Installation et activation de PCP.

Procédure

- Ouvrez et modifiez le fichier **/var/lib/pcp/config/pmlogger/config.default** pour ajouter des mesures spécifiques :

```
# It is safe to make additions from here on ...
#

log mandatory on every 5 seconds {
    xfs.write
    xfs.write_bytes
    xfs.read
    xfs.read_bytes
}

log mandatory on every 10 seconds {
    xfs.allocs
    xfs.block_map
    xfs.transactions
    xfs.log

}

[access]
disallow * : all;
allow localhost : enquire;
```

Ressources supplémentaires

- **pmlogger(1)** page de manuel
- Outils distribués avec le PCP
- Services système distribués avec PCP

6.3. ACTIVATION DU SERVICE PMLOGGER

Le service **pmlogger** doit être démarré et activé pour enregistrer les valeurs métriques sur la machine locale.

Cette procédure décrit comment activer le service **pmlogger**.

Conditions préalables

- PCP est installé. Pour plus d'informations, voir Installation et activation de PCP.

Procédure

- Démarrez et activez le service **pmlogger**:

```
# systemctl start pmlogger

# systemctl enable pmlogger
```

Verification steps

- Vérifiez si le service **pmlogger** est activé :

```
# pcp

Performance Co-Pilot configuration on workstation:

platform: Linux workstation 4.18.0-80.el8.x86_64 #1 SMP Wed Mar 13 12:02:46 UTC 2019
x86_64
hardware: 12 cpus, 2 disks, 1 node, 36023MB RAM
timezone: CEST-2
services: pmcd
pmcd: Version 4.3.0-1, 8 agents, 1 client
pmda: root pmcd proc xfs linux mmv kvm jbd2
pmlogger: primary logger: /var/log/pcp/pmlogger/workstation/20190827.15.54
```

Ressources supplémentaires

- **pmlogger(1)** page de manuel
- Outils distribués avec le PCP
- Services système distribués avec PCP
- **/var/lib/pcp/config/pmlogger/config.default** fichier

6.4. MISE EN PLACE D'UN SYSTÈME CLIENT POUR LA COLLECTE DE DONNÉES

Cette procédure décrit comment configurer un système client de manière à ce qu'un serveur central puisse collecter des métriques auprès des clients utilisant PCP.

Conditions préalables

- PCP est installé. Pour plus d'informations, voir Installation et activation de PCP.

Procédure

1. Installez le paquetage **pcp-system-tools**:

```
# dnf install pcp-system-tools
```

2. Configurez une adresse IP pour **pmcd**:

```
# echo "-i 192.168.4.62" >>/etc/pcp/pmcd/pmcd.options
```

Remplacez *192.168.4.62* par l'adresse IP sur laquelle le client doit écouter.

Par défaut, **pmcd** écoute sur localhost.

3. Configurer le pare-feu pour ajouter le site public **zone** de façon permanente :

```
# firewall-cmd --permanent --zone=public --add-port=44321/tcp
success

# firewall-cmd --reload
success
```

4. Définit un booléen SELinux :

```
# setsebool -P pcp_bind_all_unreserved_ports on
```

5. Activez les services **pmcd** et **pmlogger**:

```
# systemctl enable pmcd pmlogger
# systemctl restart pmcd pmlogger
```

Verification steps

- Vérifiez que le site **pmcd** écoute correctement l'adresse IP configurée :

```
# ss -tlp | grep 44321
LISTEN   0   5     127.0.0.1:44321    0.0.0.0:*    users:(("pmcd",pid=151595,fd=6))
LISTEN   0   5  192.168.4.62:44321    0.0.0.0:*    users:(("pmcd",pid=151595,fd=0))
LISTEN   0   5         [::1]:44321       [::]:*    users:(("pmcd",pid=151595,fd=7))
```

Ressources supplémentaires

- **pmlogger(1)**, **firewall-cmd(1)**, **ss(8)**, et **setsebool(8)** pages de manuel
- Outils distribués avec le PCP
- Services système distribués avec PCP
- **/var/lib/pcp/config/pmlogger/config.default** fichier

6.5. MISE EN PLACE D'UN SERVEUR CENTRAL POUR LA COLLECTE DES DONNÉES

Cette procédure décrit comment créer un serveur central pour collecter les métriques des clients exécutant PCP.

Conditions préalables

- PCP est installé. Pour plus d'informations, voir Installation et activation de PCP.

- Le client est configuré pour la collecte de métriques. Pour plus d'informations, voir Configuration d'un système client pour la collecte de métriques .

Procédure

1. Installez le paquetage **pcp-system-tools**:

```
# dnf install pcp-system-tools
```

2. Créez le fichier **/etc/pcp/pmlogger/control.d/remote** avec le contenu suivant :

```
# DO NOT REMOVE OR EDIT THE FOLLOWING LINE
$version=1.1

192.168.4.13 n n PCP_ARCHIVE_DIR/rhel7u4a -r -T24h10m -c config.rhel7u4a
192.168.4.14 n n PCP_ARCHIVE_DIR/rhel6u10a -r -T24h10m -c config.rhel6u10a
192.168.4.62 n n PCP_ARCHIVE_DIR/rhel8u1a -r -T24h10m -c config.rhel8u1a
192.168.4.69 n n PCP_ARCHIVE_DIR/rhel9u3a -r -T24h10m -c config.rhel9u3a
```

   Remplacez *192.168.4.13*, *192.168.4.14*, *192.168.4.62* et *192.168.4.69* par les adresses IP des clients.

3. Activez les services **pmcd** et **pmlogger**:

```
# systemctl enable pmcd pmlogger
# systemctl restart pmcd pmlogger
```

Verification steps

- Assurez-vous que vous pouvez accéder au dernier fichier d'archive de chaque répertoire :

```
# for i in /var/log/pcp/pmlogger/rhel*/*.0; do pmdumplog -L $i; done
Log Label (Log Format Version 2)
Performance metrics from host rhel6u10a.local
  commencing Mon Nov 25 21:55:04.851 2019
  ending     Mon Nov 25 22:06:04.874 2019
Archive timezone: JST-9
PID for pmlogger: 24002
Log Label (Log Format Version 2)
Performance metrics from host rhel7u4a
  commencing Tue Nov 26 06:49:24.954 2019
  ending     Tue Nov 26 07:06:24.979 2019
Archive timezone: CET-1
PID for pmlogger: 10941
[..]
```

   Les fichiers d'archive du répertoire **/var/log/pcp/pmlogger/** peuvent être utilisés pour d'autres analyses et graphiques.

Ressources supplémentaires

- **pmlogger(1)** page de manuel
- Outils distribués avec le PCP

- Services système distribués avec PCP
- **/var/lib/pcp/config/pmlogger/config.default** fichier

6.6. REPRODUIRE LES ARCHIVES DES JOURNAUX PCP AVEC PMREP

Après avoir enregistré les données métriques, vous pouvez relire les archives des journaux PCP. Pour exporter les journaux vers des fichiers texte et les importer dans des feuilles de calcul, utilisez les utilitaires PCP tels que **pcp2csv**, **pcp2xml**, **pmrep** ou **pmlogsummary**.

En utilisant l'outil **pmrep**, vous pouvez

- Consulter les fichiers journaux
- Analyse l'archive PCP sélectionnée et exporte les valeurs dans un tableau ASCII
- Extraire l'intégralité du journal d'archive ou seulement certaines valeurs métriques du journal en spécifiant des métriques individuelles sur la ligne de commande

Conditions préalables

- PCP est installé. Pour plus d'informations, voir Installation et activation de PCP.
- Le service **pmlogger** est activé. Pour plus d'informations, voir Activation du service pmlogger.
- Installez le paquetage **pcp-system-tools**:

```
# dnf install pcp-gui
```

Procédure

- Afficher les données sur la métrique :

```
$ pmrep --start @3:00am --archive 20211128 --interval 5seconds --samples 10 --output csv
disk.dev.write
Time,"disk.dev.write-sda","disk.dev.write-sdb"
2021-11-28 03:00:00,,
2021-11-28 03:00:05,4.000,5.200
2021-11-28 03:00:10,1.600,7.600
2021-11-28 03:00:15,0.800,7.100
2021-11-28 03:00:20,16.600,8.400
2021-11-28 03:00:25,21.400,7.200
2021-11-28 03:00:30,21.200,6.800
2021-11-28 03:00:35,21.000,27.600
2021-11-28 03:00:40,12.400,33.800
2021-11-28 03:00:45,9.800,20.600
```

L'exemple mentionné affiche les données sur la métrique **disk.dev.write** collectées dans une archive à un intervalle *5 second* dans un format de valeurs séparées par des virgules.

NOTE

Remplacez **20211128** dans cet exemple par un nom de fichier contenant l'archive **pmlogger** dont vous voulez afficher les données.

Ressources supplémentaires

- **pmlogger(1)**, **pmrep(1)**, et **pmlogsummary(1)** pages de manuel
- Outils distribués avec le PCP
- Services système distribués avec PCP

6.7. ACTIVATION DES ARCHIVES PCP VERSION 3

Les archives Performance Co-Pilot (PCP) stockent les valeurs historiques des métriques PCP enregistrées à partir d'un seul hôte et permettent une analyse rétrospective des performances. Les archives PCP contiennent toutes les données métriques importantes et les métadonnées nécessaires à l'analyse hors ligne ou hors site. Ces archives peuvent être lues par la plupart des outils clients PCP ou vidées de manière brute par l'outil **pmdumplog**.

À partir de PCP 6.0, les archives de la version 3 sont prises en charge en plus des archives de la version 2. Les archives de la version 2 restent les archives par défaut et continueront à bénéficier d'un support à long terme à des fins de rétrocompatibilité, tandis que les archives de la version 3 bénéficieront d'un support à long terme à partir de RHEL 9.2.

L'utilisation des archives de la version 3 du PCP offre les avantages suivants par rapport à la version 2 :

- Prise en charge du changement de domaine d'instance-deltas
- Horodatage sécurisé Y2038
- Horodatage à la nanoseconde près
- Prise en charge de fuseaux horaires arbitraires
- décalages de fichiers de 64 bits utilisés pour les volumes individuels de plus de 2 Go

Conditions préalables

- PCP est installé. Pour plus d'informations, voir Installation et activation de PCP.

Procédure

1. Ouvrez le fichier **/etc/pcp.conf** dans un éditeur de texte de votre choix et définissez la version de l'archive PCP :

```
PCP_ARCHIVE_VERSION=3
```

2. Redémarrez le service **pmlogger** pour appliquer vos changements de configuration :

```
# systemctl restart pmlogger.service
```

3. Créez un nouveau journal d'archive PCP en utilisant votre nouvelle configuration. Pour plus d'informations, voir Enregistrer les données de performance avec pmlogger .

Vérification

- Vérifiez la version de l'archive créée avec votre nouvelle configuration :

```
# pmloglabel -l /var/log/pcp/pmlogger/20230208
Log Label (Log Format Version 3)
Performance metrics from host host1
        commencing Wed Feb   08 00:11:09.396 2023
        ending          Thu  Feb   07 00:13:54.347 2023
```

Ressources supplémentaires

- **logarchive(5)** page de manuel
- **pmlogger(1)** page de manuel
- Enregistrement des données de performance avec pmlogger

CHAPITRE 7. SUIVI DES PERFORMANCES AVEC PERFORMANCE CO-PILOT

Performance Co-Pilot (PCP) est une suite d'outils, de services et de bibliothèques permettant de surveiller, de visualiser, de stocker et d'analyser les mesures de performance au niveau du système.

En tant qu'administrateur système, vous pouvez surveiller les performances du système à l'aide de l'application PCP dans Red Hat Enterprise Linux 9.

7.1. SURVEILLANCE DE POSTFIX AVEC PMDA-POSTFIX

Cette procédure décrit comment surveiller les paramètres de performance du serveur de messagerie **postfix** à l'aide de **pmda-postfix**. Elle permet de vérifier le nombre de courriers électroniques reçus par seconde.

Conditions préalables

- PCP est installé. Pour plus d'informations, voir Installation et activation de PCP.
- Le service **pmlogger** est activé. Pour plus d'informations, voir Activation du service pmlogger.

Procédure

1. Install the following packages:
   a. Installer le site **pcp-system-tools**:

```
# dnf install pcp-system-tools
```

   b. Installez le paquet **pmda-postfix** pour contrôler **postfix**:

```
# dnf install pcp-pmda-postfix postfix
```

   c. Installer le démon de journalisation :

```
# dnf install rsyslog
```

   d. Installez le client de messagerie pour le tester :

```
# dnf install mutt
```

2. Activez les services **postfix** et **rsyslog**:

```
# systemctl enable postfix rsyslog
# systemctl restart postfix rsyslog
```

3. Activez le booléen SELinux, afin que **pmda-postfix** puisse accéder aux fichiers journaux requis :

```
# setsebool -P pcp_read_generic_logs=on
```

4. Installer le site **PMDA**:

```
# cd /var/lib/pcp/pmdas/postfix/

# ./Install

Updating the Performance Metrics Name Space (PMNS) ...
Terminate PMDA if already installed ...
Updating the PMCD control file, and notifying PMCD ...
Waiting for pmcd to terminate ...
Starting pmcd ...
Check postfix metrics have appeared ... 7 metrics and 58 values
```

Verification steps

- Vérifier le fonctionnement de **pmda-postfix**:

```
echo testmail | mutt root
```

- Vérifier les métriques disponibles :

```
# pminfo postfix

postfix.received
postfix.sent
postfix.queues.incoming
postfix.queues.maildrop
postfix.queues.hold
postfix.queues.deferred
postfix.queues.active
```

Ressources supplémentaires

- **rsyslogd(8)**, **postfix(1)**, et **setsebool(8)** pages de manuel
- Outils distribués avec le PCP
- Services système distribués avec PCP
- **/var/lib/pcp/config/pmlogger/config.default** fichier

7.2. TRACER VISUELLEMENT LES ARCHIVES DES JOURNAUX PCP AVEC L'APPLICATION PCP CHARTS

Après avoir enregistré les données métriques, vous pouvez rejouer les archives des journaux PCP sous forme de graphiques. Les mesures proviennent d'un ou de plusieurs hôtes en direct, mais il est également possible d'utiliser les données des archives de journaux PCP comme source de données historiques. Pour personnaliser l'interface de l'application **PCP Charts** afin d'afficher les données des mesures de performance, vous pouvez utiliser des graphiques linéaires, des graphiques à barres ou des graphiques d'utilisation.

En utilisant l'application **PCP Charts**, vous pouvez

- Reproduire les données dans l'application **PCP Charts** et utiliser des graphiques pour visualiser les données rétrospectives avec les données en temps réel du système.

- Tracer les valeurs des mesures de performance dans des graphiques.
- Affichage simultané de plusieurs graphiques.

Conditions préalables

- PCP est installé. Pour plus d'informations, voir Installation et activation de PCP.
- Enregistrement des données de performance à l'aide de **pmlogger**. Pour plus d'informations, voir Enregistrement des données de performance à l'aide de pmlogger .
- Installez le paquetage **pcp-gui**:

```
# dnf install pcp-gui
```

Procédure

1. Lancez l'application **PCP Charts** à partir de la ligne de commande :

```
# pmchart
```

Figure 7.1. Application PCP Charts

Les paramètres du serveur **pmtime** sont situés en bas. Les boutons **start** et **pause** vous permettent de contrôler :

- Intervalle dans lequel le PCP interroge les données métriques
- La date et l'heure des mesures des données historiques

2. Cliquez sur **File** puis sur **New Chart** pour sélectionner une métrique à partir de la machine locale et des machines distantes en spécifiant leur nom d'hôte ou leur adresse. Les options de configuration avancées comprennent la possibilité de définir manuellement les valeurs des axes du graphique et de choisir manuellement la couleur des tracés.
3. Enregistrer les vues créées dans l'application **PCP Charts**:

Les options suivantes permettent de prendre des photos ou d'enregistrer les vues créées dans l'application **PCP Charts**:

- Cliquez sur **File** puis sur **Export** pour enregistrer une image de la vue actuelle.
- Cliquez sur **Record** puis sur **Start** pour démarrer un enregistrement. Cliquez sur **Record** puis sur **Stop** pour arrêter l'enregistrement. Après l'arrêt de l'enregistrement, les mesures enregistrées sont archivées pour être consultées ultérieurement.

4. Facultatif : dans l'application **PCP Charts**, le fichier de configuration principal, appelé **view**, permet de sauvegarder les métadonnées associées à un ou plusieurs graphiques. Ces métadonnées décrivent tous les aspects du graphique, y compris les mesures utilisées et les colonnes du graphique. Enregistrez la configuration personnalisée de **view** en cliquant sur **File** puis sur **Save View**, et chargez la configuration de **view** ultérieurement.
L'exemple suivant du fichier de configuration de la vue de l'application **PCP Charts** décrit un graphique d'empilement montrant le nombre total d'octets lus et écrits dans le système de fichiers XFS donné **loop1**:

```
#kmchart
version 1

chart title "Filesystem Throughput /loop1" style stacking antialiasing off
    plot legend "Read rate"   metric xfs.read_bytes   instance  "loop1"
    plot legend "Write rate"  metric xfs.write_bytes  instance  "loop1"
```

Ressources supplémentaires

- **pmchart(1)** et **pmtime(1)** pages de manuel
- Outils distribués avec le PCP

7.3. COLLECTE DE DONNÉES À PARTIR D'UN SERVEUR SQL À L'AIDE DE PCP

L'agent SQL Server est disponible dans Performance Co-Pilot (PCP), qui vous aide à surveiller et à analyser les problèmes de performance des bases de données.

Cette procédure décrit comment collecter des données pour Microsoft SQL Server via **pcp** sur votre système.

Conditions préalables

- Vous avez installé Microsoft SQL Server pour Red Hat Enterprise Linux et établi une connexion "fiable" à un serveur SQL.
- Vous avez installé le pilote Microsoft ODBC pour SQL Server pour Red Hat Enterprise Linux.

Procédure

1. Installer le PCP :

```
# dnf install pcp-zeroconf
```

2. Installer les paquets requis pour le pilote **pyodbc**:

```
# dnf install python3-pyodbc
```

3. Installer l'agent **mssql**:

   a. Installer l'agent de domaine Microsoft SQL Server pour PCP :

   ```
   # dnf install pcp-pmda-mssql
   ```

   b. Modifiez le fichier **/etc/pcp/mssql/mssql.conf** pour configurer le nom d'utilisateur et le mot de passe du compte SQL Server pour l'agent **mssql**. Assurez-vous que le compte que vous configurez a des droits d'accès aux données de performance.

   ```
   username: user_name
   password: user_password
   ```

   Remplacez *user_name* par le compte SQL Server et *user_password* par le mot de passe de l'utilisateur SQL Server pour ce compte.

4. Installer l'agent :

```
# cd /var/lib/pcp/pmdas/mssql
# ./Install
Updating the Performance Metrics Name Space (PMNS) ...
Terminate PMDA if already installed ...
Updating the PMCD control file, and notifying PMCD ...
Check mssql metrics have appeared ... 168 metrics and 598 values
[...]
```

Verification steps

- À l'aide de la commande **pcp**, vérifiez si le serveur SQL PMDA (**mssql**) est chargé et en cours d'exécution :

```
$ pcp
Performance Co-Pilot configuration on rhel.local:

platform: Linux rhel.local 4.18.0-167.el8.x86_64 #1 SMP Sun Dec 15 01:24:23 UTC 2019
x86_64
 hardware: 2 cpus, 1 disk, 1 node, 2770MB RAM
 timezone: PDT+7
 services: pmcd pmproxy
    pmcd: Version 5.0.2-1, 12 agents, 4 clients
    pmda: root pmcd proc pmproxy xfs linux nfsclient mmv kvm mssql
          jbd2 dm
 pmlogger: primary logger: /var/log/pcp/pmlogger/rhel.local/20200326.16.31
    pmie: primary engine: /var/log/pcp/pmie/rhel.local/pmie.log
```

- Voir la liste complète des mesures que PCP peut collecter à partir du serveur SQL :

```
# pminfo mssql
```

- Après avoir consulté la liste des mesures, vous pouvez indiquer le taux de transactions. Par exemple, pour connaître le nombre global de transactions par seconde, sur une fenêtre de temps de cinq secondes :

```
# pmval -t 1 -T 5 mssql.databases.transactions
```

- Affichez le graphique de ces mesures sur votre système à l'aide de la commande **pmchart**. Pour plus d'informations, voir Tracer visuellement les archives de journaux PCP avec l'application PCP Charts.

Ressources supplémentaires

- **pcp(1)**, **pminfo(1)**, **pmval(1)**, **pmchart(1)**, et **pmdamssql(1)** pages de manuel
- Performance Co-Pilot for Microsoft SQL Server with RHEL 8.2 Red Hat Developers Blog post

7.4. GÉNÉRER DES ARCHIVES PCP À PARTIR D'ARCHIVES SADC

Vous pouvez utiliser l'outil **sadf** fourni par le paquetage **sysstat** pour générer des archives PCP à partir d'archives natives **sadc**.

Conditions préalables

- Une archive **sadc** a été créée :

```
# /usr/lib64/sa/sadc 1 5 -
```

Dans cet exemple, **sadc** échantillonne les données du système une fois dans un intervalle de 5 secondes. Le fichier de sortie est spécifié comme étant **-**, ce qui fait que **sadc** écrit les données dans le fichier de données quotidiennes standard de l'activité du système. Ce fichier s'appelle saDD et se trouve par défaut dans le répertoire /var/log/sa.

Procédure

- Générer une archive PCP à partir d'une archive **sadc**:

```
# sadf -l -O pcparchive=/tmp/recording -2
```

Dans cet exemple, l'utilisation de l'option **-2** permet à **sadf** de générer une archive PCP à partir d'une archive **sadc** enregistrée il y a 2 jours.

Verification steps

Vous pouvez utiliser les commandes PCP pour inspecter et analyser l'archive PCP générée à partir d'une archive **sadc** comme vous le feriez avec une archive PCP native. Par exemple :

- Pour afficher une liste de métriques dans l'archive PCP générée à partir d'une archive **sadc**, exécutez la commande suivante :

```
$ pminfo --archive /tmp/recording
Disk.dev.avactive
Disk.dev.read
Disk.dev.write
Disk.dev.blkread
[...]
```

- Pour afficher l'espace temporel de l'archive et le nom d'hôte de l'archive PCP, exécutez :

```
$ pmdumplog --label /tmp/recording
Log Label (Log Format Version 2)
Performance metrics from host shard
        commencing Tue Jul 20 00:10:30.642477 2021
        ending     Wed Jul 21 00:10:30.222176 2021
```

- Pour tracer les valeurs des mesures de performance dans des graphiques, exécutez :

```
$ pmchart --archive /tmp/recording
```

CHAPITRE 8. ANALYSE DES PERFORMANCES DE XFS AVEC PCP

L'outil XFS PMDA fait partie du paquetage **pcp** et est activé par défaut lors de l'installation. Il est utilisé pour recueillir des données sur les performances des systèmes de fichiers XFS dans Performance Co-Pilot (PCP).

Vous pouvez utiliser PCP pour analyser les performances du système de fichiers XFS.

8.1. INSTALLATION MANUELLE DE XFS PMDA

Si le PMDA XFS n'est pas répertorié dans la sortie de configuration de **pcp**, installez l'agent PMDA manuellement.

Cette procédure décrit comment installer manuellement l'agent PMDA.

Conditions préalables

- PCP est installé. Pour plus d'informations, voir Installation et activation de PCP.

Procédure

1. Naviguez jusqu'au répertoire xfs :

```
# cd /var/lib/pcp/pmdas/xfs/
```

2. Installer manuellement le PMDA XFS :

```
xfs]# ./Install
Updating the Performance Metrics Name Space (PMNS) ...
Terminate PMDA if already installed ...
Updating the PMCD control file, and notifying PMCD ...
Check xfs metrics have appeared ... 387 metrics and 387 values
```

Verification steps

- Vérifiez que le processus **pmcd** est en cours d'exécution sur l'hôte et que le XFS PMDA est indiqué comme étant activé dans la configuration :

```
# pcp

Performance Co-Pilot configuration on workstation:

platform: Linux workstation 4.18.0-80.el8.x86_64 #1 SMP Wed Mar 13 12:02:46 UTC 2019
x86_64
hardware: 12 cpus, 2 disks, 1 node, 36023MB RAM
timezone: CEST-2
services: pmcd
pmcd: Version 4.3.0-1, 8 agents
pmda: root pmcd proc xfs linux mmv kvm jbd2
```

Ressources supplémentaires

- **pmcd(1)** page de manuel
- Outils distribués avec le PCP

8.2. EXAMEN DES PERFORMANCES DE XFS AVEC PMINFO

PCP permet à XFS PMDA d'établir des rapports sur certaines mesures XFS pour chacun des systèmes de fichiers XFS montés. Il est ainsi plus facile d'identifier les problèmes spécifiques des systèmes de fichiers montés et d'évaluer les performances.

La commande **pminfo** fournit des mesures XFS par périphérique pour chaque système de fichiers XFS monté.

Cette procédure permet d'afficher une liste de toutes les mesures disponibles fournies par le PMDA XFS.

Conditions préalables

- PCP est installé. Pour plus d'informations, voir Installation et activation de PCP.

Procédure

- Affiche la liste de toutes les mesures disponibles fournies par le PMDA XFS :

```
# pminfo xfs
```

- Afficher des informations sur les mesures individuelles. Les exemples suivants examinent les mesures XFS **read** et **write** à l'aide de l'outil **pminfo**:
  - Affiche une brève description de la métrique **xfs.write_bytes**:

```
# pminfo --oneline xfs.write_bytes

xfs.write_bytes [number of bytes written in XFS file system write operations]
```

  - Affiche une longue description de la métrique **xfs.read_bytes**:

```
# pminfo --helptext xfs.read_bytes

xfs.read_bytes
Help:
This is the number of bytes read via read(2) system calls to files in
XFS file systems. It can be used in conjunction with the read_calls
count to calculate the average size of the read operations to file in
XFS file systems.
```

  - Obtenir la valeur de performance actuelle de la métrique **xfs.read_bytes**:

```
# pminfo --fetch xfs.read_bytes

xfs.read_bytes
    value 4891346238
```

  - Obtenir les métriques XFS par périphérique avec **pminfo**:

```
# pminfo --fetch --oneline xfs.perdev.read xfs.perdev.write

xfs.perdev.read [number of XFS file system read operations]
inst [0 or "loop1"] value 0
inst [0 or "loop2"] value 0

xfs.perdev.write [number of XFS file system write operations]
inst [0 or "loop1"] value 86
inst [0 or "loop2"] value 0
```

Ressources supplémentaires

- **pminfo(1)** page de manuel
- Groupes de métriques PCP pour XFS
- Groupes de métriques PCP par périphérique pour XFS

8.3. RÉINITIALISATION DES MESURES DE PERFORMANCE XFS AVEC PMSTORE

Avec PCP, vous pouvez modifier les valeurs de certaines métriques, en particulier si la métrique agit comme une variable de contrôle, comme la métrique **xfs.control.reset**. Pour modifier la valeur d'une métrique, utilisez l'outil **pmstore**.

Cette procédure décrit comment réinitialiser les métriques XFS à l'aide de l'outil **pmstore**.

Conditions préalables

- PCP est installé. Pour plus d'informations, voir Installation et activation de PCP.

Procédure

1. Affiche la valeur d'une métrique :

```
$ pminfo -f xfs.write

xfs.write
    value 325262
```

2. Réinitialiser toutes les mesures XFS :

```
# pmstore xfs.control.reset 1

xfs.control.reset old value=0 new value=1
```

Verification steps

- Afficher les informations après la réinitialisation du système métrique :

```
$ pminfo --fetch xfs.write

xfs.write
```

```
value 0
```

Ressources supplémentaires

- **pmstore(1)** et **pminfo(1)** pages de manuel
- Outils distribués avec le PCP
- Groupes de métriques PCP pour XFS

8.4. GROUPES DE MÉTRIQUES PCP POUR XFS

Le tableau suivant décrit les groupes de métriques PCP disponibles pour XFS.

Tableau 8.1. Groupes de métriques pour XFS

Groupe métrique	Mesures fournies
xfs.*	Métriques XFS générales comprenant le nombre d'opérations de lecture et d'écriture, le nombre d'octets de lecture et d'écriture. Il existe également des compteurs pour le nombre de fois où les inodes sont vidés, mis en cluster et le nombre d'échecs de mise en cluster.
xfs.allocs.* **xfs.alloc_btree.***	Gamme de mesures concernant l'allocation d'objets dans le système de fichiers, notamment le nombre de créations/libres d'étendues et de blocs. Recherche et comparaison de l'arbre d'allocation, ainsi que création et suppression d'enregistrements d'extension dans l'arbre d'allocation.
xfs.block_map.* **xfs.bmap_btree.***	Les mesures comprennent le nombre de lectures/écritures de la carte de blocs et de suppressions de blocs, les opérations de liste d'étendue pour l'insertion, les suppressions et les consultations. Il existe également des compteurs d'opérations pour les comparaisons, les consultations, les insertions et les suppressions de la carte de blocs.
xfs.dir_ops.*	Compteurs pour les opérations de répertoire sur les systèmes de fichiers XFS pour la création, les suppressions d'entrées, le nombre d'opérations "getdent".
xfs.transactions.*	Compteurs pour le nombre de transactions de métadonnées, comprenant le nombre de transactions synchrones et asynchrones ainsi que le nombre de transactions vides.

xfs.inode_ops.*	Compteurs du nombre de fois où le système d'exploitation a recherché un inode XFS dans le cache d'inodes avec différents résultats. Ces compteurs comptabilisent les occurrences dans le cache, les échecs dans le cache, etc.
xfs.log.* **xfs.log_tail.***	Les compteurs du nombre d'écritures du tampon de journal sur les systèmes de fichiers XFS comprennent le nombre de blocs écrits sur le disque. Des mesures sont également disponibles pour le nombre de vidanges et d'épinglages de journaux.
xfs.xstrat.*	Compteurs du nombre d'octets de données de fichiers évacués par le déamon de vidange XFS, ainsi que des compteurs du nombre de tampons évacués vers l'espace contigu et non contigu du disque.
xfs.attr.*	Compte le nombre d'opérations d'obtention, de définition, de suppression et d'énumération d'attributs sur tous les systèmes de fichiers XFS.
xfs.quota.*	Mesures pour le fonctionnement des quotas sur les systèmes de fichiers XFS, y compris les compteurs pour le nombre de réclamations de quotas, les échecs de cache de quotas, les occurrences de cache et les réclamations de données de quotas.
xfs.buffer.*	Gamme de mesures concernant les objets tampons XFS. Les compteurs incluent le nombre d'appels de tampons demandés, de verrous de tampons réussis, de verrous de tampons attendus, de miss_locks, de miss_retries et d'occurrences de tampons lors de la recherche de pages.
xfs.btree.*	Métriques concernant les opérations de l'arborescence XFS.
xfs.control.reset	Métriques de configuration utilisées pour réinitialiser les compteurs de métriques pour les statistiques XFS. Les métriques de contrôle sont modifiées à l'aide de l'outil pmstore.

8.5. GROUPES DE MÉTRIQUES PCP PAR PÉRIPHÉRIQUE POUR XFS

Le tableau suivant décrit le groupe de métriques PCP disponible par périphérique pour XFS.

Tableau 8.2. Groupes de métriques PCP par périphérique pour XFS

Groupe métrique	Mesures fournies
xfs.perdev.*	Métriques XFS générales comprenant le nombre d'opérations de lecture et d'écriture, le nombre d'octets de lecture et d'écriture. Il existe également des compteurs pour le nombre de fois où les inodes sont vidés, mis en cluster et le nombre d'échecs de mise en cluster.
xfs.perdev.allocs.* **xfs.perdev.alloc_btree.***	Gamme de mesures concernant l'allocation d'objets dans le système de fichiers, notamment le nombre de créations/libres d'étendues et de blocs. Recherche et comparaison de l'arbre d'allocation, ainsi que création et suppression d'enregistrements d'extension dans l'arbre d'allocation.
xfs.perdev.block_map.* **xfs.perdev.bmap_btree.***	Les mesures comprennent le nombre de lectures/écritures de la carte de blocs et de suppressions de blocs, les opérations de liste d'étendue pour l'insertion, les suppressions et les consultations. Il existe également des compteurs d'opérations pour les comparaisons, les consultations, les insertions et les suppressions de la carte de blocs.
xfs.perdev.dir_ops.*	Compteurs pour les opérations de répertoire des systèmes de fichiers XFS pour la création, les suppressions d'entrées, le nombre d'opérations "getdent".
xfs.perdev.transactions.*	Compteurs pour le nombre de transactions de métadonnées, comprenant le nombre de transactions synchrones et asynchrones ainsi que le nombre de transactions vides.
xfs.perdev.inode_ops.*	Compteurs du nombre de fois où le système d'exploitation a recherché un inode XFS dans le cache d'inodes avec différents résultats. Ces compteurs comptabilisent les occurrences dans le cache, les échecs dans le cache, etc.
xfs.perdev.log.* **xfs.perdev.log_tail.***	Les compteurs du nombre d'écritures du tampon de journal sur les systèmes de fichiers XFS comprennent le nombre de blocs écrits sur le disque. Des mesures sont également disponibles pour le nombre de vidanges et d'épinglages de journaux.
xfs.perdev.xstrat.*	Compteurs du nombre d'octets de données de fichiers évacués par le déamon de vidange XFS, ainsi que des compteurs du nombre de tampons évacués vers l'espace contigu et non contigu du disque.

xfs.perdev.attr.*	Compte le nombre d'opérations d'obtention, de définition, de suppression et d'énumération d'attributs sur tous les systèmes de fichiers XFS.
xfs.perdev.quota.*	Mesures pour le fonctionnement des quotas sur les systèmes de fichiers XFS, y compris les compteurs pour le nombre de réclamations de quotas, les échecs de cache de quotas, les occurrences de cache et les réclamations de données de quotas.
xfs.perdev.buffer.*	Gamme de mesures concernant les objets tampons XFS. Les compteurs incluent le nombre d'appels de tampons demandés, de verrous de tampons réussis, de verrous de tampons attendus, de miss_locks, de miss_retries et d'occurrences de tampons lors de la recherche de pages.
xfs.perdev.btree.*	Métriques concernant les opérations de l'arborescence XFS.

CHAPITRE 9. MISE EN PLACE D'UNE REPRÉSENTATION GRAPHIQUE DES MESURES PCP

L'utilisation d'une combinaison de **pcp**, **grafana**, **pcp redis**, **pcp bpftrace**, et **pcp vector** fournit une représentation graphique des données en direct ou des données collectées par Performance Co-Pilot (PCP).

9.1. MISE EN PLACE DE PCP AVEC PCP-ZEROCONF

Cette procédure décrit comment configurer PCP sur un système doté du paquetage **pcp-zeroconf**. Une fois le paquetage **pcp-zeroconf** installé, le système enregistre le jeu de métriques par défaut dans des fichiers archivés.

Procédure

- Installez le paquetage **pcp-zeroconf**:

```
# dnf install pcp-zeroconf
```

Verification steps

- Assurez-vous que le service **pmlogger** est actif et qu'il commence à archiver les mesures :

```
# pcp | grep pmlogger
 pmlogger: primary logger: /var/log/pcp/pmlogger/localhost.localdomain/20200401.00.12
```

Ressources supplémentaires

- **pmlogger** page de manuel
- Suivi des performances avec Performance Co-Pilot

9.2. MISE EN PLACE D'UN SERVEUR GRAFANA

Grafana génère des graphiques accessibles depuis un navigateur. Le site **grafana-server** est un serveur dorsal pour le tableau de bord Grafana. Il écoute, par défaut, toutes les interfaces et fournit des services web accessibles via le navigateur web. Le plugin **grafana-pcp** interagit avec le protocole **pmproxy** dans le backend.

Cette procédure décrit comment configurer un site **grafana-server**.

Conditions préalables

- PCP est configuré. Pour plus d'informations, voir Configuration de PCP avec pcp-zeroconf .

Procédure

1. Install the following packages:

```
# dnf install grafana grafana-pcp
```

2. Redémarrez et activez le service suivant :

```
# systemctl restart grafana-server
# systemctl enable grafana-server
```

3. Ouvrez le pare-feu du serveur pour le trafic réseau vers le service Grafana.

```
# firewall-cmd --permanent --add-service=grafana
success

# firewall-cmd --reload
success
```

Verification steps

- S'assurer que le site **grafana-server** est à l'écoute et répond aux demandes :

```
# ss -ntlp | grep 3000
LISTEN  0  128  *:3000  *:*  users:(("grafana-server",pid=19522,fd=7))
```

- Assurez-vous que le plugin **grafana-pcp** est installé :

```
# grafana-cli plugins ls | grep performancecopilot-pcp-app

performancecopilot-pcp-app @ 3.1.0
```

Ressources supplémentaires

- **pmproxy(1)** et **grafana-server** pages de manuel

9.3. ACCÉDER À L'INTERFACE WEB DE GRAFANA

Cette procédure décrit comment accéder à l'interface web de Grafana.

En utilisant l'interface web de Grafana, vous pouvez :

- ajouter les sources de données PCP Redis, PCP bpftrace et PCP Vector
- créer un tableau de bord
- afficher une vue d'ensemble de toutes les mesures utiles
- créer des alertes dans PCP Redis

Conditions préalables

1. PCP est configuré. Pour plus d'informations, voir Configuration de PCP avec pcp-zeroconf .
2. Le site **grafana-server** est configuré. Pour plus d'informations, voir Configuration d'un serveur grafana.

Procédure

1. Sur le système client, ouvrez un navigateur et accédez à **grafana-server** sur le port **3000**, en utilisant le lien *http://192.0.2.0*:3000.

Remplacez *192.0.2.0* par l'IP de votre machine.

2. Pour la première connexion, saisissez **admin** dans les champs **Email or username** et **Password**. Grafana vous invite à définir un **New password** pour créer un compte sécurisé. Si vous souhaitez le définir plus tard, cliquez sur **Skip**.

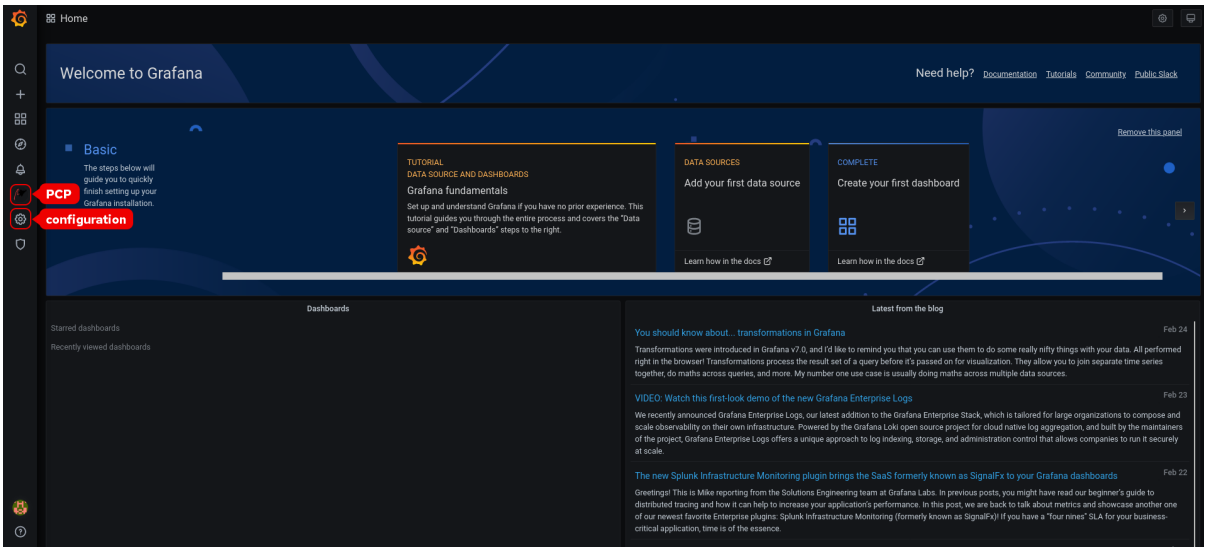
3. Dans le menu, survolez l'icône  **Configuration** puis cliquez sur **Plugins**.

4. Dans l'onglet **Plugins**, tapez performance co-pilot dans la zone de texte **Search by name or type** puis cliquez sur **Performance Co-Pilot** (PCP) plugin.

5. Dans le volet **Plugins / Performance Co-Pilot**, cliquez sur **Activer**.

6. Cliquez sur Grafana  (tourbillon). La page Grafana **Home** s'affiche.

Figure 9.1. Tableau de bord de l'accueil



NOTE

Le coin supérieur de l'écran présente une icône similaire  mais elle contrôle les paramètres généraux de **Dashboard settings**.

7. Dans la page Grafana **Home**, cliquez sur **Add your first data source**pour ajouter les sources de données PCP Redis, PCP bpftrace et PCP Vector. Pour plus d'informations sur l'ajout de sources de données, voir :

   - Pour ajouter la source de données PCP Redis, afficher le tableau de bord par défaut, créer un tableau de bord et une règle d'alerte, voir Création de tableaux de bord et d'alertes dans la source de données PCP Redis.

   - Pour ajouter une source de données pcp bpftrace et afficher le tableau de bord par défaut, voir Affichage du tableau de bord PCP bpftrace System Analysis .

   - Pour ajouter une source de données vectorielles PCP, afficher le tableau de bord par défaut, et pour afficher la liste de contrôle des vecteurs, voir Afficher la liste de contrôle des vecteurs PCP.

8. Facultatif : Dans le menu, survolez l'icône du profil **admin**  pour changer le **Preferences** en **Edit Profile**, **Change Password**, ou en **Sign out**.

Ressources supplémentaires

- **grafana-cli** et **grafana-server** pages de manuel

9.4. CONFIGURER DES CONNEXIONS SÉCURISÉES POUR GRAFANA

Vous pouvez établir des connexions sécurisées entre les composants Grafana et Performance Co-Pilot (PCP). L'établissement de connexions sécurisées entre ces composants permet d'empêcher les parties non autorisées d'accéder aux données collectées et surveillées ou de les modifier.

Conditions préalables

- PCP est installé. Pour plus d'informations, voir Installation et activation de PCP.
- Le site **grafana-server** est configuré. Pour plus d'informations, voir Configuration d'un serveur grafana.
- La clé privée du client est stockée dans le fichier **/etc/grafana/grafana.key**. Si vous utilisez un chemin différent, modifiez-le dans les étapes correspondantes de la procédure.
  Pour plus de détails sur la création d'une clé privée et d'une demande de signature de certificat (CSR), ainsi que sur la manière de demander un certificat à une autorité de certification (AC), consultez la documentation de votre AC.
- Le certificat client TLS est stocké dans le fichier **/etc/grafana/grafana.crt**. Si vous utilisez un chemin différent, modifiez le chemin dans les étapes correspondantes de la procédure.

Procédure

1. En tant qu'utilisateur root, ouvrez le fichier **/etc/grafana/grana.ini** et ajustez les options suivantes dans la section **[server]** pour refléter ce qui suit :

```
protocol = https
cert_key = /etc/grafana/grafana.key
cert_file = /etc/grafana/grafana.crt
```

2. Assurez-vous que grafana peut accéder aux certificats :

```
# su grafana -s /bin/bash -c \
  'ls -1 /etc/grafana/grafana.crt /etc/grafana/grafana.key'
/etc/grafana/grafana.crt
/etc/grafana/grafana.key
```

3. Redémarrez et activez le service Grafana pour appliquer les modifications de configuration :

```
# systemctl restart grafana-server
# systemctl enable grafana-server
```

Vérification

1. Sur le système client, ouvrez un navigateur et accédez à la machine **grafana-server** sur le port 3000, en utilisant le lien https://192.0.2.0:3000. Remplacez 192.0.2.0 par l'IP de votre machine.

2. Confirmer l'icône l'icône de verrouillage s'affiche à côté de la barre d'adresse.

NOTE

Si le protocole est défini sur **http** et qu'une connexion HTTPS est tentée, vous recevrez une erreur **ERR_SSL_PROTOCOL_ERROR**. Si le protocole est défini sur **https** et qu'une connexion HTTP est tentée, le serveur Grafana répond par un message "Client sent an HTTP request to an HTTPS server".

9.5. CONFIGURATION DE PCP REDIS

Utilisez la source de données Redis de PCP pour :

- Voir les archives de données
- Interroger les séries temporelles à l'aide du langage pmseries
- Analyser les données sur plusieurs hôtes

Conditions préalables

1. PCP est configuré. Pour plus d'informations, voir Configuration de PCP avec pcp-zeroconf .
2. Le site **grafana-server** est configuré. Pour plus d'informations, voir Configuration d'un serveur grafana.

Procédure

1. Installez le paquetage **redis**:

```
# dnf install redis
```

2. Démarrez et activez les services suivants :

```
# systemctl start pmproxy redis
# systemctl enable pmproxy redis
```

3. L'agent de transfert de courrier, par exemple **sendmail** ou **postfix**, est installé et configuré.

4. Assurez-vous que le paramètre **allow_loading_unsigned_plugins** est défini sur la base de données PCP Redis dans le fichier **grafana.ini**:

```
# vi /etc/grafana/grafana.ini

allow_loading_unsigned_plugins = pcp-redis-datasource
```

5. Redémarrer le site **grafana-server**:

```
# systemctl restart grafana-server
```

Verification steps

- Assurez-vous que les sites **pmproxy** et **redis** fonctionnent :

```
# pmseries disk.dev.read
2eb3e58d8f1e231361fb15cf1aa26fe534b4d9df
```

  Cette commande ne renvoie aucune donnée si le paquetage **redis** n'est pas installé.

Ressources supplémentaires

- **pmseries(1)** page de manuel

9.6. CONFIGURER DES CONNEXIONS SÉCURISÉES POUR PCP REDIS

Vous pouvez établir des connexions sécurisées entre Performance Co-Pilot (PCP), Grafana et PCP Redis. L'établissement de connexions sécurisées entre ces composants permet d'empêcher les parties non autorisées d'accéder aux données collectées et surveillées ou de les modifier.

Conditions préalables

- PCP est installé. Pour plus d'informations, voir Installation et activation de PCP.
- Le site **grafana-server** est configuré. Pour plus d'informations, voir Configuration d'un serveur grafana.
- PCP redis est installé. Pour plus d'informations, voir Configuration de PCP Redis.
- La clé privée du client est stockée dans le fichier **/etc/redis/client.key**. Si vous utilisez un chemin différent, modifiez-le dans les étapes correspondantes de la procédure.
  Pour plus de détails sur la création d'une clé privée et d'une demande de signature de certificat (CSR), ainsi que sur la manière de demander un certificat à une autorité de certification (AC), consultez la documentation de votre AC.
- Le certificat client TLS est stocké dans le fichier **/etc/redis/client.crt**. Si vous utilisez un chemin différent, modifiez le chemin dans les étapes correspondantes de la procédure.
- La clé du serveur TLS est stockée dans le fichier **/etc/redis/redis.key**. Si vous utilisez un chemin différent, modifiez le chemin dans les étapes correspondantes de la procédure.
- Le certificat du serveur TLS est stocké dans le fichier **/etc/redis/redis.crt**. Si vous utilisez un chemin différent, modifiez le chemin dans les étapes correspondantes de la procédure.
- Le certificat CA est stocké dans le fichier **/etc/redis/ca.crt**. Si vous utilisez un chemin différent, modifiez-le dans les étapes correspondantes de la procédure.

En outre, pour le démon **pmproxy**:

- La clé privée du serveur est stockée dans le fichier **/etc/pcp/tls/server.key**. Si vous utilisez un chemin différent, modifiez le chemin dans les étapes correspondantes de la procédure.

Procédure

1. En tant qu'utilisateur root, ouvrez le fichier **/etc/redis/redis.conf** et ajustez les options TLS/SSL pour refléter les propriétés suivantes :

```
port 0
tls-port 6379
tls-cert-file /etc/redis/redis.crt
tls-key-file /etc/redis/redis.key
tls-client-key-file /etc/redis/client.key
tls-client-cert-file /etc/redis/client.crt
tls-ca-cert-file /etc/redis/ca.crt
```

2. Assurez-vous que **redis** peut accéder aux certificats TLS :

```
# su redis -s /bin/bash -c \
  'ls -1 /etc/redis/ca.crt /etc/redis/redis.key /etc/redis/redis.crt'
/etc/redis/ca.crt
/etc/redis/redis.crt
/etc/redis/redis.key
```

3. Redémarrez le serveur **redis** pour appliquer les changements de configuration :

```
# systemctl restart redis
```

Vérification

- Confirmez que la configuration TLS fonctionne :

```
# redis-cli --tls --cert /etc/redis/client.crt \
    --key /etc/redis/client.key \
    --cacert /etc/redis/ca.crt <<< "PING"
PONG
```

Une configuration TLS infructueuse peut entraîner le message d'erreur suivant :

```
Could not negotiate a TLS connection: Invalid CA Certificate File/Directory
```

9.7. CRÉATION DE PANNEAUX ET D'ALERTES DANS LA SOURCE DE DONNÉES REDIS DE PCP

Après avoir ajouté la source de données PCP Redis, vous pouvez afficher le tableau de bord avec un aperçu des mesures utiles, ajouter une requête pour visualiser le graphique de charge et créer des alertes qui vous aideront à visualiser les problèmes du système après qu'ils se soient produits.

Conditions préalables

1. Le PCP Redis est configuré. Pour plus d'informations, voir Configuration de PCP Redis.
2. Le site **grafana-server** est accessible. Pour plus d'informations, voir Accéder à l'interface web de Grafana.

Procédure

1. Connectez-vous à l'interface web de Grafana.
2. Dans la page Grafana **Home**, cliquez sur **Add your first data source**

3. Dans le volet **Add data source**, tapez redis dans la zone de texte **Filter by name or type** et cliquez sur **PCP Redis**.

4. Dans le volet **Data Sources / PCP Redis**, effectuez les opérations suivantes :

   a. Ajoutez **http://localhost:44322** dans le champ **URL** et cliquez sur **Save & Test**.

   b. Cliquez sur **Onglet Tableaux de bord → Importer → PCP Redis : Aperçu de l'hôte** pour afficher un tableau de bord avec une vue d'ensemble de toutes les mesures utiles.

   Figure 9.2. PCP Redis : Présentation de l'hôte

5. Ajouter un nouveau panneau :

   a. Dans le menu, survolez le signe  **Icône de création → Tableau de bord → Ajouter une nouvelle icône de tableau de bord** pour ajouter un tableau de bord.

   b. Dans l'onglet **Query**, sélectionnez l'option **PCP Redis** dans la liste des requêtes au lieu de l'option **default** et dans le champ de texte **A**, entrez une métrique, par exemple **kernel.all.load** pour visualiser le graphique de charge du noyau.

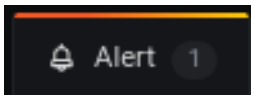
   c. Optionnel : Ajoutez **Panel title** et **Description**, et mettez à jour les autres options du site **Settings**.

   d. Cliquez sur **Enregistrer** pour appliquer les modifications et sauvegarder le tableau de bord. Ajouter **Dashboard name**.

   e. Cliquez sur **Appliquer** pour appliquer les modifications et revenir au tableau de bord.

Figure 9.3. Panneau de requêtes Redis PCP

6. Créer une règle d'alerte :

   a. Dans le site **PCP Redis query panel**, cliquez sur 

Alert puis cliquez sur **Create Alert**.

   b. Modifiez les champs **Name**, **Evaluate query**, et **For** à partir du champ **Rule**, et spécifiez le champ **Conditions** pour votre alerte.

   c. Cliquez sur **Enregistrer** pour appliquer les modifications et sauvegarder le tableau de bord. Cliquez sur **Appliquer** pour appliquer les modifications et revenir au tableau de bord.

Figure 9.4. Création d'alertes dans le tableau de bord Redis de PCP

   d. Facultatif : dans le même panneau, faites défiler vers le bas et cliquez sur l'icône **Supprimer** pour supprimer la règle créée.

   e. Optionnel : Dans le menu, cliquez sur  **Alerting** pour afficher les règles d'alerte créées avec différents états d'alerte, pour modifier la règle d'alerte ou pour mettre en pause la règle existante à partir de l'onglet **Alert Rules**.

Pour ajouter un canal de notification pour la règle d'alerte créée afin de recevoir une notification d'alerte de Grafana, voir Ajouter des canaux de notification pour les alertes .

9.8. AJOUT DE CANAUX DE NOTIFICATION POUR LES ALERTES

En ajoutant des canaux de notification, vous pouvez recevoir une notification d'alerte de Grafana chaque fois que les conditions de la règle d'alerte sont remplies et que le système nécessite une surveillance supplémentaire.

Vous pouvez recevoir ces alertes après avoir sélectionné un type parmi la liste des notificateurs pris en charge, qui comprend **DingDing**, **Discord**, **Email**, **Google Hangouts Chat**, **HipChat**, **Kafka REST Proxy**, **LINE**, **Microsoft Teams**, **OpsGenie**, **PagerDuty**, **Prometheus Alertmanager**, **Pushover**, **Sensu**, **Slack**, **Telegram**, **Threema Gateway**, **VictorOps**, et **webhook**.

Conditions préalables

1. Le site **grafana-server** est accessible. Pour plus d'informations, voir Accéder à l'interface web de Grafana.
2. Une règle d'alerte est créée. Pour plus d'informations, voir Création de panneaux et d'alertes dans la source de données Redis de PCP.
3. Configurez le SMTP et ajoutez une adresse électronique d'expéditeur valide dans le fichier **grafana/grafana.ini**:

```
# vi /etc/grafana/grafana.ini

[smtp]
enabled = true
from_address = abc@gmail.com
```

   Remplacez *abc@gmail.com* par une adresse électronique valide.

4. Redémarrage **grafana-server**

```
# systemctl restart grafana-server.service
```

Procédure

1. Dans le menu, survolez l'icône  **Icône d'alerte → cliquez sur Points de contact → Nouveau point de contact**.
2. Dans la vue détaillée de **New contact point**, procédez comme suit :
   a. Saisissez votre nom dans la zone de texte **Name**
   b. Sélectionnez l'option **Contact point type**, par exemple, Email et saisissez l'adresse électronique. Vous pouvez ajouter plusieurs adresses électroniques en utilisant le séparateur **;**.
   c. En option : Configurez **Optional Email settings** et **Notification settings**.
3. Cliquez sur **Enregistrer le point de contact**.

4. Sélectionnez un canal de notification dans la règle d'alerte :

   a. Dans le menu, sélectionnez l'icône **Notification policies** puis cliquez sur **New specific policy**.

   b. Choisissez le site **Contact point** que vous venez de créer

   c. Cliquez sur le bouton **Save policy**

Ressources supplémentaires

- Documentation Grafana en amont pour les notifications d'alertes

9.9. MISE EN PLACE DE L'AUTHENTIFICATION ENTRE LES COMPOSANTS DU PCP

Vous pouvez configurer l'authentification à l'aide du mécanisme d'authentification **scram-sha-256**, qui est pris en charge par PCP via le cadre SASL (Simple Authentication Security Layer).

Procédure

1. Installez le cadre **sasl** pour le mécanisme d'authentification **scram-sha-256**:

```
# dnf install cyrus-sasl-scram cyrus-sasl-lib
```

2. Spécifiez le mécanisme d'authentification pris en charge et le chemin d'accès à la base de données des utilisateurs dans le fichier **pmcd.conf**:

```
# vi /etc/sasl2/pmcd.conf

mech_list: scram-sha-256

sasldb_path: /etc/pcp/passwd.db
```

3. Créer un nouvel utilisateur :

```
# useradd -r metrics
```

   Remplacez *metrics* par votre nom d'utilisateur.

4. Ajouter l'utilisateur créé dans la base de données des utilisateurs :

```
# saslpasswd2 -a pmcd metrics

Password:
Again (for verification):
```

   Pour ajouter l'utilisateur créé, vous devez saisir le mot de passe du compte *metrics*.

5. Définir les autorisations de la base de données des utilisateurs :

```
# chown root:pcp /etc/pcp/passwd.db
# chmod 640 /etc/pcp/passwd.db
```

6. Redémarrez le service **pmcd**:

```
# systemctl restart pmcd
```

Verification steps

- Vérifiez la configuration de **sasl**:

```
# pminfo -f -h "pcp://127.0.0.1?username=metrics" disk.dev.read
Password:
disk.dev.read
inst [0 or "sda"] value 19540
```

Ressources supplémentaires

- **saslauthd(8)**, **pminfo(1)**, et **sha256** pages de manuel
- Comment puis-je configurer l'authentification entre les composants PCP, tels que PMDA et pmcd dans RHEL 8.2 ?

9.10. INSTALLATION DE PCP BPFTRACE

Installez l'agent PCP **bpftrace** afin d'introspecter un système et de recueillir des mesures à partir des points de contrôle du noyau et de l'espace utilisateur.

L'agent **bpftrace** utilise des scripts bpftrace pour collecter les mesures. Les scripts **bpftrace** utilisent le filtre de paquets Berkeley amélioré (**eBPF**).

Cette procédure décrit comment installer un **pcp bpftrace**.

Conditions préalables

1. PCP est configuré. Pour plus d'informations, voir Configuration de PCP avec pcp-zeroconf .
2. Le site **grafana-server** est configuré. Pour plus d'informations, voir Configuration d'un serveur grafana.
3. Le mécanisme d'authentification **scram-sha-256** est configuré. Pour plus d'informations, voir Configuration de l'authentification entre les composants PCP .

Procédure

1. Installez le paquetage **pcp-pmda-bpftrace**:

```
# dnf install pcp-pmda-bpftrace
```

2. Modifiez le fichier **bpftrace.conf** et ajoutez l'utilisateur que vous avez créé dans le fichier {setting-up-authentication-between-pcp-components} :

```
# vi /var/lib/pcp/pmdas/bpftrace/bpftrace.conf

[dynamic_scripts]
```

```
enabled = true
auth_enabled = true
allowed_users = root,metrics
```

Remplacez *metrics* par votre nom d'utilisateur.

3. Installer **bpftrace** PMDA :

```
# cd /var/lib/pcp/pmdas/bpftrace/
# ./Install
Updating the Performance Metrics Name Space (PMNS) ...
Terminate PMDA if already installed ...
Updating the PMCD control file, and notifying PMCD ...
Check bpftrace metrics have appeared ... 7 metrics and 6 values
```

Le site **pmda-bpftrace** est maintenant installé et ne peut être utilisé qu'après authentification de l'utilisateur. Pour plus d'informations, voir Affichage du tableau de bord d'analyse du système PCP bpftrace.

Ressources supplémentaires

- **pmdabpftrace(1)** et **bpftrace** pages de manuel

9.11. VISUALISATION DU TABLEAU DE BORD D'ANALYSE DU SYSTÈME PCP BPFTRACE

En utilisant la source de données PCP bpftrace, vous pouvez accéder aux données en direct provenant de sources qui ne sont pas disponibles en tant que données normales sur le site **pmlogger** ou dans les archives

Dans la source de données PCP bpftrace, vous pouvez consulter le tableau de bord avec une vue d'ensemble des mesures utiles.

Conditions préalables

1. Le programme PCP bpftrace est installé. Pour plus d'informations, voir Installation de PCP bpftrace.
2. Le site **grafana-server** est accessible. Pour plus d'informations, voir Accéder à l'interface web de Grafana.

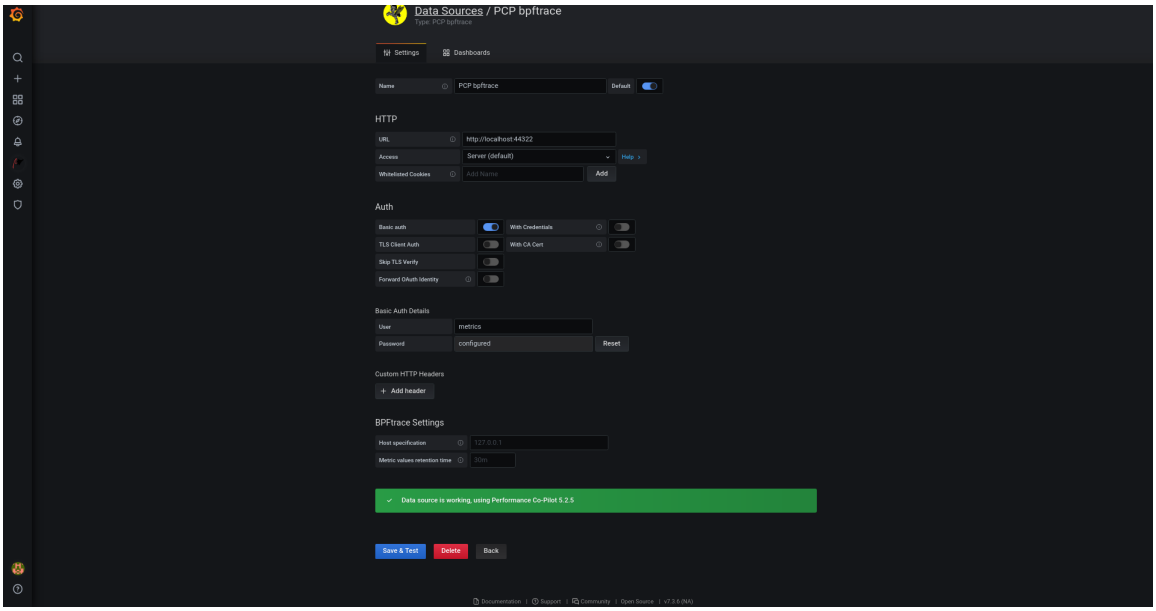
Procédure

1. Connectez-vous à l'interface web de Grafana.
2. Dans la page Grafana **Home**, cliquez sur **Add your first data source**
3. Dans le volet **Add data source**, tapez bpftrace dans la zone de texte **Filter by name or type** et cliquez sur **PCP bpftrace**.
4. Dans le volet **Data Sources / PCP bpftrace**, effectuez les opérations suivantes :
   a. Ajoutez **http://localhost:44322** dans le champ **URL**.

b. Activez l'option **Basic Auth** et ajoutez les informations d'identification de l'utilisateur créé dans les champs **User** et **Password**.

c. Cliquez sur **Enregistrer & Test**.

Figure 9.5. Ajout du PCP bpftrace dans la source de données



d. Cliquez sur **Onglet Tableaux de bord → Importer → PCP bpftrace : Analyse du système** pour afficher un tableau de bord avec une vue d'ensemble de toutes les mesures utiles.

Figure 9.6. PCP bpftrace : Analyse du système

9.12. INSTALLATION DE PCP VECTOR

Cette procédure décrit comment installer un **pcp vector**.

Conditions préalables

1. PCP est configuré. Pour plus d'informations, voir Configuration de PCP avec pcp-zeroconf .

2. Le site **grafana-server** est configuré. Pour plus d'informations, voir Configuration d'un serveur grafana.

Procédure

1. Installez le paquetage **pcp-pmda-bcc**:

```
# dnf install pcp-pmda-bcc
```

2. Installez le PMDA **bcc**:

```
# cd /var/lib/pcp/pmdas/bcc
# ./Install
[Wed Apr  1 00:27:48] pmdabcc(22341) Info: Initializing, currently in 'notready' state.
[Wed Apr  1 00:27:48] pmdabcc(22341) Info: Enabled modules:
[Wed Apr  1 00:27:48] pmdabcc(22341) Info: ['biolatency', 'sysfork',
[...]
Updating the Performance Metrics Name Space (PMNS) ...
Terminate PMDA if already installed ...
Updating the PMCD control file, and notifying PMCD ...
Check bcc metrics have appeared ... 1 warnings, 1 metrics and 0 values
```

Ressources supplémentaires

- **pmdabcc(1)** page de manuel

9.13. VISUALISATION DE LA LISTE DE CONTRÔLE DU VECTEUR PCP

La source de données PCP Vector affiche des métriques en temps réel et utilise les métriques **pcp**. Elle analyse les données pour des hôtes individuels.

Après avoir ajouté la source de données PCP Vector, vous pouvez afficher le tableau de bord avec une vue d'ensemble des mesures utiles et consulter les liens de dépannage ou de référence dans la liste de contrôle.

Conditions préalables

1. Le PCP Vector est installé. Pour plus d'informations, voir Installation de PCP Vector.
2. Le site **grafana-server** est accessible. Pour plus d'informations, voir Accéder à l'interface web de Grafana.

Procédure

1. Connectez-vous à l'interface web de Grafana.
2. Dans la page Grafana **Home**, cliquez sur **Add your first data source**
3. Dans le volet **Add data source**, tapez vector dans la zone de texte **Filter by name or type** et cliquez sur **PCP Vector**.
4. Dans le volet **Data Sources / PCP Vector**, effectuez les opérations suivantes :
   a. Ajoutez **http://localhost:44322** dans le champ **URL** et cliquez sur **Save & Test**.

b. Cliquez sur **Onglet Tableaux de bord → Importer → Vecteur PCP : Aperçu de l'hôte**pour afficher un tableau de bord avec une vue d'ensemble de toutes les mesures utiles.

Figure 9.7. Vecteur PCP : Présentation de l'hôte

5. Dans le menu, survolez la rubrique  **Performance Co-Pilot** puis cliquez sur **PCP Vector Checklist**.

   Dans la liste de contrôle PCP, cliquez sur  ou sur  pour afficher les liens de dépannage ou de référence correspondants.

Figure 9.8. Liste de contrôle du vecteur du copilote de performance / PCP

9.14. RÉSOLUTION DES PROBLÈMES LIÉS À GRAFANA

Il est parfois nécessaire de résoudre des problèmes liés à Grafana, tels que Grafana n'affiche aucune donnée, le tableau de bord est noir, ou d'autres problèmes similaires.

Procédure

- Vérifiez que le service **pmlogger** est opérationnel en exécutant la commande suivante :

```
$ systemctl status pmlogger
```

- Vérifiez si des fichiers ont été créés ou modifiés sur le disque en exécutant la commande suivante :

```
$ ls /var/log/pcp/pmlogger/$(hostname)/ -rlt
total 4024
-rw-r--r--. 1 pcp pcp   45996 Oct 13  2019 20191013.20.07.meta.xz
-rw-r--r--. 1 pcp pcp     412 Oct 13  2019 20191013.20.07.index
-rw-r--r--. 1 pcp pcp   32188 Oct 13  2019 20191013.20.07.0.xz
-rw-r--r--. 1 pcp pcp   44756 Oct 13  2019 20191013.20.30-00.meta.xz
[..]
```

- Vérifiez que le service **pmproxy** fonctionne en exécutant la commande suivante :

```
$ systemctl status pmproxy
```

- Vérifiez que **pmproxy** est en cours d'exécution, que la prise en charge des séries temporelles est activée et qu'une connexion à Redis est établie en consultant le fichier **/var/log/pcp/pmproxy/pmproxy.log** et en vous assurant qu'il contient le texte suivant :

```
pmproxy(1716) Info: Redis slots, command keys, schema version setup
```

  Ici, **1716** est le PID de pmproxy, qui sera différent pour chaque invocation de **pmproxy**.

- Vérifiez si la base de données Redis contient des clés en exécutant la commande suivante :

```
$ redis-cli dbsize
(integer) 34837
```

- Vérifiez si des mesures PCP se trouvent dans la base de données Redis et si **pmproxy** est en mesure d'y accéder en exécutant les commandes suivantes :

```
$ pmseries disk.dev.read
2eb3e58d8f1e231361fb15cf1aa26fe534b4d9df

$ pmseries "disk.dev.read[count:10]"
2eb3e58d8f1e231361fb15cf1aa26fe534b4d9df
    [Mon Jul 26 12:21:10.085468000 2021] 117971
70e83e88d4e1857a3a31605c6d1333755f2dd17c
    [Mon Jul 26 12:21:00.087401000 2021] 117758
70e83e88d4e1857a3a31605c6d1333755f2dd17c
    [Mon Jul 26 12:20:50.085738000 2021] 116688
70e83e88d4e1857a3a31605c6d1333755f2dd17c
[...]
```

```
$ redis-cli --scan --pattern "*$(pmseries 'disk.dev.read')"

pcp:metric.name:series:2eb3e58d8f1e231361fb15cf1aa26fe534b4d9df
pcp:values:series:2eb3e58d8f1e231361fb15cf1aa26fe534b4d9df
pcp:desc:series:2eb3e58d8f1e231361fb15cf1aa26fe534b4d9df
pcp:labelvalue:series:2eb3e58d8f1e231361fb15cf1aa26fe534b4d9df
pcp:instances:series:2eb3e58d8f1e231361fb15cf1aa26fe534b4d9df
pcp:labelflags:series:2eb3e58d8f1e231361fb15cf1aa26fe534b4d9df
```

-

- Vérifiez s'il y a des erreurs dans les journaux de Grafana en exécutant la commande suivante :

```
$ journalctl -e -u grafana-server
-- Logs begin at Mon 2021-07-26 11:55:10 IST, end at Mon 2021-07-26 12:30:15 IST. --
Jul 26 11:55:17 localhost.localdomain systemd[1]: Starting Grafana instance...
Jul 26 11:55:17 localhost.localdomain grafana-server[1171]: t=2021-07-26T11:55:17+0530
lvl=info msg="Starting Grafana" logger=server version=7.3.6 c>
Jul 26 11:55:17 localhost.localdomain grafana-server[1171]: t=2021-07-26T11:55:17+0530
lvl=info msg="Config loaded from" logger=settings file=/usr/s>
Jul 26 11:55:17 localhost.localdomain grafana-server[1171]: t=2021-07-26T11:55:17+0530
lvl=info msg="Config loaded from" logger=settings file=/etc/g>
[...]
```

CHAPITRE 10. OPTIMISER LES PERFORMANCES DU SYSTÈME À L'AIDE DE LA CONSOLE WEB

Découvrez comment définir un profil de performance dans la console web RHEL afin d'optimiser les performances du système pour une tâche donnée.

10.1. OPTIONS DE RÉGLAGE DES PERFORMANCES DANS LA CONSOLE WEB

Red Hat Enterprise Linux 9 fournit plusieurs profils de performance qui optimisent le système pour les tâches suivantes :

- Systèmes utilisant le bureau
- Performance en termes de débit
- Performance en matière de latence
- Performance du réseau
- Faible consommation d'énergie
- Machines virtuelles

Le service **TuneD** optimise les options du système en fonction du profil sélectionné.

Dans la console web, vous pouvez définir le profil de performance utilisé par votre système.

Ressources supplémentaires

- Démarrer avec TuneD

10.2. DÉFINITION D'UN PROFIL DE PERFORMANCE DANS LA CONSOLE WEB

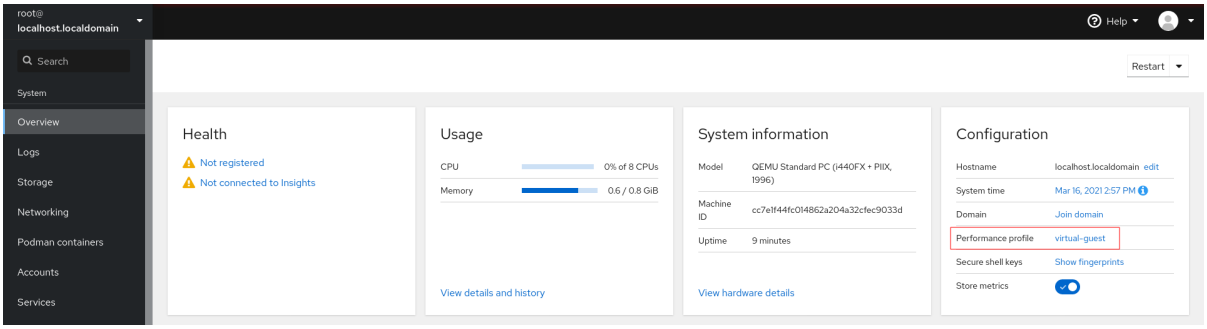
Cette procédure utilise la console web pour optimiser les performances du système pour une tâche sélectionnée.

Conditions préalables

- Assurez-vous que la console web est installée et accessible. Pour plus de détails, voir Installation de la console web.

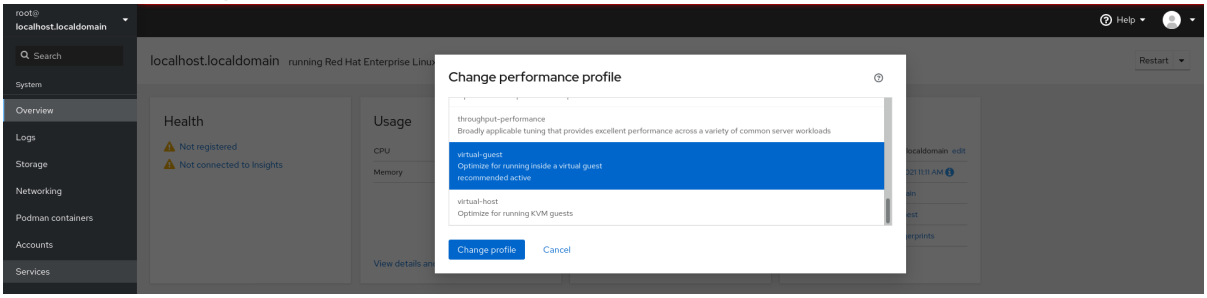
Procédure

1. Connectez-vous à la console web RHEL. Pour plus d'informations, voir Connexion à la console web.
2. Cliquez sur **Overview**.
3. Dans le champ **Performance Profile**, cliquez sur le profil de performance actuel.



4. Dans la boîte de dialogue **Change Performance Profile**, modifiez le profil si nécessaire.

5. Cliquez sur **Change Profile**.



Verification steps

- L'onglet **Overview** affiche désormais le profil de performance sélectionné.

10.3. CONTRÔLE DES PERFORMANCES SUR LE SYSTÈME LOCAL À L'AIDE DE LA CONSOLE WEB

La console web de Red Hat Enterprise Linux utilise la méthode USE (Utilization Saturation and Errors) pour le dépannage. La nouvelle page de mesures de performance présente une vue historique de vos données organisée chronologiquement avec les données les plus récentes en haut de la page.

Vous pouvez y consulter les événements, les erreurs et la représentation graphique de l'utilisation et de la saturation des ressources.

Conditions préalables

- La console web est installée et accessible. Pour plus de détails, voir Installation de la console web.

- Le paquet **cockpit-pcp**, qui permet de collecter les mesures de performance, est installé :

  a. Pour installer le paquet à partir de l'interface de la console web :

     i. Connectez-vous à la console web avec des privilèges administratifs. Pour plus d'informations, voir Connexion à la console web .

     ii. Dans la page **Overview**, cliquez sur **View details and history**.

     iii. Cliquez sur le bouton **Installer cockpit-pcp**.

     iv. Dans la fenêtre de dialogue **Install software**, cliquez sur **Install**.

  b. Pour installer le paquet à partir de l'interface de ligne de commande, utilisez :

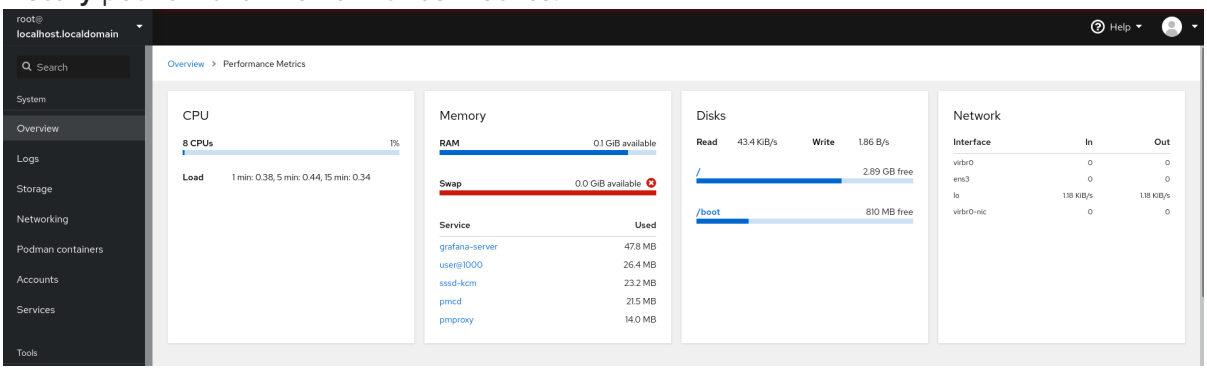
```
# dnf install cockpit-pcp
```

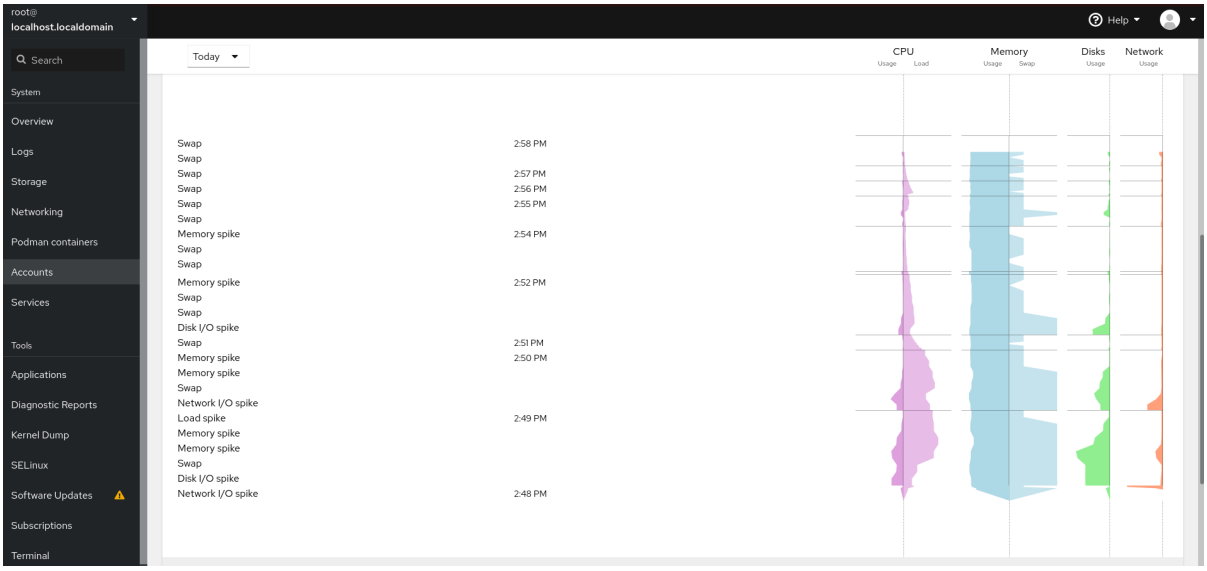
- Le service PCP est activé :

```
# systemctl enable --now pmlogger.service pmproxy.service
```

Procédure

1. Connectez-vous à la console web RHEL 9. Dans la page **Overview**, cliquez sur **View details and history** pour afficher **Performance Metrics**.





10.4. SURVEILLANCE DES PERFORMANCES SUR PLUSIEURS SYSTÈMES À L'AIDE DE LA CONSOLE WEB ET DE GRAFANA

Grafana vous permet de collecter des données à partir de plusieurs systèmes à la fois et d'examiner une représentation graphique de leurs métriques PCP collectées. Vous pouvez configurer la surveillance et l'exportation des mesures de performance pour plusieurs systèmes dans l'interface de la console web.

Cette procédure vous montre comment activer l'exportation des mesures de performance avec PCP à partir de l'interface de la console Web de RHEL 9.

Conditions préalables

- La console web doit être installée et accessible. Pour plus de détails, voir Installation de la console web.
- Installez le paquetage **cockpit-pcp**.

1. Depuis l'interface de la console web :
   a. Connectez-vous à la console web avec des privilèges administratifs. Pour plus d'informations, voir Connexion à la console web .
   b. Dans la page **Overview**, cliquez sur **View details and history**.
   c. Cliquez sur le bouton **Installer cockpit-pcp**.
   d. Dans la fenêtre de dialogue **Install software**, cliquez sur **Install**.
   e. Déconnectez-vous et reconnectez-vous pour voir l'historique des mesures.
2. Pour installer le paquet à partir de l'interface de ligne de commande, utilisez :

```
# dnf install cockpit-pcp
```

- Activer le service PCP :

```
# systemctl enable --now pmlogger.service pmproxy.service
```

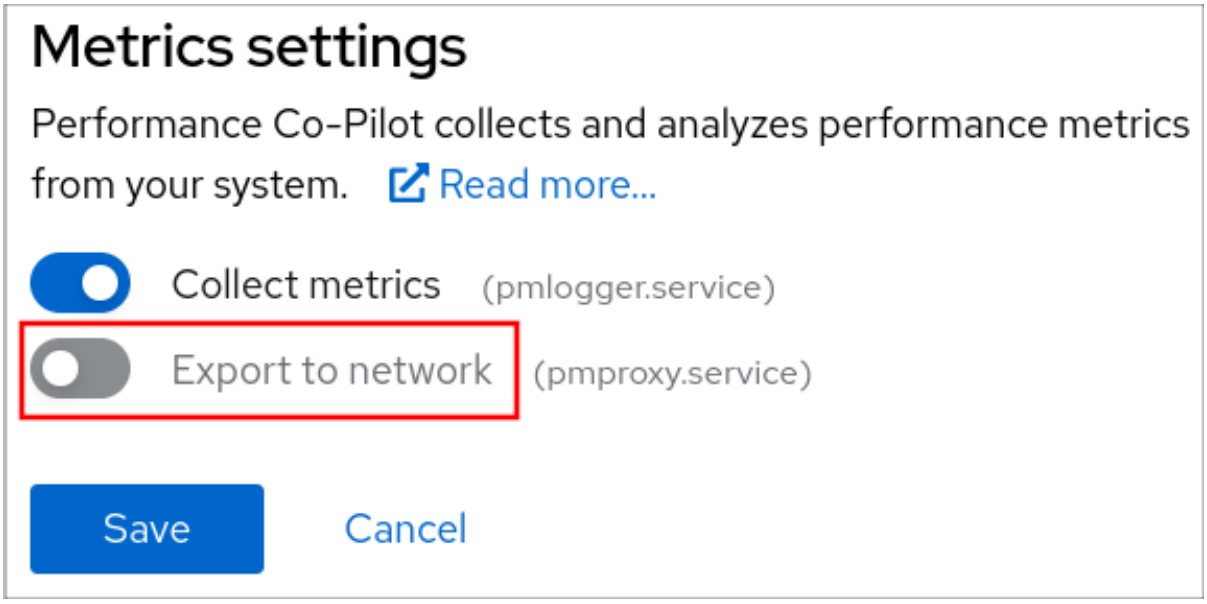
- Configurez le tableau de bord Grafana. Pour plus d'informations, voir Configurer un serveur Grafana.
- Installez le paquetage **redis**.

```
# dnf install redis
```

Vous pouvez également installer le paquet à partir de l'interface de la console web plus tard dans la procédure.

Procédure

1. Dans la page **Overview**, cliquez sur **View details and history** dans le tableau **Usage**.
2. Cliquez sur le bouton **Paramètres de mesure**.
3. Placez le curseur **Export to network** en position active.



Si le service **redis** n'est pas installé, vous serez invité à l'installer.

4. Pour ouvrir le service **pmproxy**, sélectionnez une zone dans une liste déroulante et cliquez sur le bouton **Add pmproxy**.

5. Cliquez sur **Save**.

Vérification

1. Cliquez sur **Networking**.

2. Dans le tableau **Firewall**, cliquez sur **n active zones** ou sur le bouton **Modifier les règles et les zones**.

3. Recherchez **pmproxy** dans la zone que vous avez sélectionnée.

IMPORTANT

Répétez cette procédure sur tous les systèmes que vous souhaitez surveiller.

CHAPITRE 11. PARAMÉTRAGE DU PLANIFICATEUR DE DISQUE

L'ordonnanceur de disque est chargé d'ordonner les demandes d'E/S soumises à un périphérique de stockage.

Vous pouvez configurer le planificateur de différentes manières :

- Définissez le planificateur à l'aide de **TuneD**, comme décrit dans la section Définition du planificateur de disque à l'aide de TuneD
- Définissez le planificateur à l'aide de **udev**, comme décrit dans la section Définition du planificateur de disque à l'aide des règles udev
- Modifier temporairement le planificateur d'un système en cours d'exécution, comme décrit dans la section Configuration temporaire d'un planificateur pour un disque spécifique

NOTE

Dans Red Hat Enterprise Linux 9, les périphériques de blocs ne prennent en charge que l'ordonnancement à plusieurs files d'attente. Cela permet aux performances de la couche de blocs de bien s'adapter aux disques durs rapides (SSD) et aux systèmes multicœurs.

Les ordonnanceurs traditionnels à file d'attente unique, qui étaient disponibles dans Red Hat Enterprise Linux 7 et les versions antérieures, ont été supprimés.

11.1. PLANIFICATEURS DE DISQUES DISPONIBLES

Les ordonnanceurs de disques à plusieurs files d'attente suivants sont pris en charge dans Red Hat Enterprise Linux 9 :

none

Met en œuvre un algorithme d'ordonnancement FIFO (premier entré, premier sorti). Il fusionne les demandes au niveau du bloc générique par le biais d'un simple cache de dernier arrivé.

mq-deadline

Tente de fournir une latence garantie pour les demandes à partir du moment où les demandes atteignent l'ordonnanceur.
L'ordonnanceur **mq-deadline** classe les demandes d'E/S en file d'attente dans un lot de lecture ou d'écriture, puis planifie leur exécution dans l'ordre croissant de l'adressage logique des blocs (LBA). Par défaut, les lots de lecture sont prioritaires sur les lots d'écriture, car les applications sont plus susceptibles de se bloquer sur les opérations d'E/S de lecture. Après que **mq-deadline** a traité un lot, il vérifie pendant combien de temps les opérations d'écriture ont été privées de temps processeur et planifie le lot de lecture ou d'écriture suivant, selon le cas.

Cet ordonnanceur convient à la plupart des cas d'utilisation, mais surtout à ceux dans lesquels les opérations d'écriture sont principalement asynchrones.

bfq

Cible les systèmes de bureau et les tâches interactives.
Le planificateur **bfq** garantit qu'une seule application n'utilise jamais la totalité de la bande passante. En effet, l'unité de stockage est toujours aussi réactive que si elle était inactive. Dans sa configuration par défaut, **bfq** s'efforce de fournir la latence la plus faible possible plutôt que d'atteindre un débit maximal.

bfq est basé sur le code **cfq**. Il n'accorde pas le disque à chaque processus pour une tranche de temps fixe, mais attribue au processus une adresse *budget* mesurée en nombre de secteurs.

Ce planificateur convient à la copie de fichiers volumineux et le système ne devient pas insensible dans ce cas.

kyber

L'ordonnanceur s'adapte pour atteindre un objectif de latence en calculant les latences de chaque demande d'E/S soumise à la couche d'E/S par bloc. Vous pouvez configurer les temps de latence cibles pour les demandes de lecture, en cas d'oubli de cache, et d'écriture synchrone.
Cet ordonnanceur est adapté aux périphériques rapides, par exemple NVMe, SSD, ou d'autres périphériques à faible latence.

11.2. DIFFÉRENTS ORDONNANCEURS DE DISQUES POUR DIFFÉRENTS CAS D'UTILISATION

En fonction des tâches effectuées par votre système, les ordonnanceurs de disques suivants sont recommandés comme base de référence avant toute analyse et tout réglage :

Tableau 11.1. Ordonnanceurs de disques pour différents cas d'utilisation

Use case	Planificateur de disque
Disque dur traditionnel avec interface SCSI	Utilisez **mq-deadline** ou **bfq**.
SSD haute performance ou système lié à l'unité centrale avec stockage rapide	Utilisez **none**, surtout si vous utilisez des applications d'entreprise. Vous pouvez également utiliser **kyber**.
Tâches de bureau ou interactives	Utiliser **bfq**.
Invité virtuel	Utilisez **mq-deadline**. Avec un pilote d'adaptateur de bus hôte (HBA) capable de gérer plusieurs files d'attente, utilisez **none**.

11.3. LE PLANIFICATEUR DE DISQUE PAR DÉFAUT

Les périphériques en mode bloc utilisent le planificateur de disque par défaut, à moins que vous n'en spécifiiez un autre.

NOTE

Pour les périphériques de bloc **non-volatile Memory Express (NVMe)** en particulier, l'ordonnanceur par défaut est **none** et Red Hat recommande de ne pas le modifier.

Le noyau sélectionne un planificateur de disque par défaut en fonction du type de périphérique. Le planificateur sélectionné automatiquement est généralement le paramètre optimal. Si vous avez besoin d'un planificateur différent, Red Hat vous recommande d'utiliser les règles **udev** ou l'application **TuneD** pour le configurer. Faites correspondre les périphériques sélectionnés et changez l'ordonnanceur uniquement pour ces périphériques.

11.4. DÉTERMINATION DE L'ORDONNANCEUR DE DISQUE ACTIF

Cette procédure permet de déterminer quel planificateur de disque est actuellement actif sur une unité de bloc donnée.

Procédure

- Lire le contenu du **/sys/block/*device*/queue/scheduler** fichier :

```
# cat /sys/block/device/queue/scheduler

[mq-deadline] kyber bfq none
```

  Dans le nom du fichier, remplacez *device* par le nom du bloc, par exemple **sdc**.

  Le planificateur actif est indiqué entre crochets (**[ ]**).

11.5. PARAMÉTRAGE DU PLANIFICATEUR DE DISQUE À L'AIDE DE TUNED

Cette procédure permet de créer et d'activer un profil **TuneD** qui définit un planificateur de disque donné pour les périphériques de bloc sélectionnés. Le paramètre persiste lors des redémarrages du système.

Dans les commandes et la configuration suivantes, remplacer :

- *device* avec le nom du dispositif de blocage, par exemple **sdf**
- *selected-scheduler* avec le planificateur de disque que vous souhaitez définir pour le périphérique, par exemple **bfq**

Conditions préalables

- Le service **TuneD** est installé et activé. Pour plus de détails, voir Installation et activation de TuneD.

Procédure

1. Facultatif : Sélectionnez un profil **TuneD** existant sur lequel votre profil sera basé. Pour obtenir une liste des profils disponibles, voir les profils TuneD distribués avec RHEL .
   Pour savoir quel profil est actuellement actif, utilisez :

```
$ tuned-adm active
```

2. Créez un nouveau répertoire qui contiendra votre profil **TuneD**:

```
# mkdir /etc/tuned/my-profile
```

3. Recherchez l'identifiant unique du système du bloc sélectionné :

```
$ udevadm info --query=property --name=/dev/device | grep -E '(WWN|SERIAL)'

ID_WWN=0x5002538d00000000_
```

```
ID_SERIAL=Generic-_SD_MMC_20120501030900000-0:0
ID_SERIAL_SHORT=20120501030900000
```

NOTE

La commande de cet exemple renverra toutes les valeurs identifiées par un World Wide Name (WWN) ou un numéro de série associé au dispositif de bloc spécifié. Bien qu'il soit préférable d'utiliser un WWN, celui-ci n'est pas toujours disponible pour un dispositif donné et toutes les valeurs renvoyées par la commande de l'exemple peuvent être utilisées comme *device system unique ID*.

4. Créer le fichier de **/etc/tuned/*my-profile*/tuned.conf** fichier de configuration. Dans le fichier, définissez les options suivantes :

   a. Facultatif : Inclure un profil existant :

   ```
   [main]
   include=existing-profile
   ```

   b. Définir le planificateur de disque sélectionné pour le périphérique qui correspond à l'identifiant WWN :

   ```
   [disk]
   devices_udev_regex=IDNAME=device system unique id
   elevator=selected-scheduler
   ```

   Ici :

   - Remplacer *IDNAME* par le nom de l'identifiant utilisé (par exemple, **ID_WWN**).
   - Remplacer *device system unique id* par la valeur de l'identifiant choisi (par exemple, **0x5002538d00000000**).
     Pour faire correspondre plusieurs appareils dans l'option **devices_udev_regex**, mettez les identifiants entre parenthèses et séparez-les par des barres verticales :

     ```
     devices_udev_regex=(ID_WWN=0x5002538d00000000)|
     (ID_WWN=0x1234567800000000)
     ```

5. Activez votre profil :

   ```
   # tuned-adm profile my-profile
   ```

Verification steps

1. Vérifiez que le profil TuneD est actif et appliqué :

   ```
   $ tuned-adm active

   Current active profile: my-profile
   ```

   ```
   $ tuned-adm verify

   Verification succeeded, current system settings match the preset profile.
   ```

```
See TuneD log file ('/var/log/tuned/tuned.log') for details.
```

2. Lire le contenu du **/sys/block/*device*/queue/scheduler** fichier :

```
# cat /sys/block/device/queue/scheduler

[mq-deadline] kyber bfq none
```

Dans le nom du fichier, remplacez *device* par le nom du bloc, par exemple **sdc**.

Le planificateur actif est indiqué entre crochets (**[]**).

Ressources supplémentaires

- Personnalisation des profils TuneD.

11.6. DÉFINITION DE L'ORDONNANCEUR DE DISQUE À L'AIDE DES RÈGLES UDEV

Cette procédure permet de définir un planificateur de disque donné pour des périphériques de bloc spécifiques à l'aide des règles **udev**. Le paramètre persiste lors des redémarrages du système.

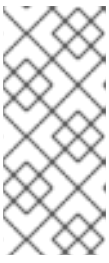
Dans les commandes et la configuration suivantes, remplacer :

- *device* avec le nom du dispositif de blocage, par exemple **sdf**
- *selected-scheduler* avec le planificateur de disque que vous souhaitez définir pour le périphérique, par exemple **bfq**

Procédure

1. Recherchez l'identifiant unique du système du dispositif de blocage :

```
$ udevadm info --name=/dev/device | grep -E '(WWN|SERIAL)'
E: ID_WWN=0x5002538d00000000
E: ID_SERIAL=Generic-_SD_MMC_20120501030900000-0:0
E: ID_SERIAL_SHORT=20120501030900000
```



NOTE

La commande de cet exemple renverra toutes les valeurs identifiées par un World Wide Name (WWN) ou un numéro de série associé au dispositif de bloc spécifié. Bien qu'il soit préférable d'utiliser un WWN, celui-ci n'est pas toujours disponible pour un dispositif donné et toutes les valeurs renvoyées par la commande de l'exemple peuvent être utilisées comme *device system unique ID*.

2. Configurez la règle **udev**. Créez le fichier **/etc/udev/rules.d/99-scheduler.rules** avec le contenu suivant :

```
ACTION=="ajouter/modifier", SUBSYSTEM=="bloquer", ENV{IDNAME}=="device system unique id", ATTR{queue/scheduler}="selected-scheduler"
```

Ici :

- Remplacer *IDNAME* par le nom de l'identifiant utilisé (par exemple, **ID_WWN**).
- Remplacer *device system unique id* par la valeur de l'identifiant choisi (par exemple, **0x5002538d00000000**).

3. Recharger **udev** règles :

```
# udevadm control --reload-rules
```

4. Appliquer la configuration de l'ordonnanceur :

```
# udevadm trigger --type=devices --action=change
```

Verification steps

- Vérifier l'ordonnanceur actif :

```
# cat /sys/block/device/queue/scheduler
```

11.7. DÉFINITION TEMPORAIRE D'UN PLANIFICATEUR POUR UN DISQUE SPÉCIFIQUE

Cette procédure permet de définir un planificateur de disque donné pour des périphériques de bloc spécifiques. Le paramètre ne persiste pas lors des redémarrages du système.

Procédure

- Inscrivez le nom de l'ordonnanceur sélectionné dans le fichier **/sys/block/*device*/queue/scheduler** dans le fichier

```
# echo selected-scheduler > /sys/block/device/queue/scheduler
```

Dans le nom du fichier, remplacez *device* par le nom du bloc, par exemple **sdc**.

Verification steps

- Vérifiez que l'ordonnanceur est actif sur l'appareil :

```
# cat /sys/block/device/queue/scheduler
```

CHAPITRE 12. OPTIMISER LES PERFORMANCES D'UN SERVEUR SAMBA

Découvrez quels paramètres peuvent améliorer les performances de Samba dans certaines situations, et quels paramètres peuvent avoir un impact négatif sur les performances.

Certaines parties de cette section ont été adoptées à partir de la documentation Performance Tuning publiée dans le Samba Wiki. Licence : CC BY 4.0. Auteurs et contributeurs : Voir l'onglet historique de la page Wiki.

Conditions préalables

- Samba est configuré comme un serveur de fichiers ou d'impression

12.1. RÉGLAGE DE LA VERSION DU PROTOCOLE SMB

Chaque nouvelle version de SMB ajoute des fonctionnalités et améliore les performances du protocole. Les systèmes d'exploitation récents Windows et Windows Server prennent toujours en charge la dernière version du protocole. Si Samba utilise également la dernière version du protocole, les clients Windows qui se connectent à Samba bénéficient des améliorations de performance. Dans Samba, la valeur par défaut du protocole max du serveur est définie sur la dernière version stable du protocole SMB prise en charge.

NOTE

Pour que la dernière version stable du protocole SMB soit toujours activée, ne définissez pas le paramètre **server max protocol**. Si vous définissez ce paramètre manuellement, vous devrez le modifier à chaque nouvelle version du protocole SMB pour que la dernière version du protocole soit activée.

La procédure suivante explique comment utiliser la valeur par défaut du paramètre **server max protocol**.

Procédure

1. Supprimer le paramètre **server max protocol** de la section **[global]** du fichier **/etc/samba/smb.conf**.
2. Recharger la configuration de Samba

```
# smbcontrol all reload-config
```

12.2. OPTIMISATION DES PARTAGES AVEC DES RÉPERTOIRES CONTENANT UN GRAND NOMBRE DE FICHIERS

Linux prend en charge les noms de fichiers sensibles à la casse. C'est pourquoi Samba doit rechercher les noms de fichiers en majuscules et en minuscules dans les répertoires lors de la recherche ou de l'accès à un fichier. Vous pouvez configurer un partage pour qu'il crée de nouveaux fichiers uniquement en minuscules ou en majuscules, ce qui améliore les performances.

Conditions préalables

- Samba est configuré comme serveur de fichiers

Procédure

1. Renommer tous les fichiers du partage en minuscules.

NOTE

En utilisant les réglages de cette procédure, les fichiers dont les noms ne sont pas en minuscules ne seront plus affichés.

2. Définissez les paramètres suivants dans la section du partage :

```
case sensitive = true
default case = lower
preserve case = no
short preserve case = no
```

Pour plus de détails sur les paramètres, voir leur description dans la page de manuel **smb.conf(5)**.

3. Vérifiez le fichier **/etc/samba/smb.conf**:

```
# testparm
```

4. Recharger la configuration de Samba :

```
# smbcontrol all reload-config
```

Après avoir appliqué ces paramètres, les noms de tous les fichiers nouvellement créés sur ce partage utilisent des minuscules. Grâce à ces paramètres, Samba n'a plus besoin de rechercher les majuscules et les minuscules dans le répertoire, ce qui améliore les performances.

12.3. PARAMÈTRES POUVANT AVOIR UN IMPACT NÉGATIF SUR LES PERFORMANCES

Par défaut, le noyau de Red Hat Enterprise Linux est réglé pour des performances réseau élevées. Par exemple, le noyau utilise un mécanisme de réglage automatique pour la taille des tampons. La définition du paramètre **socket options** dans le fichier **/etc/samba/smb.conf** remplace ces paramètres du noyau. Par conséquent, la définition de ce paramètre diminue les performances du réseau Samba dans la plupart des cas.

Pour utiliser les paramètres optimisés du noyau, supprimez le paramètre **socket options** de la section **[global]** du site **/etc/samba/smb.conf**.

CHAPITRE 13. OPTIMIZING VIRTUAL MACHINE PERFORMANCE

Virtual machines (VMs) always experience some degree of performance deterioration in comparison to the host. The following sections explain the reasons for this deterioration and provide instructions on how to minimize the performance impact of virtualization in RHEL 9, so that your hardware infrastructure resources can be used as efficiently as possible.

13.1. WHAT INFLUENCES VIRTUAL MACHINE PERFORMANCE

VMs are run as user-space processes on the host. The hypervisor therefore needs to convert the host's system resources so that the VMs can use them. As a consequence, a portion of the resources is consumed by the conversion, and the VM therefore cannot achieve the same performance efficiency as the host.

The impact of virtualization on system performance

More specific reasons for VM performance loss include:

- Virtual CPUs (vCPUs) are implemented as threads on the host, handled by the Linux scheduler.
- VMs do not automatically inherit optimization features, such as NUMA or huge pages, from the host kernel.
- Disk and network I/O settings of the host might have a significant performance impact on the VM.
- Network traffic typically travels to a VM through a software-based bridge.
- Depending on the host devices and their models, there might be significant overhead due to emulation of particular hardware.

The severity of the virtualization impact on the VM performance is influenced by a variety factors, which include:

- The number of concurrently running VMs.
- The amount of virtual devices used by each VM.
- The device types used by the VMs.

Reducing VM performance loss

RHEL 9 provides a number of features you can use to reduce the negative performance effects of virtualization. Notably:

- The **TuneD** service can automatically optimize the resource distribution and performance of your VMs.
- Block I/O tuning can improve the performances of the VM's block devices, such as disks.
- NUMA tuning can increase vCPU performance.
- Virtual networking can be optimized in various ways.

IMPORTANT

Tuning VM performance can have adverse effects on other virtualization functions. For example, it can make migrating the modified VM more difficult.

13.2. OPTIMIZING VIRTUAL MACHINE PERFORMANCE USING TUNED

The **TuneD** utility is a tuning profile delivery mechanism that adapts RHEL for certain workload characteristics, such as requirements for CPU-intensive tasks or storage-network throughput responsiveness. It provides a number of tuning profiles that are pre-configured to enhance performance and reduce power consumption in a number of specific use cases. You can edit these profiles or create new profiles to create performance solutions tailored to your environment, including virtualized environments.

To optimize RHEL 9 for virtualization, use the following profiles:

- For RHEL 9 virtual machines, use the **virtual-guest** profile. It is based on the generally applicable ***throughput-performance*** profile, but also decreases the swappiness of virtual memory.
- For RHEL 9 virtualization hosts, use the **virtual-host** profile. This enables more aggressive writeback of dirty memory pages, which benefits the host performance.

Conditions préalables

- The **TuneD** service is installed and enabled.

Procédure

To enable a specific **TuneD** profile:

1. List the available **TuneD** profiles.

```
# tuned-adm list

Available profiles:
- balanced             - General non-specialized TuneD profile
- desktop              - Optimize for the desktop use-case
[...]
- virtual-guest        - Optimize for running inside a virtual guest
- virtual-host         - Optimize for running KVM guests
Current active profile: balanced
```

2. **Optional:** Create a new **TuneD** profile or edit an existing **TuneD** profile.
   For more information, see Customizing TuneD profiles.

3. Activate a **TuneD** profile.

```
# tuned-adm profile selected-profile
```

   - To optimize a virtualization host, use the *virtual-host* profile.

```
# tuned-adm profile virtual-host
```

   - On a RHEL guest operating system, use the *virtual-guest* profile.

```
# tuned-adm profile virtual-guest
```

Ressources supplémentaires

- Monitoring and managing system status and performance

13.3. OPTIMIZING LIBVIRT DAEMONS

The **libvirt** virtualization suite works as a management layer for the RHEL hypervisor, and your **libvirt** configuration significantly impacts your virtualization host. Notably, RHEL 9 contains two different types of **libvirt** daemons, monolithic or modular, and which type of daemons you use affects how granularly you can configure individual virtualization drivers.

13.3.1. Types of libvirt daemons

RHEL 9 supports the following **libvirt** daemon types:

Monolithic libvirt

The traditional **libvirt** daemon, **libvirtd**, controls a wide variety of virtualization drivers, using a single configuration file - **/etc/libvirt/libvirtd.conf**.
As such, **libvirtd** allows for centralized hypervisor configuration, but may use system resources inefficiently. Therefore, **libvirtd** will become unsupported in a future major release of RHEL.

However, if you updated to RHEL 9 from RHEL 8, your host still uses **libvirtd** by default.

Modular libvirt

Newly introduced in RHEL 9, modular **libvirt** provides a specific daemon for each virtualization driver. These include the following:

- **virtqemud** - A primary daemon for hypervisor management
- **virtinterfaced** - A secondary daemon for host NIC management
- **virtnetworkd** - A secondary daemon for virtual network management
- **virtnodedevd** - A secondary daemon for host physical device management
- **virtnwfilterd** - A secondary daemon for host firewall management
- **virtsecretd** - A secondary daemon for host secret management
- **virtstoraged** - A secondary daemon for storage management

Each of the daemons has a separate configuration file - for example **/etc/libvirt/virtqemud.conf**. As such, modular **libvirt** daemons provide better options for fine-tuning **libvirt** resource management.

If you performed a fresh install of RHEL 9, modular **libvirt** is configured by default.

Prochaines étapes

- If your RHEL 9 uses **libvirtd**, Red Hat recommends switching to modular daemons. For instructions, see Enabling modular libvirt daemons.

13.3.2. Enabling modular libvirt daemons

In RHEL 9, the **libvirt** library uses modular daemons that handle individual virtualization driver sets on your host. For example, the **virtqemud** daemon handles QEMU drivers.

If you performed a fresh install of a RHEL 9 host, your hypervisor uses modular **libvirt** daemons by default. However, if you upgraded your host from RHEL 8 to RHEL 9, your hypervisor uses the monolithic **libvirtd** daemon, which is the default in RHEL 8.

If that is the case, Red Hat recommends enabling the modular **libvirt** daemons instead, because they provide better options for fine-tuning **libvirt** resource management. In addition, **libvirtd** will become unsupported in a future major release of RHEL.

Conditions préalables

- Your hypervisor is using the monolithic **libvirtd** service.

```
# systemctl is-active libvirtd.service
active
```

  If this command displays **active**, you are using **libvirtd**.

- Your virtual machines are shut down.

Procédure

1. Stop **libvirtd** and its sockets.

```
# systemctl stop libvirtd.service
# systemctl stop libvirtd{,-ro,-admin,-tcp,-tls}.socket
```

2. Disable **libvirtd** to prevent it from starting on boot.

```
$ systemctl disable libvirtd.service
$ systemctl disable libvirtd{,-ro,-admin,-tcp,-tls}.socket
```

3. Enable the modular **libvirt** daemons.

```
# for drv in qemu interface network nodedev nwfilter secret storage; do systemctl unmask virt${drv}d.service; systemctl unmask virt${drv}d{,-ro,-admin}.socket; systemctl enable virt${drv}d.service; systemctl enable virt${drv}d{,-ro,-admin}.socket; done
```

4. Start the sockets for the modular daemons.

```
# for drv in qemu network nodedev nwfilter secret storage; do systemctl start virt${drv}d{,-ro,-admin}.socket; done
```

5. **Optional:** If you require connecting to your host from remote hosts, enable and start the virtualization proxy daemon.

   a. Check whether the **libvirtd-tls.socket** service is enabled on your system.

```
# cat /etc/libvirt/libvirt.conf | grep listen_tls

listen_tls = 0
```

b. If **libvirtd-tls.socket** is not enabled (**listen_tls = 0**), activate **virtproxyd** as follows:

```
# systemctl unmask virtproxyd.service
# systemctl unmask virtproxyd{,-ro,-admin}.socket
# systemctl enable virtproxyd.service
# systemctl enable virtproxyd{,-ro,-admin}.socket
# systemctl start virtproxyd{,-ro,-admin}.socket
```

c. If **libvirtd-tls.socket** is enabled (**listen_tls = 1**), activate **virtproxyd** as follows:

```
# systemctl unmask virtproxyd.service
# systemctl unmask virtproxyd{,-ro,-admin,-tls}.socket
# systemctl enable virtproxyd.service
# systemctl enable virtproxyd{,-ro,-admin,-tls}.socket
# systemctl start virtproxyd{,-ro,-admin,-tls}.socket
```

To enable the TLS socket of **virtproxyd**, your host must have TLS certificates configured to work with **libvirt**. For more information, see the Upstream libvirt documentation.

Vérification

1. Activate the enabled virtualization daemons.

```
# virsh uri
qemu:///system
```

2. Verify that your host is using the **virtqemud** modular daemon.

```
# systemctl is-active virtqemud.service
active
```

If the status is **active**, you have successfully enabled modular **libvirt** daemons.

13.4. CONFIGURING VIRTUAL MACHINE MEMORY

To improve the performance of a virtual machine (VM), you can assign additional host RAM to the VM. Similarly, you can decrease the amount of memory allocated to a VM so the host memory can be allocated to other VMs or tasks.

To perform these actions, you can use the web console or the command-line interface.

13.4.1. Adding and removing virtual machine memory using the web console

To improve the performance of a virtual machine (VM) or to free up the host resources it is using, you can use the web console to adjust amount of memory allocated to the VM.

Conditions préalables

- The guest OS is running the memory balloon drivers. To verify this is the case:

1. Ensure the VM's configuration includes the **memballoon** device:

```
# virsh dumpxml testguest | grep memballoon
<memballoon model='virtio'>
  </memballoon>
```

If this commands displays any output and the model is not set to **none**, the **memballoon** device is present.

2. Ensure the balloon drivers are running in the guest OS.

   - Dans les invités Windows, les pilotes sont installés dans le cadre du paquet de pilotes **virtio-win**. Pour plus d'informations, voir Installation des pilotes KVM paravirtualisés pour les machines virtuelles Windows.
   - In Linux guests, the drivers are generally included by default and activate when the **memballoon** device is present.

- Le plug-in VM de la console web est installé sur votre système.

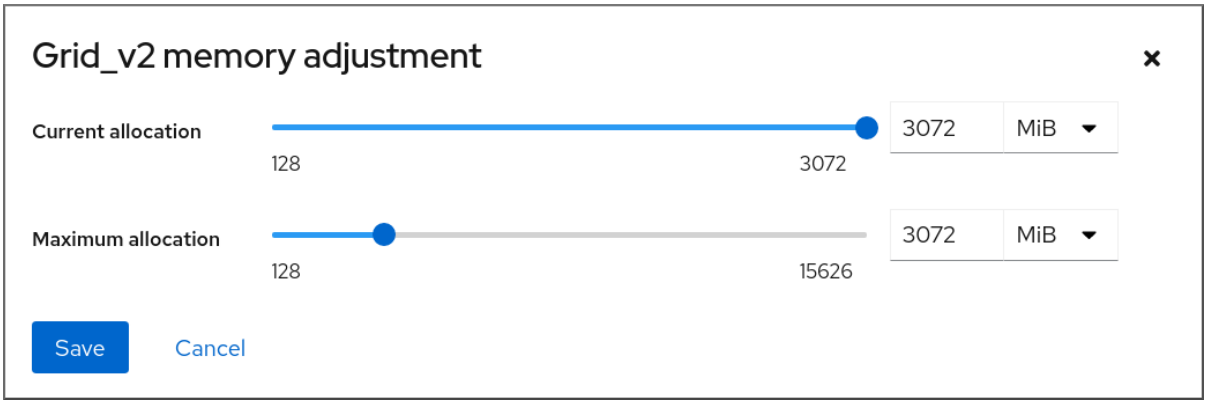
Procédure

1. **Optional:** Obtain the information about the maximum memory and currently used memory for a VM. This will serve as a baseline for your changes, and also for verification.

```
# virsh dominfo testguest
Max memory:     2097152 KiB
Used memory:    2097152 KiB
```

2. In the **Virtual Machines** interface, click the VM whose information you want to see.
A new page opens with an Overview section with basic information about the selected VM and a Console section to access the VM's graphical interface.

3. Click **edit** next to the **Memory** line in the Overview pane.
The **Memory Adjustment** dialog appears.



4. Configure the virtual CPUs for the selected VM.

   - **Maximum allocation** - Sets the maximum amount of host memory that the VM can use for its processes. You can specify the maximum memory when creating the VM or increase it later. You can specify memory as multiples of MiB or GiB.
   Adjusting maximum memory allocation is only possible on a shut-off VM.
   - **Current allocation** - Sets the actual amount of memory allocated to the VM. This value can

be less than the Maximum allocation but cannot exceed it. You can adjust the value to regulate the memory available to the VM for its processes. You can specify memory as multiples of MiB or GiB.
If you do not specify this value, the default allocation is the **Maximum allocation** value.

5. Cliquez sur **Enregistrer**.
The memory allocation of the VM is adjusted.

Ressources supplémentaires

- Ajouter et supprimer la mémoire d'une machine virtuelle à l'aide de l'interface de ligne de commande
- Optimiser les performances de l'unité centrale de la machine virtuelle

13.4.2. Adding and removing virtual machine memory using the command-line interface

To improve the performance of a virtual machine (VM) or to free up the host resources it is using, you can use the CLI to adjust amount of memory allocated to the VM.

Conditions préalables

- The guest OS is running the memory balloon drivers. To verify this is the case:
  1. Ensure the VM's configuration includes the **memballoon** device:

     ```
     # virsh dumpxml testguest | grep memballoon
     <memballoon model='virtio'>
       </memballoon>
     ```

     If this commands displays any output and the model is not set to **none**, the **memballoon** device is present.

  2. Ensure the ballon drivers are running in the guest OS.
     - Dans les invités Windows, les pilotes sont installés dans le cadre du paquet de pilotes **virtio-win**. Pour plus d'informations, voir Installation des pilotes KVM paravirtualisés pour les machines virtuelles Windows.
     - In Linux guests, the drivers are generally included by default and activate when the **memballoon** device is present.

Procédure

1. **Optional:** Obtain the information about the maximum memory and currently used memory for a VM. This will serve as a baseline for your changes, and also for verification.

   ```
   # virsh dominfo testguest
   Max memory:     2097152 KiB
   Used memory:    2097152 KiB
   ```

2. Adjust the maximum memory allocated to a VM. Increasing this value improves the performance potential of the VM, and reducing the value lowers the performance footprint the VM has on your host. Note that this change can only be performed on a shut-off VM, so adjusting a running

VM requires a reboot to take effect.
For example, to change the maximum memory that the *testguest* VM can use to 4096 MiB:

```
# virt-xml testguest --edit --memory memory=4096,currentMemory=4096
Domain 'testguest' defined successfully.
Changes will take effect after the domain is fully powered off.
```

Pour augmenter la mémoire maximale d'une VM en cours d'exécution, vous pouvez attacher un périphérique de mémoire à la VM. Cette opération est également appelée **memory hot plug**. Pour plus d'informations, voir Attacher des périphériques aux machines virtuelles.

AVERTISSEMENT

Removing memory devices from a running VM (also referred as a memory hot unplug) is not supported, and highly discouraged by Red Hat.

3. **Optional:** You can also adjust the memory currently used by the VM, up to the maximum allocation. This regulates the memory load that the VM has on the host until the next reboot, without changing the maximum VM allocation.

```
# virsh setmem testguest --current 2048
```

Vérification

1. Confirm that the memory used by the VM has been updated:

```
# virsh dominfo testguest
Max memory:     4194304 KiB
Used memory:    2097152 KiB
```

2. **Optional:** If you adjusted the current VM memory, you can obtain the memory balloon statistics of the VM to evaluate how effectively it regulates its memory use.

```
 # virsh domstats --balloon testguest
Domain: 'testguest'
  balloon.current=365624
  balloon.maximum=4194304
  balloon.swap_in=0
  balloon.swap_out=0
  balloon.major_fault=306
  balloon.minor_fault=156117
  balloon.unused=3834448
  balloon.available=4035008
  balloon.usable=3746340
  balloon.last-update=1587971682
  balloon.disk_caches=75444
  balloon.hugetlb_pgalloc=0
  balloon.hugetlb_pgfail=0
  balloon.rss=1005456
```

Ressources supplémentaires

- Ajouter et supprimer la mémoire d'une machine virtuelle à l'aide de la console web
- Optimiser les performances de l'unité centrale de la machine virtuelle

13.4.3. Ressources supplémentaires

- Attacher des périphériques aux machines virtuelles Attacher des périphériques aux machines virtuelles.

13.5. OPTIMIZING VIRTUAL MACHINE I/O PERFORMANCE

The input and output (I/O) capabilities of a virtual machine (VM) can significantly limit the VM's overall efficiency. To address this, you can optimize a VM's I/O by configuring block I/O parameters.

13.5.1. Tuning block I/O in virtual machines

When multiple block devices are being used by one or more VMs, it might be important to adjust the I/O priority of specific virtual devices by modifying their *I/O weights*.

Increasing the I/O weight of a device increases its priority for I/O bandwidth, and therefore provides it with more host resources. Similarly, reducing a device's weight makes it consume less host resources.

NOTE

Each device's **weight** value must be within the **100** to **1000** range. Alternatively, the value can be **0**, which removes that device from per-device listings.

Procédure

To display and set a VM's block I/O parameters:

1. Display the current **<blkio>** parameters for a VM:
   # virsh dumpxml *VM-name*

```
<domain>
 [...]
 <blkiotune>
  <weight>800</weight>
  <device>
   <path>/dev/sda</path>
   <weight>1000</weight>
  </device>
  <device>
   <path>/dev/sdb</path>
   <weight>500</weight>
  </device>
 </blkiotune>
 [...]
</domain>
```

2. Edit the I/O weight of a specified device:

```
# virsh blkiotune VM-name --device-weights device, I/O-weight
```

For example, the following changes the weight of the */dev/sda* device in the *liftrul* VM to 500.

```
# virsh blkiotune liftbrul --device-weights /dev/sda, 500
```

13.5.2. Disk I/O throttling in virtual machines

When several VMs are running simultaneously, they can interfere with system performance by using excessive disk I/O. Disk I/O throttling in KVM virtualization provides the ability to set a limit on disk I/O requests sent from the VMs to the host machine. This can prevent a VM from over-utilizing shared resources and impacting the performance of other VMs.

To enable disk I/O throttling, set a limit on disk I/O requests sent from each block device attached to VMs to the host machine.

Procédure

1. Use the **virsh domblklist** command to list the names of all the disk devices on a specified VM.

```
# virsh domblklist rollin-coal
Target     Source
------------------------------------------------
vda        /var/lib/libvirt/images/rollin-coal.qcow2
sda        -
sdb        /home/horridly-demanding-processes.iso
```

2. Find the host block device where the virtual disk that you want to throttle is mounted.
   For example, if you want to throttle the **sdb** virtual disk from the previous step, the following output shows that the disk is mounted on the **/dev/nvme0n1p3** partition.

```
$ lsblk
NAME                                          MAJ:MIN RM   SIZE RO TYPE  MOUNTPOINT
zram0                                         252:0    0     4G  0 disk  [SWAP]
nvme0n1                                         259:0    0 238.5G  0 disk
├─nvme0n1p1                                     259:1    0   600M  0 part  /boot/efi
├─nvme0n1p2                                     259:2    0     1G  0 part  /boot
└─nvme0n1p3                                     259:3    0 236.9G  0 part
  └─luks-a1123911-6f37-463c-b4eb-fxzy1ac12fea 253:0    0 236.9G  0 crypt /home
```

3. Set I/O limits for the block device using the **virsh blkiotune** command.

```
# virsh blkiotune VM-name --parameter device,limit
```

   The following example throttles the **sdb** disk on the **rollin-coal** VM to 1000 read and write I/O operations per second and to 50 MB per second read and write throughput.

```
# virsh blkiotune rollin-coal --device-read-iops-sec /dev/nvme0n1p3,1000 --device-write-iops-sec /dev/nvme0n1p3,1000 --device-write-bytes-sec /dev/nvme0n1p3,52428800 --device-read-bytes-sec /dev/nvme0n1p3,52428800
```

Informations complémentaires

- Disk I/O throttling can be useful in various situations, for example when VMs belonging to different customers are running on the same host, or when quality of service guarantees are given for different VMs. Disk I/O throttling can also be used to simulate slower disks.
- I/O throttling can be applied independently to each block device attached to a VM and supports limits on throughput and I/O operations.
- Red Hat ne prend pas en charge l'utilisation de la commande **virsh blkdeviotune** pour configurer la limitation des E/S dans les machines virtuelles. Pour plus d'informations sur les fonctionnalités non prises en charge lors de l'utilisation de RHEL 9 en tant qu'hôte de VM, voir Fonctionnalités non prises en charge dans la virtualisation RHEL 9 .

13.5.3. Enabling multi-queue virtio-scsi

When using **virtio-scsi** storage devices in your virtual machines (VMs), the *multi-queue virtio-scsi* feature provides improved storage performance and scalability. It enables each virtual CPU (vCPU) to have a separate queue and interrupt to use without affecting other vCPUs.

Procédure

- To enable multi-queue virtio-scsi support for a specific VM, add the following to the VM's XML configuration, where *N* is the total number of vCPU queues:

```
<controller type='scsi' index='0' model='virtio-scsi'>
  <driver queues='N' />
</controller>
```

13.6. OPTIMIZING VIRTUAL MACHINE CPU PERFORMANCE

Much like physical CPUs in host machines, vCPUs are critical to virtual machine (VM) performance. As a result, optimizing vCPUs can have a significant impact on the resource efficiency of your VMs. To optimize your vCPU:

1. Adjust how many host CPUs are assigned to the VM. You can do this using the CLI or the web console.
2. Ensure that the vCPU model is aligned with the CPU model of the host. For example, to set the *testguest1* VM to use the CPU model of the host:

```
# virt-xml testguest1 --edit --cpu host-model
```

3. Manage kernel same-page merging (KSM) .
4. If your host machine uses Non-Uniform Memory Access (NUMA), you can also **configure NUMA** for its VMs. This maps the host's CPU and memory processes onto the CPU and memory processes of the VM as closely as possible. In effect, NUMA tuning provides the vCPU with a more streamlined access to the system memory allocated to the VM, which can improve the vCPU processing effectiveness.
   For details, see Configuring NUMA in a virtual machine and Sample vCPU performance tuning scenario.

13.6.1. Adding and removing virtual CPUs using the command-line interface

To increase or optimize the CPU performance of a virtual machine (VM), you can add or remove virtual CPUs (vCPUs) assigned to the VM.

When performed on a running VM, this is also referred to as vCPU hot plugging and hot unplugging. However, note that vCPU hot unplug is not supported in RHEL 9, and Red Hat highly discourages its use.

Conditions préalables

- **Optional:** View the current state of the vCPUs in the targeted VM. For example, to display the number of vCPUs on the *testguest* VM:

```
# virsh vcpucount testguest
maximum      config         4
maximum      live           2
current      config         2
current      live           1
```

This output indicates that *testguest* is currently using 1 vCPU, and 1 more vCPu can be hot plugged to it to increase the VM's performance. However, after reboot, the number of vCPUs *testguest* uses will change to 2, and it will be possible to hot plug 2 more vCPUs.

Procédure

1. Adjust the maximum number of vCPUs that can be attached to a VM, which takes effect on the VM's next boot.
   For example, to increase the maximum vCPU count for the *testguest* VM to 8:

```
# virsh setvcpus testguest 8 --maximum --config
```

   Note that the maximum may be limited by the CPU topology, host hardware, the hypervisor, and other factors.

2. Adjust the current number of vCPUs attached to a VM, up to the maximum configured in the previous step. For example:

   - To increase the number of vCPUs attached to the running *testguest* VM to 4:

```
# virsh setvcpus testguest 4 --live
```

     This increases the VM's performance and host load footprint of *testguest* until the VM's next boot.

   - To permanently decrease the number of vCPUs attached to the *testguest* VM to 1:

```
# virsh setvcpus testguest 1 --config
```

     This decreases the VM's performance and host load footprint of *testguest* after the VM's next boot. However, if needed, additional vCPUs can be hot plugged to the VM to temporarily increase its performance.

Vérification

- Confirm that the current state of vCPU for the VM reflects your changes.

```
# virsh vcpucount testguest
```

```
maximum     config      8
maximum     live        4
current     config      1
current     live        4
```

Ressources supplémentaires

- Gérer les CPU virtuels à l'aide de la console web

13.6.2. Managing virtual CPUs using the web console

Using the RHEL 9 web console, you can review and configure virtual CPUs used by virtual machines (VMs) to which the web console is connected.

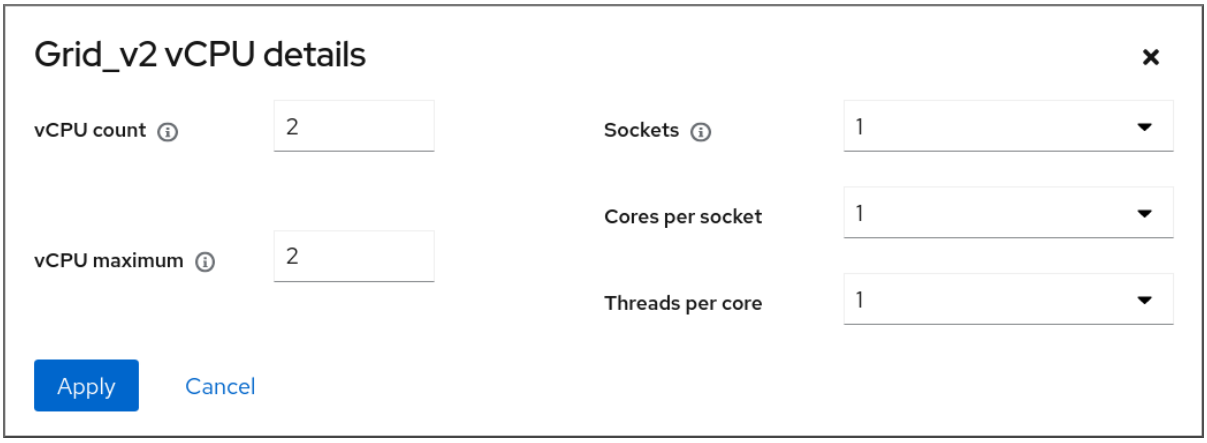
Conditions préalables

- Le plug-in VM de la console web est installé sur votre système.

Procédure

1. In the **Virtual Machines** interface, click the VM whose information you want to see.
   A new page opens with an Overview section with basic information about the selected VM and a Console section to access the VM's graphical interface.

2. Click **edit** next to the number of vCPUs in the Overview pane.
   The vCPU details dialog appears.



1. Configure the virtual CPUs for the selected VM.

   - **vCPU Count** - The number of vCPUs currently in use.

     NOTE

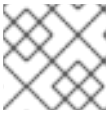
     The vCPU count cannot be greater than the vCPU Maximum.

   - **vCPU Maximum** - The maximum number of virtual CPUs that can be configured for the VM. If this value is higher than the **vCPU Count**, additional vCPUs can be attached to the VM.

   - **Sockets** - The number of sockets to expose to the VM.

- **Cores per socket**- The number of cores for each socket to expose to the VM.
- **Threads per core** - The number of threads for each core to expose to the VM.
  Note that the **Sockets**, **Cores per socket**, and **Threads per core** options adjust the CPU topology of the VM. This may be beneficial for vCPU performance and may impact the functionality of certain software in the guest OS. If a different setting is not required by your deployment, keep the default values.

2. Cliquez sur **Appliquer**.
   The virtual CPUs for the VM are configured.



NOTE

Changes to virtual CPU settings only take effect after the VM is restarted.

Ressources supplémentaires

- Ajout et suppression de CPU virtuels à l'aide de l'interface de ligne de commande

13.6.3. Configuring NUMA in a virtual machine

The following methods can be used to configure Non-Uniform Memory Access (NUMA) settings of a virtual machine (VM) on a RHEL 9 host.

Conditions préalables

- The host is a NUMA-compatible machine. To detect whether this is the case, use the **virsh nodeinfo** command and see the **NUMA cell(s)** line:

```
# virsh nodeinfo
CPU model:           x86_64
CPU(s):              48
CPU frequency:       1200 MHz
CPU socket(s):       1
Core(s) per socket:  12
Thread(s) per core:  2
NUMA cell(s):        2
Memory size:         67012964 KiB
```

  If the value of the line is 2 or greater, the host is NUMA-compatible.

Procédure

For ease of use, you can set up a VM's NUMA configuration using automated utilities and services. However, manual NUMA setup is more likely to yield a significant performance improvement.

Automatic methods

- Set the VM's NUMA policy to **Preferred**. For example, to do so for the *testguest5* VM:

```
# virt-xml testguest5 --edit --vcpus placement=auto
# virt-xml testguest5 --edit --numatune mode=preferred
```

- Enable automatic NUMA balancing on the host:

```
# echo 1 > /proc/sys/kernel/numa_balancing
```

- Use the **numad** command to automatically align the VM CPU with memory resources.

```
# numad
```

Manual methods

1. Pin specific vCPU threads to a specific host CPU or range of CPUs. This is also possible on non-NUMA hosts and VMs, and is recommended as a safe method of vCPU performance improvement.
   For example, the following commands pin vCPU threads 0 to 5 of the *testguest6* VM to host CPUs 1, 3, 5, 7, 9, and 11, respectively:

```
# virsh vcpupin testguest6 0 1
# virsh vcpupin testguest6 1 3
# virsh vcpupin testguest6 2 5
# virsh vcpupin testguest6 3 7
# virsh vcpupin testguest6 4 9
# virsh vcpupin testguest6 5 11
```

   Afterwards, you can verify whether this was successful:

```
# virsh vcpupin testguest6
VCPU   CPU Affinity
----------------------
0      1
1      3
2      5
3      7
4      9
5      11
```

2. After pinning vCPU threads, you can also pin QEMU process threads associated with a specified VM to a specific host CPU or range of CPUs. For example, the following commands pin the QEMU process thread of *testguest6* to CPUs 13 and 15, and verify this was successful:

```
# virsh emulatorpin testguest6 13,15
# virsh emulatorpin testguest6
emulator: CPU Affinity
----------------------------------
       *: 13,15
```

3. Finally, you can also specify which host NUMA nodes will be assigned specifically to a certain VM. This can improve the host memory usage by the VM's vCPU. For example, the following commands set *testguest6* to use host NUMA nodes 3 to 5, and verify this was successful:

```
# virsh numatune testguest6 --nodeset 3-5
# virsh numatune testguest6
```

NOTE

For best performance results, it is recommended to use all of the manual tuning methods listed above

Problèmes connus

- Il n'est actuellement pas possible d'effectuer des réglages NUMA sur les hôtes IBM Z .

Ressources supplémentaires

- Exemple de scénario d'optimisation des performances des vCPU
- Affichez la configuration NUMA actuelle de votre système à l'aide de l'utilitaire **numastat**

13.6.4. Sample vCPU performance tuning scenario

To obtain the best vCPU performance possible, Red Hat recommends using manual **vcpupin**, **emulatorpin**, and **numatune** settings together, for example like in the following scenario.

Starting scenario

- Your host has the following hardware specifics:
  - 2 NUMA nodes
  - 3 CPU cores on each node
  - 2 threads on each core

  The output of **virsh nodeinfo** of such a machine would look similar to:

```
# virsh nodeinfo
CPU model:           x86_64
CPU(s):              12
CPU frequency:       3661 MHz
CPU socket(s):       2
Core(s) per socket:  3
Thread(s) per core:  2
NUMA cell(s):        2
Memory size:         31248692 KiB
```

- You intend to modify an existing VM to have 8 vCPUs, which means that it will not fit in a single NUMA node.
  Therefore, you should distribute 4 vCPUs on each NUMA node and make the vCPU topology resemble the host topology as closely as possible. This means that vCPUs that run as sibling threads of a given physical CPU should be pinned to host threads on the same core. For details, see the *Solution* below:

Solution

1. Obtain the information on the host topology:

```
# virsh capabilities
```

The output should include a section that looks similar to the following:

```
<topology>
 <cells num="2">
  <cell id="0">
   <memory unit="KiB">15624346</memory>
   <pages unit="KiB" size="4">3906086</pages>
   <pages unit="KiB" size="2048">0</pages>
   <pages unit="KiB" size="1048576">0</pages>
   <distances>
    <sibling id="0" value="10" />
    <sibling id="1" value="21" />
   </distances>
   <cpus num="6">
    <cpu id="0" socket_id="0" core_id="0" siblings="0,3" />
    <cpu id="1" socket_id="0" core_id="1" siblings="1,4" />
    <cpu id="2" socket_id="0" core_id="2" siblings="2,5" />
    <cpu id="3" socket_id="0" core_id="0" siblings="0,3" />
    <cpu id="4" socket_id="0" core_id="1" siblings="1,4" />
    <cpu id="5" socket_id="0" core_id="2" siblings="2,5" />
   </cpus>
  </cell>
  <cell id="1">
   <memory unit="KiB">15624346</memory>
   <pages unit="KiB" size="4">3906086</pages>
   <pages unit="KiB" size="2048">0</pages>
   <pages unit="KiB" size="1048576">0</pages>
   <distances>
    <sibling id="0" value="21" />
    <sibling id="1" value="10" />
   </distances>
   <cpus num="6">
    <cpu id="6" socket_id="1" core_id="3" siblings="6,9" />
    <cpu id="7" socket_id="1" core_id="4" siblings="7,10" />
    <cpu id="8" socket_id="1" core_id="5" siblings="8,11" />
    <cpu id="9" socket_id="1" core_id="3" siblings="6,9" />
    <cpu id="10" socket_id="1" core_id="4" siblings="7,10" />
    <cpu id="11" socket_id="1" core_id="5" siblings="8,11" />
   </cpus>
  </cell>
 </cells>
</topology>
```

2. **Optional:** Test the performance of the VM using the applicable tools and utilities.

3. Set up and mount 1 GiB huge pages on the host:

   a. Add the following line to the host's kernel command line:

   ```
   default_hugepagesz=1G hugepagesz=1G
   ```

   b. Create the **/etc/systemd/system/hugetlb-gigantic-pages.service** file with the following content:

   ```
   [Unit]
   ```

```
Description=HugeTLB Gigantic Pages Reservation
DefaultDependencies=no
Before=dev-hugepages.mount
ConditionPathExists=/sys/devices/system/node
ConditionKernelCommandLine=hugepagesz=1G

[Service]
Type=oneshot
RemainAfterExit=yes
ExecStart=/etc/systemd/hugetlb-reserve-pages.sh

[Install]
WantedBy=sysinit.target
```

c. Create the **/etc/systemd/hugetlb-reserve-pages.sh** file with the following content:

```
#!/bin/sh

nodes_path=/sys/devices/system/node/
if [ ! -d $nodes_path ]; then
 echo "ERROR: $nodes_path does not exist"
 exit 1
fi

reserve_pages()
{
 echo $1 > $nodes_path/$2/hugepages/hugepages-1048576kB/nr_hugepages
}

reserve_pages 4 node1
reserve_pages 4 node2
```

This reserves four 1GiB huge pages from *node1* and four 1GiB huge pages from *node2*.

d. Make the script created in the previous step executable:

```
# chmod +x /etc/systemd/hugetlb-reserve-pages.sh
```

e. Enable huge page reservation on boot:

```
# systemctl enable hugetlb-gigantic-pages
```

4. Use the **virsh edit** command to edit the XML configuration of the VM you wish to optimize, in this example *super-VM*:

```
# virsh edit super-vm
```

5. Adjust the XML configuration of the VM in the following way:

a. Set the VM to use 8 static vCPUs. Use the **<vcpu/>** element to do this.

b. Pin each of the vCPU threads to the corresponding host CPU threads that it mirrors in the topology. To do so, use the **<vcpupin/>** elements in the **<cputune>** section.
Note that, as shown by the **virsh capabilities** utility above, host CPU threads are not ordered sequentially in their respective cores. In addition, the vCPU threads should be

pinned to the highest available set of host cores on the same NUMA node. For a table illustration, see the **Sample topology** section below.

The XML configuration for steps a. and b. can look similar to:

```
<cputune>
  <vcpupin vcpu='0' cpuset='1'/>
  <vcpupin vcpu='1' cpuset='4'/>
  <vcpupin vcpu='2' cpuset='2'/>
  <vcpupin vcpu='3' cpuset='5'/>
  <vcpupin vcpu='4' cpuset='7'/>
  <vcpupin vcpu='5' cpuset='10'/>
  <vcpupin vcpu='6' cpuset='8'/>
  <vcpupin vcpu='7' cpuset='11'/>
  <emulatorpin cpuset='6,9'/>
</cputune>
```

c. Set the VM to use 1 GiB huge pages:

```
<memoryBacking>
  <hugepages>
    <page size='1' unit='GiB'/>
  </hugepages>
</memoryBacking>
```

d. Configure the VM's NUMA nodes to use memory from the corresponding NUMA nodes on the host. To do so, use the **<memnode/>** elements in the **<numatune/>** section:

```
<numatune>
  <memory mode="preferred" nodeset="1"/>
  <memnode cellid="0" mode="strict" nodeset="0"/>
  <memnode cellid="1" mode="strict" nodeset="1"/>
</numatune>
```

e. Ensure the CPU mode is set to **host-passthrough**, and that the CPU uses cache in **passthrough** mode:

```
<cpu mode="host-passthrough">
  <topology sockets="2" cores="2" threads="2"/>
  <cache mode="passthrough"/>
```

Vérification

1. Confirm that the resulting XML configuration of the VM includes a section similar to the following:

```
[...]
 <memoryBacking>
   <hugepages>
     <page size='1' unit='GiB'/>
   </hugepages>
 </memoryBacking>
 <vcpu placement='static'>8</vcpu>
 <cputune>
```

```
  <vcpupin vcpu='0' cpuset='1'/>
  <vcpupin vcpu='1' cpuset='4'/>
  <vcpupin vcpu='2' cpuset='2'/>
  <vcpupin vcpu='3' cpuset='5'/>
  <vcpupin vcpu='4' cpuset='7'/>
  <vcpupin vcpu='5' cpuset='10'/>
  <vcpupin vcpu='6' cpuset='8'/>
  <vcpupin vcpu='7' cpuset='11'/>
  <emulatorpin cpuset='6,9'/>
 </cputune>
 <numatune>
  <memory mode="preferred" nodeset="1"/>
  <memnode cellid="0" mode="strict" nodeset="0"/>
  <memnode cellid="1" mode="strict" nodeset="1"/>
 </numatune>
 <cpu mode="host-passthrough">
  <topology sockets="2" cores="2" threads="2"/>
  <cache mode="passthrough"/>
  <numa>
   <cell id="0" cpus="0-3" memory="2" unit="GiB">
    <distances>
     <sibling id="0" value="10"/>
     <sibling id="1" value="21"/>
    </distances>
   </cell>
   <cell id="1" cpus="4-7" memory="2" unit="GiB">
    <distances>
     <sibling id="0" value="21"/>
     <sibling id="1" value="10"/>
    </distances>
   </cell>
  </numa>
 </cpu>
</domain>
```

2. **Optional:** Test the performance of the VM using the applicable tools and utilities to evaluate the impact of the VM's optimization.

Sample topology

- The following tables illustrate the connections between the vCPUs and the host CPUs they should be pinned to:

Tableau 13.1. Host topology

CPU threads	0	3	1	4	2	5	6	9	7	10	8	11
Cores	0		1		2		3		4		5	
Sockets	0						1					
NUMA nodes	0						1					

Tableau 13.2. VM topology

vCPU threads	0	1	2	3	4	5	6	7
Cores	0		1		2		3	
Sockets	0				1			
NUMA nodes	0				1			

Tableau 13.3. Combined host and VM topology

vCPU threads			0	1	2	3			4	5	6	7
Host CPU threads	0	3	1	4	2	5	6	9	7	10	8	11
Cores	0		1		2		3		4		5	
Sockets	0						1					
NUMA nodes	0						1					

In this scenario, there are 2 NUMA nodes and 8 vCPUs. Therefore, 4 vCPU threads should be pinned to each node.

In addition, Red Hat recommends leaving at least a single CPU thread available on each node for host system operations.

Because in this example, each NUMA node houses 3 cores, each with 2 host CPU threads, the set for node 0 translates as follows:

```
<vcpupin vcpu='0' cpuset='1'/>
<vcpupin vcpu='1' cpuset='4'/>
<vcpupin vcpu='2' cpuset='2'/>
<vcpupin vcpu='3' cpuset='5'/>
```

13.6.5. Managing kernel same-page merging

Kernel Same-Page Merging (KSM) improves memory density by sharing identical memory pages between virtual machines (VMs). However, enabling KSM increases CPU utilization, and might adversely affect overall performance depending on the workload.

Depending on your requirements, you can either enable or disable KSM for a single session or persistently.

NOTE

In RHEL 9 and later, KSM is disabled by default.

Conditions préalables

- Root access to your host system.

Procédure

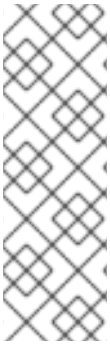
- Disable KSM:
  - To deactivate KSM for a single session, use the **systemctl** utility to stop **ksm** and **ksmtuned** services.

```
# systemctl stop ksm

# systemctl stop ksmtuned
```

  - To deactivate KSM persistently, use the **systemctl** utility to disable **ksm** and **ksmtuned** services.

```
# systemctl disable ksm
Removed /etc/systemd/system/multi-user.target.wants/ksm.service.
# systemctl disable ksmtuned
Removed /etc/systemd/system/multi-user.target.wants/ksmtuned.service.
```



NOTE

Memory pages shared between VMs before deactivating KSM will remain shared. To stop sharing, delete all the **PageKSM** pages in the system using the following command:

```
# echo 2 > /sys/kernel/mm/ksm/run
```

After anonymous pages replace the KSM pages, the **khugepaged** kernel service will rebuild transparent hugepages on the VM's physical memory.

- Enable KSM:

AVERTISSEMENT

Enabling KSM increases CPU utilization and affects overall CPU performance.

1. Install the **ksmtuned** service:

```
# yum install ksmtuned
```

2. Start the service:
   - To enable KSM for a single session, use the **systemctl** utility to start the **ksm** and **ksmtuned** services.

```
# systemctl start ksm
# systemctl start ksmtuned
```

- To enable KSM persistently, use the **systemctl** utility to enable the **ksm** and **ksmtuned** services.

```
# systemctl enable ksm
Created symlink /etc/systemd/system/multi-user.target.wants/ksm.service →
/usr/lib/systemd/system/ksm.service

# systemctl enable ksmtuned
Created symlink /etc/systemd/system/multi-user.target.wants/ksmtuned.service →
/usr/lib/systemd/system/ksmtuned.service
```

13.7. OPTIMIZING VIRTUAL MACHINE NETWORK PERFORMANCE

Due to the virtual nature of a VM's network interface card (NIC), the VM loses a portion of its allocated host network bandwidth, which can reduce the overall workload efficiency of the VM. The following tips can minimize the negative impact of virtualization on the virtual NIC (vNIC) throughput.

Procédure

Use any of the following methods and observe if it has a beneficial effect on your VM network performance:

Enable the vhost_net module

On the host, ensure the **vhost_net** kernel feature is enabled:

```
# lsmod | grep vhost
vhost_net              32768  1
vhost                  53248  1 vhost_net
tap                    24576  1 vhost_net
tun                    57344  6 vhost_net
```

If the output of this command is blank, enable the **vhost_net** kernel module:

```
# modprobe vhost_net
```

Set up multi-queue virtio-net

To set up the *multi-queue virtio-net* feature for a VM, use the **virsh edit** command to edit to the XML configuration of the VM. In the XML, add the following to the **<devices>** section, and replace **N** with the number of vCPUs in the VM, up to 16:

```
<interface type='network'>
    <source network='default'/>
    <model type='virtio'/>
    <driver name='vhost' queues='N'/>
</interface>
```

If the VM is running, restart it for the changes to take effect.

Batching network packets

In Linux VM configurations with a long transmission path, batching packets before submitting them to the kernel may improve cache utilization. To set up packet batching, use the following command on the host, and replace *tap0* with the name of the network interface that the VMs use:

-

```
# ethtool -C tap0 rx-frames 64
```

SR-IOV

Si votre carte réseau hôte prend en charge SR-IOV, utilisez l'affectation de périphériques SR-IOV pour vos cartes réseau virtuelles. Pour plus d'informations, voir Gestion des périphériques SR-IOV.

Ressources supplémentaires

- Comprendre les réseaux virtuels

13.8. VIRTUAL MACHINE PERFORMANCE MONITORING TOOLS

To identify what consumes the most VM resources and which aspect of VM performance needs optimization, performance diagnostic tools, both general and VM-specific, can be used.

Default OS performance monitoring tools

For standard performance evaluation, you can use the utilities provided by default by your host and guest operating systems:

- On your RHEL 9 host, as root, use the **top** utility or the **system monitor** application, and look for **qemu** and **virt** in the output. This shows how much host system resources your VMs are consuming.
  - If the monitoring tool displays that any of the **qemu** or **virt** processes consume a large portion of the host CPU or memory capacity, use the **perf** utility to investigate. For details, see below.
  - In addition, if a **vhost_net** thread process, named for example *vhost_net-1234*, is displayed as consuming an excessive amount of host CPU capacity, consider using virtual network optimization features, such as **multi-queue virtio-net**.
- On the guest operating system, use performance utilities and applications available on the system to evaluate which processes consume the most system resources.
  - On Linux systems, you can use the **top** utility.
  - On Windows systems, you can use the **Task Manager** application.

perf kvm

You can use the **perf** utility to collect and analyze virtualization-specific statistics about the performance of your RHEL 9 host. To do so:

1. On the host, install the *perf* package:

   ```
   # dnf install perf
   ```

2. Use one of the **perf kvm stat** commands to display perf statistics for your virtualization host:
   - For real-time monitoring of your hypervisor, use the **perf kvm stat live** command.
   - To log the perf data of your hypervisor over a period of time, activate the logging using the **perf kvm stat record** command. After the command is canceled or interrupted, the data is saved in the **perf.data.guest** file, which can be analyzed using the **perf kvm stat report** command.

3. Analyze the **perf** output for types of **VM-EXIT** events and their distribution. For example, the **PAUSE_INSTRUCTION** events should be infrequent, but in the following output, the high occurrence of this event suggests that the host CPUs are not handling the running vCPUs well. In such a scenario, consider shutting down some of your active VMs, removing vCPUs from these VMs, or tuning the performance of the vCPUs.

```
# perf kvm stat report

Analyze events for all VMs, all VCPUs:


             VM-EXIT    Samples  Samples%     Time%    Min Time    Max Time         Avg time

  EXTERNAL_INTERRUPT     365634    31.59%    18.04%      0.42us  58780.59us    204.08us ( +-   0.99% )
            MSR_WRITE     293428    25.35%     0.13%      0.59us  17873.02us      1.80us ( +-   4.63% )
     PREEMPTION_TIMER     276162    23.86%     0.23%      0.51us  21396.03us      3.38us ( +-   5.19% )
   PAUSE_INSTRUCTION     189375    16.36%    11.75%      0.72us  29655.25us    256.77us ( +-   0.70% )
                 HLT      20440     1.77%    69.83%      0.62us  79319.41us  14134.56us ( +-   0.79% )
             VMCALL      12426     1.07%     0.03%      1.02us   5416.25us      8.77us ( +-   7.36% )
       EXCEPTION_NMI         27     0.00%     0.00%      0.69us      1.34us      0.98us ( +-   3.50% )
       EPT_MISCONFIG          5     0.00%     0.00%      5.15us     10.85us      7.88us ( +-  11.67% )

Total Samples:1157497, Total events handled time:413728274.66us.
```

Other event types that can signal problems in the output of **perf kvm stat** include:

- **INSN_EMULATION** - suggests suboptimal VM I/O configuration.

For more information on using **perf** to monitor virtualization performance, see the **perf-kvm** man page.

numastat

To see the current NUMA configuration of your system, you can use the **numastat** utility, which is provided by installing the **numactl** package.

The following shows a host with 4 running VMs, each obtaining memory from multiple NUMA nodes. This is not optimal for vCPU performance, and warrants adjusting:

```
# numastat -c qemu-kvm

Per-node process memory usage (in MBs)
PID              Node 0 Node 1 Node 2 Node 3 Node 4 Node 5 Node 6 Node 7 Total
---------------  ------ ------ ------ ------ ------ ------ ------ ------ -----
51722 (qemu-kvm)     68     16    357   6936      2      3    147    598  8128
51747 (qemu-kvm)    245     11      5     18   5172   2532      1     92  8076
53736 (qemu-kvm)     62    432   1661    506   4851    136     22    445  8116
```

```
53773 (qemu-kvm)   1393     3     1     2    12     0     0   6702  8114
---------------  ------ ------ ------ ------ ------ ------ ------ ------ -----
Total              1769    463   2024   7462  10037   2672    169   7837 32434
```

In contrast, the following shows memory being provided to each VM by a single node, which is significantly more efficient.

```
# numastat -c qemu-kvm

Per-node process memory usage (in MBs)
PID               Node 0 Node 1 Node 2 Node 3 Node 4 Node 5 Node 6 Node 7 Total
---------------  ------ ------ ------ ------ ------ ------ ------ ------ -----
51747 (qemu-kvm)      0      0      7      0   8072      0      1      0  8080
53736 (qemu-kvm)      0      0      7      0      0      0   8113      0  8120
53773 (qemu-kvm)      0      0      7      0      0      0      1   8110  8118
59065 (qemu-kvm)      0      0   8050      0      0      0      0      0  8051
---------------  ------ ------ ------ ------ ------ ------ ------ ------ -----
Total                 0      0   8072      0   8072      0   8114   8110 32368
```

13.9. RESSOURCES SUPPLÉMENTAIRES

- Optimisation des machines virtuelles Windows

CHAPITRE 14. IMPORTANCE DE LA GESTION DE L'ÉNERGIE

La réduction de la consommation d'énergie globale des systèmes informatiques permet de réaliser des économies. L'optimisation efficace de la consommation d'énergie de chaque composant du système passe par l'étude des différentes tâches effectuées par le système et par la configuration de chaque composant pour s'assurer que ses performances sont adaptées à cette tâche. La réduction de la consommation d'énergie d'un composant spécifique ou du système dans son ensemble permet de réduire la chaleur et les performances.

Une bonne gestion de l'énergie se traduit par :

- réduction de la chaleur pour les serveurs et les centres de calcul
- réduction des coûts secondaires, notamment en ce qui concerne le refroidissement, l'espace, les câbles, les générateurs et les alimentations sans interruption (ASI)
- prolongation de la durée de vie de la batterie des ordinateurs portables
- réduction de la production de dioxyde de carbone
- respecter les réglementations gouvernementales ou les exigences légales concernant les technologies de l'information vertes, par exemple Energy Star
- respecter les lignes directrices de l'entreprise pour les nouveaux systèmes

Cette section décrit les informations relatives à la gestion de l'alimentation de vos systèmes Red Hat Enterprise Linux.

14.1. PRINCIPES DE BASE DE LA GESTION DE L'ÉNERGIE

Une gestion efficace de l'énergie repose sur les principes suivants :

An idle CPU should only wake up when needed

Depuis Red Hat Enterprise Linux 6, le noyau fonctionne sur **tickless**, ce qui signifie que les interruptions périodiques précédentes ont été remplacées par des interruptions à la demande. Par conséquent, les CPU inactifs sont autorisés à rester inactifs jusqu'à ce qu'une nouvelle tâche soit mise en file d'attente pour le traitement, et les CPU qui sont entrés dans des états de puissance plus faibles peuvent rester dans ces états plus longtemps. Toutefois, les avantages de cette fonction peuvent être annulés si votre système comporte des applications qui créent des événements de temporisation inutiles. Les événements d'interrogation, tels que les vérifications des changements de volume ou des mouvements de la souris, sont des exemples de tels événements.
Red Hat Enterprise Linux comprend des outils qui vous permettent d'identifier et d'auditer des applications sur la base de leur utilisation du processeur. Pour plus d'informations, consultez les sections Vue d'ensemble de l'audit et de l'analyse et Outils d'audit.

Unused hardware and devices should be disabled completely

C'est le cas des périphériques qui ont des pièces mobiles, comme les disques durs. En outre, certaines applications peuvent laisser un périphérique inutilisé mais activé "ouvert" ; dans ce cas, le noyau suppose que le périphérique est utilisé, ce qui peut empêcher le périphérique de passer en état d'économie d'énergie.

Low activity should translate to low wattage

Dans de nombreux cas, cependant, cela dépend d'un matériel moderne et d'une configuration correcte du BIOS ou de l'UEFI sur les systèmes modernes, y compris les architectures non-x86. Assurez-vous que vous utilisez le dernier micrologiciel officiel pour vos systèmes et que les fonctions

de gestion de l'énergie sont activées dans les sections de gestion de l'énergie ou de configuration des périphériques du BIOS. Parmi les fonctions à rechercher, citons

- Prise en charge du contrôle collaboratif des performances des processeurs (CPPC) pour ARM64
- Prise en charge de PowerNV pour les systèmes Power d'IBM
- SpeedStep
- PowerNow !
- Cool'n'Quiet
- ACPI (état C)
- Intelligent
  Si votre matériel prend en charge ces fonctionnalités et qu'elles sont activées dans le BIOS, Red Hat Enterprise Linux les utilise par défaut.

Different forms of CPU states and their effects

Les unités centrales modernes et l'interface ACPI (Advanced Configuration and Power Interface) proposent différents états d'alimentation. Les trois états différents sont les suivants :

- Sommeil (états C)
- Fréquence et tension (états P)
- Production de chaleur (états T ou états thermiques)
  Un processeur fonctionnant dans l'état de veille le plus bas consomme le moins de watts, mais il faut aussi beaucoup plus de temps pour le réveiller de cet état lorsque c'est nécessaire. Dans de très rares cas, cela peut conduire à ce que l'unité centrale doive se réveiller immédiatement chaque fois qu'elle vient de s'endormir. Dans ce cas, l'unité centrale est occupée en permanence et perd une partie de l'économie d'énergie potentielle qu'aurait permis l'utilisation d'un autre état.

A turned off machine uses the least amount of power

L'un des meilleurs moyens d'économiser de l'énergie est d'éteindre les systèmes. Par exemple, votre entreprise peut développer une culture d'entreprise axée sur la sensibilisation à l'"informatique verte", avec pour consigne d'éteindre les machines pendant la pause déjeuner ou en rentrant chez soi. Vous pouvez également consolider plusieurs serveurs physiques en un seul serveur plus grand et les virtualiser à l'aide de la technologie de virtualisation fournie avec Red Hat Enterprise Linux.

14.2. VUE D'ENSEMBLE DE L'AUDIT ET DE L'ANALYSE

L'audit manuel détaillé, l'analyse et le réglage d'un système unique sont généralement l'exception, car le temps et le coût consacrés à cette opération dépassent généralement les avantages tirés de ces derniers éléments de réglage du système.

Toutefois, il peut être très utile d'effectuer ces tâches une seule fois pour un grand nombre de systèmes presque identiques, en réutilisant les mêmes paramètres pour tous les systèmes. Prenons l'exemple du déploiement de milliers d'ordinateurs de bureau ou d'un cluster HPC dont les machines sont presque identiques. Une autre raison d'effectuer des audits et des analyses est de fournir une base de comparaison permettant d'identifier les régressions ou les changements dans le comportement du système à l'avenir. Les résultats de cette analyse peuvent être très utiles dans les cas où des mises à jour

du matériel, du BIOS ou des logiciels ont lieu régulièrement et où vous souhaitez éviter toute surprise en ce qui concerne la consommation d'énergie. En règle générale, un audit et une analyse approfondis vous donnent une bien meilleure idée de ce qui se passe réellement sur un système donné.

L'audit et l'analyse d'un système en ce qui concerne la consommation d'énergie sont relativement difficiles, même avec les systèmes les plus modernes disponibles. La plupart des systèmes ne fournissent pas les moyens nécessaires pour mesurer la consommation d'énergie par le biais d'un logiciel. Il existe cependant des exceptions :

- la console de gestion iLO des systèmes de serveurs Hewlett Packard dispose d'un module de gestion de l'alimentation auquel vous pouvez accéder via le web.
- IBM propose une solution similaire dans son module de gestion de l'énergie BladeCenter.
- Sur certains systèmes Dell, l'assistant informatique offre également des fonctions de surveillance de l'alimentation.

D'autres fournisseurs sont susceptibles d'offrir des capacités similaires pour leurs plates-formes de serveurs, mais comme on peut le voir, il n'existe pas de solution unique prise en charge par tous les fournisseurs. Les mesures directes de la consommation d'énergie ne sont souvent nécessaires que pour maximiser les économies dans la mesure du possible.

14.3. OUTILS D'AUDIT

Red Hat Enterprise Linux 8 propose des outils permettant d'effectuer l'audit et l'analyse du système. La plupart d'entre eux peuvent être utilisés comme sources d'informations supplémentaires au cas où vous souhaiteriez vérifier ce que vous avez déjà découvert ou au cas où vous auriez besoin d'informations plus approfondies sur certaines parties.

Nombre de ces outils sont également utilisés pour l'optimisation des performances :

PowerTOP

Il identifie les composants spécifiques des applications du noyau et de l'espace utilisateur qui réveillent fréquemment le processeur. Utilisez la commande **powertop** en tant que root pour lancer l'outil **PowerTop** et **powertop --calibrate** pour calibrer le moteur d'estimation de la consommation d'énergie. Pour plus d'informations sur PowerTop, voir Gérer la consommation d'énergie avec PowerTOP.

Diskdevstat and netdevstat

Il s'agit d'outils SystemTap qui collectent des informations détaillées sur l'activité du disque et du réseau de toutes les applications exécutées sur un système. En utilisant les statistiques collectées par ces outils, vous pouvez identifier les applications qui gaspillent de l'énergie avec de nombreuses petites opérations d'E/S plutôt qu'avec un nombre réduit d'opérations plus importantes. En utilisant la commande **dnf install tuned-utils-systemtap kernel-debuginfo** en tant que root, installez les outils **diskdevstat** et **netdevstat**.
Pour afficher les informations détaillées sur l'activité du disque et du réseau, utilisez :

```
# diskdevstat

PID  UID  DEV  WRITE_CNT  WRITE_MIN  WRITE_MAX  WRITE_AVG  READ_CNT
READ_MIN  READ_MAX  READ_AVG  COMMAND

3575  1000  dm-2  59       0.000     0.365      0.006      5       0.000      0.000     0.000
mozStorage #5
3575  1000  dm-2  7        0.000     0.000      0.000      0       0.000      0.000     0.000
localStorage DB
```

```
[...]


# netdevstat

PID  UID  DEV      XMIT_CNT  XMIT_MIN  XMIT_MAX  XMIT_AVG  RECV_CNT
RECV_MIN  RECV_MAX  RECV_AVG  COMMAND
3572 991  enp0s31f6  40      0.000     0.882     0.108     0       0.000     0.000     0.000
openvpn
3575 1000 enp0s31f6  27      0.000     1.363     0.160     0       0.000     0.000     0.000
Socket Thread
[...]
```

Avec ces commandes, vous pouvez spécifier trois paramètres : **update_interval**, **total_duration**, et **display_histogram**.

TuneD

Il s'agit d'un outil de réglage du système basé sur des profils qui utilise le gestionnaire de périphériques **udev** pour surveiller les périphériques connectés et qui permet un réglage statique et dynamique des paramètres du système. Vous pouvez utiliser la commande **tuned-adm recommend** pour déterminer quel profil Red Hat recommande comme étant le plus approprié pour un produit particulier. Pour plus d'informations sur TuneD, reportez-vous aux sections Premiers pas avec TuneD et Personnalisation des profils TuneD. En utilisant l'utilitaire **powertop2tuned utility**, vous pouvez créer des profils TuneD personnalisés à partir des suggestions de **PowerTOP**. Pour plus d'informations sur l'utilitaire **powertop2tuned**, voir Optimiser la consommation d'énergie .

Virtual memory statistics (vmstat)

Il est fourni par le paquetage **procps-ng**. Cet outil permet d'afficher des informations détaillées sur les processus, la mémoire, la pagination, les entrées/sorties par bloc, les pièges et l'activité de l'unité centrale.
Pour afficher ces informations, utilisez :

```
$ vmstat
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
r  b  swpd  free    buff   cache   si   so  bi   bo   in  cs  us  sy id  wa  st
1  0   0   5805576 380856 4852848   0    0  119  73  814  640  2   2 96   0   0
```

La commande **vmstat -a** permet d'afficher la mémoire active et inactive. Pour plus d'informations sur les autres options de **vmstat**, consultez la page de manuel **vmstat**.

iostat

Il est fourni par le paquetage **sysstat**. Cet outil est similaire à **vmstat**, mais uniquement pour la surveillance des E/S sur les périphériques en mode bloc. Il fournit également des statistiques et des résultats plus détaillés.
Pour surveiller les entrées/sorties du système, utilisez

```
$ iostat
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          2.05    0.46    1.55    0.26    0.00   95.67

Device     tps     kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
nvme0n1    53.54     899.48     616.99      3445229     2363196
```

```
dm-0      42.84    753.72    238.71     2886921     914296
dm-1       0.03      0.60      0.00        2292          0
dm-2      24.15    143.12    379.80      548193    1454712
```

blktrace

Il fournit des informations détaillées sur le temps passé dans le sous-système d'E/S.
Pour visualiser ces informations dans un format lisible par l'homme, utilisez :

```
# blktrace -d /dev/dm-0 -o - | blkparse -i -

253,0   1    1   0.000000000  17694  Q   W 76423384 + 8 [kworker/u16:1]
253,0   2    1   0.001926913     0   C   W 76423384 + 8 [0]
[...]
```

Ici, la première colonne, **253,0**, est le tuple majeur et mineur du périphérique. La deuxième colonne, **1**, donne des informations sur le processeur, suivies de colonnes pour les horodatages et le PID du processus émettant le processus IO.

La sixième colonne, **Q**, indique le type d'événement, la septième colonne, **W** pour l'opération d'écriture, la huitième colonne, **76423384**, est le numéro de bloc, et **8** est le nombre de blocs demandés.

Le dernier champ, **[kworker/u16:1]**, est le nom du processus.

Par défaut, la commande **blktrace** s'exécute indéfiniment jusqu'à ce que le processus soit explicitement tué. L'option **-w** permet de spécifier la durée d'exécution.

turbostat

Il est fourni par le paquetage **kernel-tools**. Il fournit des informations sur la topologie du processeur, la fréquence, les statistiques sur l'état de fonctionnement au repos, la température et la consommation d'énergie des processeurs x86-64.
Pour consulter ce résumé, utilisez :

```
# turbostat

CPUID(0): GenuineIntel 0x16 CPUID levels; 0x80000008 xlevels; family:model:stepping 0x6:8e:a
(6:142:10)
CPUID(1): SSE3 MONITOR SMX EIST TM2 TSC MSR ACPI-TM HT TM
CPUID(6): APERF, TURBO, DTS, PTM, HWP, HWPnotify, HWPwindow, HWPepp, No-HWPpkg,
EPB
[...]
```

Par défaut, **turbostat** imprime un résumé des résultats du compteur pour l'ensemble de l'écran, suivi des résultats du compteur toutes les 5 secondes. Spécifiez une période différente entre les résultats du compteur avec l'option **-i**, par exemple, exécutez **turbostat -i 10** pour imprimer les résultats toutes les 10 secondes à la place.

Turbostat est également utile pour identifier les serveurs qui sont inefficaces en termes de consommation d'énergie ou de temps d'inactivité. Il permet également d'identifier le taux d'interruptions de gestion du système (SMI) survenant sur le système. Il peut également être utilisé pour vérifier les effets des réglages de la gestion de l'énergie.

cpupower

IT est une collection d'outils permettant d'examiner et de régler les fonctions d'économie d'énergie des processeurs. Utilisez la commande **cpupower** avec les options **frequency-info**, **frequency-set**, **idle-info**, **idle-set**, **set**, **info**, et **monitor** pour afficher et définir les valeurs relatives au processeur. Par exemple, pour afficher les gouverneurs cpufreq disponibles, utilisez la commande suivante

```
$ cpupower frequency-info --governors
analyzing CPU 0:
  available cpufreq governors: performance powersave
```

Pour plus d'informations sur **cpupower**, voir Affichage des informations relatives à l'unité centrale.

GNOME Power Manager

Il s'agit d'un démon installé dans le cadre de l'environnement de bureau GNOME. Le gestionnaire d'alimentation GNOME vous informe des changements dans l'état de l'alimentation de votre système, par exemple, le passage de la batterie à l'alimentation secteur. Il signale également l'état de la batterie et vous avertit lorsque celle-ci est faible.

Ressources supplémentaires

- **powertop(1)**pages de manuel : **diskdevstat(8)**, **netdevstat(8)**, **tuned(8)**, **vmstat(8)**, **iostat(1)**, **blktrace(8)**, **blkparse(8)**, et **turbostat(8)**
- **cpupower(1)**pages de manuel : **cpupower-set(1)**, **cpupower-info(1)**, **cpupower-idle(1)**, **cpupower-frequency-set(1)**, **cpupower-frequency-info(1)**, et **cpupower-monitor(1)**

CHAPITRE 15. GÉRER LA CONSOMMATION D'ÉNERGIE AVEC POWERTOP

En tant qu'administrateur système, vous pouvez utiliser l'outil **PowerTOP** pour analyser et gérer la consommation d'énergie.

15.1. L'OBJECTIF DE POWERTOP

PowerTOP est un programme qui diagnostique les problèmes liés à la consommation d'énergie et fournit des suggestions sur la manière de prolonger la durée de vie de la batterie.

L'outil **PowerTOP** peut fournir une estimation de la consommation totale d'énergie du système ainsi que de la consommation individuelle d'énergie pour chaque processus, périphérique, travailleur du noyau, temporisateur et gestionnaire d'interruptions. L'outil peut également identifier les composants spécifiques des applications du noyau et de l'espace utilisateur qui réveillent fréquemment le processeur.

Red Hat Enterprise Linux 9 utilise la version 2.x de **PowerTOP**.

15.2. UTILISATION DE POWERTOP

Conditions préalables

- Pour pouvoir utiliser **PowerTOP**assurez-vous que le paquetage **powertop** a été installé sur votre système :

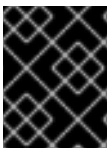
```
# dnf install powertop
```

15.2.1. Démarrage de PowerTOP

Procédure

- Pour exécuter **PowerTOP**utilisez la commande suivante :

```
# powertop
```



IMPORTANT

Les ordinateurs portables doivent fonctionner sur batterie lors de l'exécution de la commande **powertop**.

15.2.2. Étalonnage de PowerTOP

Procédure

1. Sur un ordinateur portable, vous pouvez calibrer le moteur d'estimation de puissance en exécutant la commande suivante :

```
# powertop --calibrate
```

2. Laissez l'étalonnage se terminer sans interagir avec la machine pendant le processus. L'étalonnage prend du temps car le processus effectue divers tests, passe par des niveaux de luminosité et allume et éteint les appareils.

3. Lorsque le processus d'étalonnage est terminé, **PowerTOP** démarre normalement. Laissez-le fonctionner pendant environ une heure pour collecter les données. Lorsque suffisamment de données sont collectées, les chiffres de l'estimation de la puissance sont affichés dans la première colonne du tableau de sortie.

NOTE

Notez que **powertop --calibrate** ne peut être utilisé que sur des ordinateurs portables.

15.2.3. Réglage de l'intervalle de mesure

Par défaut, **PowerTOP** prend des mesures à intervalles de 20 secondes.

Si vous souhaitez modifier cette fréquence de mesure, utilisez la procédure suivante :

Procédure

- Exécutez la commande **powertop** avec l'option **--time**:

```
# powertop --time=time in seconds
```

15.2.4. Ressources supplémentaires

Pour plus de détails sur l'utilisation de **PowerTOP**voir la page de manuel **powertop**.

15.3. STATISTIQUES POWERTOP

Pendant qu'il s'exécute, **PowerTOP** recueille des statistiques sur le système.

PowerTOPfournit plusieurs onglets :

- **Overview**
- **Idle stats**
- **Frequency stats**
- **Device stats**
- **Tunables**
- **WakeUp**

Vous pouvez utiliser les touches **Tab** et **Shift Tab** pour parcourir ces onglets.

15.3.1. L'onglet Vue d'ensemble

Dans l'onglet **Overview**, vous pouvez afficher une liste des composants qui envoient le plus souvent des réveils au processeur ou qui consomment le plus d'énergie. Les éléments de l'onglet **Overview**, y compris les processus, les interruptions, les périphériques et les autres ressources, sont triés en fonction

de leur utilisation.

Les colonnes adjacentes de l'onglet **Overview** fournissent les informations suivantes :

Utilisation

Estimation de la puissance de l'utilisation de la ressource.

Événements/s

Réveils par seconde. Le nombre de réveils par seconde indique l'efficacité des services ou des périphériques et pilotes du noyau. Moins de réveils signifie que moins d'énergie est consommée. Les composants sont classés en fonction de l'optimisation de leur consommation d'énergie.

Catégorie

Classification du composant, par exemple processus, dispositif ou temporisateur.

Description

Description du composant.

S'il est correctement calibré, une estimation de la consommation d'énergie pour chaque élément répertorié dans la première colonne est également affichée.

En outre, l'onglet **Overview** comprend une ligne avec des statistiques sommaires telles que

- Consommation électrique totale
- Durée de vie restante de la batterie (uniquement si applicable)
- Résumé du nombre total de réveils par seconde, d'opérations du GPU par seconde et d'opérations du système de fichiers virtuels par seconde

15.3.2. L'onglet Statistiques d'inactivité

L'onglet **Idle stats** montre l'utilisation des états C pour tous les processeurs et cœurs, tandis que l'onglet **Frequency stats** montre l'utilisation des états P, y compris le mode Turbo, le cas échéant, pour tous les processeurs et cœurs. La durée des états C ou P indique dans quelle mesure l'utilisation du processeur a été optimisée. Plus le CPU reste longtemps dans des états C ou P élevés (par exemple, C4 est plus élevé que C3), meilleure est l'optimisation de l'utilisation du CPU. Idéalement, le taux de résidence est de 90 % ou plus dans les états C ou P les plus élevés lorsque le système est inactif.

15.3.3. L'onglet Statistiques de l'appareil

L'onglet **Device stats** fournit des informations similaires à l'onglet **Overview**, mais uniquement pour les appareils.

15.3.4. L'onglet Tunables

L'onglet **Tunables** contient les suggestions de **PowerTOP**l'onglet contient les suggestions d'optimisation du système pour réduire la consommation d'énergie.

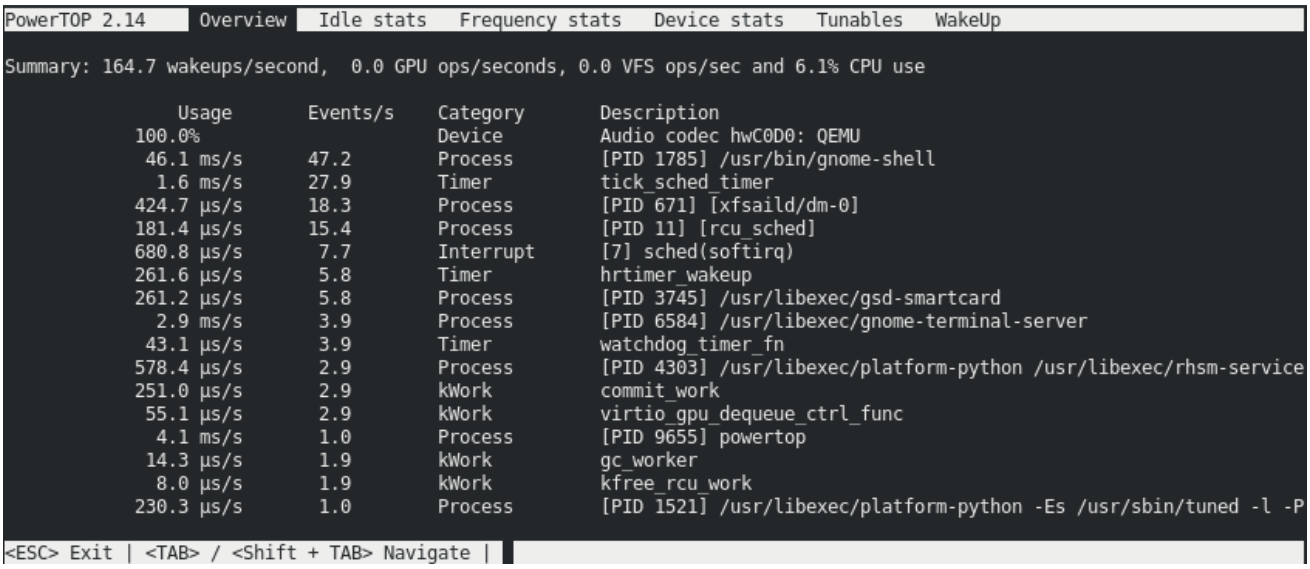
Utilisez les touches **up** et **down** pour vous déplacer dans les suggestions, et la touche **enter** pour activer ou désactiver la suggestion.

15.3.5. L'onglet WakeUp

L'onglet **WakeUp** affiche les paramètres de réveil de l'appareil que les utilisateurs peuvent modifier au besoin.

Utilisez les touches **up** et **down** pour vous déplacer parmi les paramètres disponibles, et la touche **enter** pour activer ou désactiver un paramètre.

Figure 15.1. Sortie PowerTOP



Ressources supplémentaires

Pour plus de détails sur **PowerTOP**voir la page d'accueil de PowerTOP .

15.4. POURQUOI POWERTOP N'AFFICHE-T-IL PAS LES VALEURS DES STATISTIQUES DE FRÉQUENCE DANS CERTAINS CAS ?

Lors de l'utilisation du pilote Intel P-State, PowerTOP n'affiche les valeurs dans l'onglet **Frequency Stats** que si le pilote est en mode passif. Mais, même dans ce cas, les valeurs peuvent être incomplètes.

Au total, il existe trois modes possibles pour le pilote Intel P-State :

- Mode actif avec états P matériels (HWP)
- Mode actif sans HWP
- Mode passif

Le passage au pilote ACPI CPUfreq permet à PowerTOP d'afficher des informations complètes. Toutefois, il est recommandé de conserver les paramètres par défaut de votre système.

Pour voir quel pilote est chargé et dans quel mode, exécutez :

```
# cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_driver
```

- **intel_pstate** est renvoyée si le pilote Intel P-State est chargé et en mode actif.
- **intel_cpufreq** est renvoyée si le pilote Intel P-State est chargé et en mode passif.
- **acpi-cpufreq** est renvoyé si le pilote ACPI CPUfreq est chargé.

Lorsque vous utilisez le pilote Intel P-State, ajoutez l'argument suivant à la ligne de commande de démarrage du noyau pour forcer le pilote à fonctionner en mode passif :

```
intel_pstate=passive
```

Pour désactiver le pilote Intel P-State et utiliser à la place le pilote ACPI CPUfreq, ajoutez l'argument suivant à la ligne de commande de démarrage du noyau :

```
intel_pstate=disable
```

15.5. GÉNÉRER UNE SORTIE HTML

Outre la sortie **powertop's** dans le terminal, vous pouvez également générer un rapport HTML.

Procédure

- Exécutez la commande **powertop** avec l'option **--html**:

  ```
  # powertop --html=htmlfile.html
  ```

  Remplacez le paramètre **htmlfile.html** par le nom requis pour le fichier de sortie.

15.6. OPTIMISER LA CONSOMMATION D'ÉNERGIE

Pour optimiser la consommation d'énergie, vous pouvez utiliser le service **powertop** ou l'utilitaire **powertop2tuned**.

15.6.1. Optimisation de la consommation d'énergie à l'aide du service powertop

Vous pouvez utiliser le service **powertop** pour activer automatiquement toutes les suggestions de **PowerTOP**vous pouvez utiliser le service **Tunables** pour activer automatiquement toutes les suggestions de dans l'onglet "boot" :

Procédure

- Activer le service **powertop**:

  ```
  # systemctl enable powertop
  ```

15.6.2. L'utilitaire powertop2tuned

L'utilitaire **powertop2tuned** vous permet de créer des profils personnalisés **TuneD** à partir de **PowerTOP** suggestions.

Par défaut, **powertop2tuned** crée des profils dans le répertoire **/etc/tuned/** et base le profil personnalisé sur le profil actuellement sélectionné **TuneD** sélectionné. Pour des raisons de sécurité, tous les **PowerTOP** sont initialement désactivés dans le nouveau profil.

Pour activer les réglages, vous pouvez

- Décommentez-les dans le site **/etc/tuned/profile_name/tuned.conf file**.
- Utilisez l'option **--enable** ou **-e** pour générer un nouveau profil qui permet la plupart des accords suggérés par **PowerTOP**.

Certains réglages potentiellement problématiques, tels que l'autosuspend USB, sont désactivés par défaut et doivent être décommentés manuellement.

15.6.3. Optimisation de la consommation d'énergie à l'aide de l'utilitaire powertop2tuned

Conditions préalables

- L'utilitaire **powertop2tuned** est installé sur le système :

```
# dnf install tuned-utils
```

Procédure

1. Créer un profil personnalisé :

```
# powertop2tuned nouveau_nom_du_profil
```

2. Activer le nouveau profil :

```
# tuned-adm profile new_profile_name
```

Informations complémentaires

- Pour obtenir une liste complète des options prises en charge par **powertop2tuned**, utilisez le lien suivant :

```
powertop2tuned --help
```

15.6.4. Comparaison entre powertop.service et powertop2tuned

L'optimisation de la consommation d'énergie avec **powertop2tuned** est préférable à **powertop.service** pour les raisons suivantes :

- L'utilitaire **powertop2tuned** représente l'intégration de **PowerTOP** dans **TuneD**qui permet de bénéficier des avantages des deux outils.
- L'utilitaire **powertop2tuned** permet un contrôle fin des réglages activés.
- Avec **powertop2tuned**, les réglages potentiellement dangereux ne sont pas automatiquement activés.
- Avec **powertop2tuned**, le retour en arrière est possible sans redémarrage.

CHAPITRE 16. DÉMARRER AVEC PERF

En tant qu'administrateur système, vous pouvez utiliser l'outil **perf** pour collecter et analyser les données de performance de votre système.

16.1. INTRODUCTION À LA PERF

L'outil de l'espace utilisateur **perf** s'interface avec le sous-système basé sur le noyau *Performance Counters for Linux* (PCL). **perf** est un outil puissant qui utilise l'unité de surveillance des performances (PMU) pour mesurer, enregistrer et surveiller une variété d'événements matériels et logiciels. **perf** prend également en charge les tracepoints, les kprobes et les uprobes.

16.2. INSTALLATION DE PERF

Cette procédure installe l'outil **perf** dans l'espace utilisateur.

Procédure

- Installer l'outil **perf**:

```
# dnf install perf
```

16.3. COMMANDES COURANTES DE PERF

perf stat

Cette commande fournit des statistiques globales pour les événements de performance courants, notamment les instructions exécutées et les cycles d'horloge consommés. Des options permettent de sélectionner des événements autres que les événements de mesure par défaut.

perf record

Cette commande enregistre les données de performance dans un fichier, **perf.data**, qui peut être analysé ultérieurement à l'aide de la commande **perf report**.

perf report

Cette commande permet de lire et d'afficher les données de performance du fichier **perf.data** créé par **perf record**.

perf list

Cette commande dresse la liste des événements disponibles sur une machine donnée. Ces événements varient en fonction de la configuration matérielle et logicielle du système.

perf top

Cette commande a une fonction similaire à celle de l'utilitaire **top**. Elle génère et affiche un profil de compteur de performance en temps réel.

perf trace

Cette commande a une fonction similaire à celle de l'outil **strace**. Elle surveille les appels système utilisés par un thread ou un processus spécifié et tous les signaux reçus par cette application.

perf help

Cette commande permet d'afficher la liste complète des commandes **perf**.

Ressources supplémentaires

- Ajouter l'option **--help** à une sous-commande pour ouvrir la page de manuel.

CHAPITRE 17. PROFILAGE DE L'UTILISATION DE L'UNITÉ CENTRALE EN TEMPS RÉEL AVEC PERF TOP

Vous pouvez utiliser la commande **perf top** pour mesurer l'utilisation de l'unité centrale de différentes fonctions en temps réel.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.

17.1. L'OBJECTIF DE PERF TOP

La commande **perf top** est utilisée pour établir le profil du système en temps réel et fonctionne de la même manière que l'utilitaire **top**. Cependant, alors que l'utilitaire **top** vous indique généralement le temps d'utilisation du processeur d'un processus ou d'un thread donné, **perf top** vous indique le temps d'utilisation du processeur de chaque fonction spécifique. Dans son état par défaut, **perf top** vous renseigne sur les fonctions utilisées par tous les processeurs, tant dans l'espace utilisateur que dans l'espace noyau. Pour utiliser **perf top**, vous devez disposer d'un accès root.

17.2. PROFILAGE DE L'UTILISATION DE L'UNITÉ CENTRALE AVEC PERF TOP

Cette procédure active **perf top** et établit le profil de l'utilisation de l'unité centrale en temps réel.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.
- Vous avez un accès root

Procédure

- Lancez l'interface de surveillance **perf top**:

```
# perf top
```

L'interface de surveillance ressemble à ce qui suit :

```
Samples: 8K of event 'cycles', 2000 Hz, Event count (approx.): 4579432780 lost: 0/0 drop:
0/0
Overhead  Shared Object       Symbol
   2.20%  [kernel]            [k] do_syscall_64
   2.17%  [kernel]            [k] module_get_kallsym
   1.49%  [kernel]            [k] copy_user_enhanced_fast_string
   1.37%  libpthread-2.29.so  [.] pthread_mutex_lock 1.31% [unknown] [.] 0000000000000000
1.07% [kernel] [k] psi_task_change 1.04% [kernel] [k] switch_mm_irqs_off 0.94% [kernel] [k]
fget
   0.74%  [kernel]            [k] entry_SYSCALL_64
   0.69%  [kernel]            [k] syscall_return_via_sysret
   0.69%  libxul.so           [.] 0x000000000113f9b0
   0.67%  [kernel]            [k] kallsyms_expand_symbol.constprop.0
   0.65%  firefox             [.] moz_xmalloc
   0.65%  libpthread-2.29.so  [.] __pthread_mutex_unlock_usercnt
```

```
0.60%  firefox          [.] free
0.60%  libxul.so        [.] 0x000000000241d1cd
0.60%  [kernel]         [k] do_sys_poll
0.58%  [kernel]         [k] menu_select
0.56%  [kernel]         [k] _raw_spin_lock_irqsave
0.55%  perf             [.] 0x00000000002ae0f3
```

Dans cet exemple, c'est la fonction noyau **do_syscall_64** qui utilise le plus de temps processeur.

Ressources supplémentaires

- **perf-top(1)** page de manuel

17.3. INTERPRÉTATION DE LA SORTIE DE PERF TOP

L'interface de surveillance **perf top** affiche les données dans plusieurs colonnes :

La colonne "Overhead" (frais généraux)

Affiche le pourcentage de CPU utilisé par une fonction donnée.

La colonne "Objets partagés

Affiche le nom du programme ou de la bibliothèque qui utilise la fonction.

La colonne "Symbol

Affiche le nom ou le symbole de la fonction. Les fonctions exécutées dans l'espace noyau sont identifiées par **[k]** et les fonctions exécutées dans l'espace utilisateur sont identifiées par **[.]**.

17.4. POURQUOI PERF AFFICHE-T-IL CERTAINS NOMS DE FONCTIONS COMME DES ADRESSES DE FONCTIONS BRUTES ?

Pour les fonctions du noyau, **perf** utilise les informations du fichier **/proc/kallsyms** pour faire correspondre les échantillons à leurs noms de fonction ou symboles respectifs. Pour les fonctions exécutées dans l'espace utilisateur, cependant, vous pouvez voir des adresses de fonctions brutes parce que le binaire est dépouillé.

Le paquet **debuginfo** de l'exécutable doit être installé ou, si l'exécutable est une application développée localement, l'application doit être compilée avec les informations de débogage activées (l'option **-g** dans GCC) pour afficher les noms de fonction ou les symboles dans une telle situation.

NOTE

Il n'est pas nécessaire de réexécuter la commande **perf record** après avoir installé la commande **debuginfo** associée à un exécutable. Il suffit de réexécuter la commande **perf report**.

Ressources complémentaires

- Activation du débogage avec les informations de débogage

17.5. ACTIVATION DES DÉPÔTS DE DÉBOGAGE ET DE SOURCES

Une installation standard de Red Hat Enterprise Linux n'active pas les référentiels de débogage et de sources. Ces référentiels contiennent des informations nécessaires pour déboguer les composants du système et mesurer leurs performances.

Procédure

- Activez les canaux du paquet d'informations de source et de débogage : La partie **$(uname -i)** est automatiquement remplacée par une valeur correspondant à l'architecture de votre système :

Nom de l'architecture	Valeur
64-bit Intel et AMD	x86_64
aRM 64 bits	aarch64
IBM POWER	ppc64le
iBM Z 64 bits	s390x

17.6. OBTENIR LES PAQUETS D'INFORMATIONS DE DÉBOGAGE POUR UNE APPLICATION OU UNE BIBLIOTHÈQUE À L'AIDE DE GDB

Les informations de débogage sont nécessaires pour déboguer le code. Pour le code installé à partir d'un paquetage, le débogueur GNU (GDB) reconnaît automatiquement les informations de débogage manquantes, résout le nom du paquetage et fournit des conseils concrets sur la manière d'obtenir le paquetage.

Conditions préalables

- L'application ou la bibliothèque que vous souhaitez déboguer doit être installée sur le système.
- GDB et l'outil **debuginfo-install** doivent être installés sur le système. Pour plus de détails, voir Configuration pour le débogage d'applications .
- Les dépôts fournissant les paquets **debuginfo** et **debugsource** doivent être configurés et activés sur le système. Pour plus de détails, voir Activation des référentiels de débogage et de sources.

Procédure

1. Lancez GDB attaché à l'application ou à la bibliothèque que vous souhaitez déboguer. GDB reconnaît automatiquement les informations de débogage manquantes et suggère une commande à exécuter.

```
$ gdb -q /bin/ls
Reading symbols from /bin/ls...Reading symbols from .gnu_debugdata for /usr/bin/ls...(no debugging symbols found)...done.
(no debugging symbols found)...done.
Missing separate debuginfos, use: dnf debuginfo-install coreutils-8.30-6.el8.x86_64
(gdb)
```

2. Quittez GDB : tapez **q** et confirmez avec **Enter**.

```
(gdb) q
```

3. Exécutez la commande suggérée par GDB pour installer les paquets **debuginfo** requis :

```
# dnf debuginfo-install coreutils-8.30-6.el8.x86_64
```

L'outil de gestion des paquets **dnf** fournit un résumé des changements, demande une confirmation et, une fois que vous avez confirmé, télécharge et installe tous les fichiers nécessaires.

4. Si GDB n'est pas en mesure de suggérer le paquet **debuginfo**, suivez la procédure décrite dans Obtenir manuellement les paquets debuginfo pour une application ou une bibliothèque .

Ressources supplémentaires

- Guide d'utilisation de Red Hat Developer Toolset, section Installation des informations de débogage
- Comment puis-je télécharger ou installer des paquets debuginfo pour les systèmes RHEL ? - Solution de la base de connaissances de Red Hat

CHAPITRE 18. COMPTER LES ÉVÉNEMENTS PENDANT L'EXÉCUTION D'UN PROCESSUS AVEC PERF STAT

Vous pouvez utiliser la commande **perf stat** pour compter les événements matériels et logiciels pendant l'exécution du processus.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.

18.1. L'OBJECTIF DE PERF STAT

La commande **perf stat** exécute une commande spécifiée, comptabilise les événements matériels et logiciels survenus pendant l'exécution de la commande et génère des statistiques à partir de ces chiffres. Si vous ne spécifiez aucun événement, **perf stat** comptabilise un ensemble d'événements matériels et logiciels courants.

18.2. COMPTAGE DES ÉVÉNEMENTS AVEC PERF STAT

Vous pouvez utiliser **perf stat** pour compter les événements matériels et logiciels survenant au cours de l'exécution des commandes et générer des statistiques sur ces comptages. Par défaut, **perf stat** fonctionne en mode "per-thread".

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.

Procédure

- Comptez les événements.
  - L'exécution de la commande **perf stat** sans accès root ne comptabilisera que les événements se produisant dans l'espace utilisateur :

```
$ perf stat ls
```

Exemple 18.1. Sortie de perf stat exécuté sans accès root

```
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos

 Performance counter stats for 'ls':

              1.28 msec task-clock:u              #    0.165 CPUs utilized
                 0      context-switches:u        #    0.000 M/sec
                 0      cpu-migrations:u          #    0.000 K/sec
               104      page-faults:u             #    0.081 M/sec
         1,054,302      cycles:u                  #    0.823 GHz
         1,136,989      instructions:u            #    1.08  insn per cycle
           228,531      branches:u                #  178.447 M/sec
            11,331      branch-misses:u           #    4.96% of all branches

       0.007754312 seconds time elapsed
```

```
0.000000000 seconds user
0.007717000 seconds sys
```

Comme vous pouvez le voir dans l'exemple précédent, lorsque **perf stat** s'exécute sans accès root, les noms des événements sont suivis de **:u**, ce qui indique que ces événements n'ont été comptés que dans l'espace utilisateur.

- Pour compter les événements de l'espace utilisateur et de l'espace noyau, vous devez disposer d'un accès root lorsque vous exécutez **perf stat**:

```
# perf stat ls
```

Exemple 18.2. Résultat de perf stat exécuté avec l'accès root

```
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos

 Performance counter stats for 'ls':

          3.09 msec task-clock                #    0.119 CPUs utilized
            18      context-switches          #    0.006 M/sec
             3      cpu-migrations            #    0.969 K/sec
           108      page-faults               #    0.035 M/sec
     6,576,004      cycles                    #    2.125 GHz
     5,694,223      instructions              #    0.87  insn per cycle
     1,092,372      branches                  #  352.960 M/sec
        31,515      branch-misses             #    2.89% of all branches

       0.026020043 seconds time elapsed

       0.000000000 seconds user
       0.014061000 seconds sys
```

- Par défaut, **perf stat** fonctionne en mode "per-thread". Pour passer au comptage des événements à l'échelle du processeur, passez l'option **-a** à **perf stat**. Pour compter les événements à l'échelle du processeur, vous devez disposer d'un accès root :

```
# perf stat -a ls
```

Ressources supplémentaires

- **perf-stat(1)** page de manuel

18.3. INTERPRÉTATION DE LA SORTIE DE L'ÉTAT DE PERF

perf stat exécute une commande spécifiée et compte les occurrences d'événements pendant l'exécution de la commande et affiche les statistiques de ces comptages dans trois colonnes :

1. Le nombre d'occurrences comptées pour un événement donné
2. Le nom de l'événement qui a été compté

3. Lorsque des mesures connexes sont disponibles, un ratio ou un pourcentage est affiché après le signe dièse (**#**) dans la colonne la plus à droite.
   Par exemple, en mode par défaut, **perf stat** compte à la fois les cycles et les instructions et, par conséquent, calcule et affiche les instructions par cycle dans la colonne la plus à droite. Vous pouvez observer un comportement similaire en ce qui concerne les échecs de branchements en pourcentage de tous les branchements, puisque les deux événements sont comptés par défaut.

18.4. ATTACHER LE STATUT DE PERF À UN PROCESSUS EN COURS D'EXÉCUTION

Vous pouvez attacher **perf stat** à un processus en cours d'exécution. Cela demandera à **perf stat** de compter les occurrences d'événements uniquement dans les processus spécifiés pendant l'exécution d'une commande.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.

Procédure

- Attachez **perf stat** à un processus en cours :

  ```
  $ perf stat -p ID1,ID2 sleep seconds
  ```

  L'exemple précédent comptabilise les événements dans les processus avec les identifiants ***ID1*** et ***ID2*** pendant une période de ***seconds*** secondes, comme indiqué à l'aide de la commande **sleep**.

Ressources supplémentaires

- **perf-stat(1)** page de manuel

CHAPITRE 19. ENREGISTREMENT ET ANALYSE DES PROFILS DE PERFORMANCE AVEC PERF

L'outil **perf** vous permet d'enregistrer des données sur les performances et de les analyser ultérieurement.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.

19.1. L'OBJECTIF DE LA FICHE DE PERF

La commande **perf record** échantillonne les données de performance et les stocke dans un fichier, **perf.data**, qui peut être lu et visualisé avec d'autres commandes **perf**. **perf.data** est généré dans le répertoire courant et peut être consulté ultérieurement, éventuellement sur une autre machine.

Si vous ne spécifiez pas de commande pour que **perf record** enregistre, il enregistrera jusqu'à ce que vous arrêtiez manuellement le processus en appuyant sur **Ctrl C**. Vous pouvez attacher **perf record** à des processus spécifiques en passant l'option **-p** suivie d'un ou plusieurs identifiants de processus. Vous pouvez exécuter **perf record** sans accès root, mais vous n'obtiendrez alors que des données de performance dans l'espace utilisateur. Dans le mode par défaut, **perf record** utilise les cycles du processeur comme événement d'échantillonnage et fonctionne en mode per-thread avec le mode inherit activé.

19.2. ENREGISTREMENT D'UN PROFIL DE PERFORMANCE SANS ACCÈS ROOT

Vous pouvez utiliser **perf record** sans accès root pour échantillonner et enregistrer des données de performance dans l'espace utilisateur uniquement.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.

Procédure

- Prélever des échantillons et enregistrer les données de performance :

  ```
  enregistrement de la perf command
  ```

  Remplacez ***command*** par la commande pendant laquelle vous souhaitez échantillonner les données. Si vous ne spécifiez pas de commande, **perf record** échantillonnera les données jusqu'à ce que vous l'arrêtiez manuellement en appuyant sur la touche **Ctrl**+**C**.

Ressources supplémentaires

- **perf-record(1)** page de manuel

19.3. ENREGISTREMENT D'UN PROFIL DE PERFORMANCE AVEC UN ACCÈS ROOT

Vous pouvez utiliser **perf record** avec un accès root pour échantillonner et enregistrer des données de performance à la fois dans l'espace utilisateur et dans l'espace noyau.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.
- Vous avez un accès root.

Procédure

- Prélever des échantillons et enregistrer les données de performance :

```
# perf record command
```

Remplacez ***command*** par la commande pendant laquelle vous souhaitez échantillonner les données. Si vous ne spécifiez pas de commande, **perf record** échantillonnera les données jusqu'à ce que vous l'arrêtiez manuellement en appuyant sur la touche **Ctrl**+**C**.

Ressources supplémentaires

- **perf-record(1)** page de manuel

19.4. ENREGISTREMENT D'UN PROFIL DE PERFORMANCE EN MODE PER-CPU

Vous pouvez utiliser **perf record** en mode per-CPU pour échantillonner et enregistrer des données de performance à la fois dans l'espace utilisateur et dans l'espace noyau simultanément pour tous les threads d'une unité centrale surveillée. Par défaut, le mode per-CPU surveille toutes les unités centrales en ligne.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.

Procédure

- Prélever des échantillons et enregistrer les données de performance :

```
# perf record -a command
```

Remplacez ***command*** par la commande pendant laquelle vous souhaitez échantillonner les données. Si vous ne spécifiez pas de commande, **perf record** échantillonnera les données jusqu'à ce que vous l'arrêtiez manuellement en appuyant sur la touche **Ctrl**+**C**.

Ressources supplémentaires

- **perf-record(1)** page de manuel

19.5. CAPTURER LES DONNÉES DU GRAPHIQUE D'APPEL AVEC L'ENREGISTREMENT DES PERFORMANCES

Vous pouvez configurer l'outil **perf record** de manière à ce qu'il enregistre la fonction qui appelle d'autres fonctions dans le profil de performance. Cela permet d'identifier un goulot d'étranglement si plusieurs processus appellent la même fonction.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.

Procédure

- L'option **--call-graph** permet d'échantillonner et d'enregistrer les données de performance :

```
$ perf record --call-graph method command
```

  - Remplacez ***command*** par la commande pendant laquelle vous souhaitez échantillonner les données. Si vous ne spécifiez pas de commande, **perf record** échantillonnera les données jusqu'à ce que vous l'arrêtiez manuellement en appuyant sur la touche **Ctrl**+**C**.
  - Remplacer *method* par l'une des méthodes de déroulement suivantes :

    fp

    Utilise la méthode du pointeur de cadre. En fonction de l'optimisation du compilateur, comme avec les binaires compilés avec l'option GCC **--fomit-frame-pointer**, ceci peut ne pas être capable de dérouler la pile.

    dwarf

    Utilise les informations du cadre d'appel DWARF pour dérouler la pile.

    lbr

    Utilise le dernier enregistrement de branche sur les processeurs Intel.

Ressources supplémentaires

- **perf-record(1)** page de manuel

19.6. ANALYSE DE PERF.DATA AVEC PERF REPORT

Vous pouvez utiliser **perf report** pour afficher et analyser un fichier **perf.data**.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.
- Il existe un fichier **perf.data** dans le répertoire actuel.
- Si le fichier **perf.data** a été créé avec un accès root, vous devez également exécuter **perf report** avec un accès root.

Procédure

- Affiche le contenu du fichier **perf.data** pour une analyse plus approfondie :

```
# perf report
```

  Cette commande affiche une sortie similaire à la suivante :

```
Samples: 2K of event 'cycles', Event count (approx.): 235462960
Overhead  Command          Shared Object                     Symbol
   2.36%  kswapd0          [kernel.kallsyms]                 [k] page_vma_mapped_walk
   2.13%  sssd_kcm         libc-2.28.so                      [.] memset_avx2_erms 2.13% perf
[kernel.kallsyms] [k] smp_call_function_single 1.53% gnome-shell libc-2.28.so [.]
strcmp_avx2
   1.17%  gnome-shell      libglib-2.0.so.0.5600.4           [.] g_hash_table_lookup
   0.93%  Xorg             libc-2.28.so                      [.] memmove_avx_unaligned_erms 0.89%
gnome-shell libgobject-2.0.so.0.5600.4 [.] g_object_unref 0.87% kswapd0 [kernel.kallsyms]
[k] page_referenced_one 0.86% gnome-shell libc-2.28.so [.] memmove_avx_unaligned_erms
   0.83%  Xorg             [kernel.kallsyms]                 [k] alloc_vmap_area
   0.63%  gnome-shell      libglib-2.0.so.0.5600.4           [.] g_slice_alloc
   0.53%  gnome-shell      libgirepository-1.0.so.1.0.0      [.] g_base_info_unref
   0.53%  gnome-shell      ld-2.28.so                        [.] _dl_find_dso_for_object
   0.49%  kswapd0          [kernel.kallsyms]                 [k] vma_interval_tree_iter_next
   0.48%  gnome-shell      libpthread-2.28.so                [.] pthread_getspecific 0.47% gnome-
shell libgirepository-1.0.so.1.0.0 [.] 0x0000000000013b1d 0.45% gnome-shell libglib-
2.0.so.0.5600.4 [.] g_slice_free1 0.45% gnome-shell libgobject-2.0.so.0.5600.4 [.]
g_type_check_instance_is_fundamentally_a 0.44% gnome-shell libc-2.28.so [.] malloc 0.41%
swapper [kernel.kallsyms] [k] apic_timer_interrupt 0.40% gnome-shell ld-2.28.so [.]
_dl_lookup_symbol_x 0.39% kswapd0 [kernel.kallsyms] [k]
raw_callee_save___pv_queued_spin_unlock
```

Ressources supplémentaires

- **perf-report(1)** page de manuel

19.7. INTERPRÉTATION DU RAPPORT DE PERF

Le tableau affiché par la commande **perf report** trie les données en plusieurs colonnes :

La colonne "Overhead" (frais généraux)

Indique le pourcentage de l'ensemble des échantillons collectés dans cette fonction particulière.

La colonne "Commande

Indique le processus à partir duquel les échantillons ont été prélevés.

La colonne "Objet partagé

Affiche le nom de l'image ELF d'où proviennent les échantillons (le nom [kernel.kallsyms] est utilisé lorsque les échantillons proviennent du noyau).

La colonne "Symbole

Affiche le nom ou le symbole de la fonction.

En mode par défaut, les fonctions sont triées par ordre décroissant, celles dont les frais généraux sont les plus élevés étant affichées en premier.

19.8. GÉNÉRER UN FICHIER PERF.DATA LISIBLE SUR UN AUTRE APPAREIL

Vous pouvez utiliser l'outil **perf** pour enregistrer des données de performance dans un fichier **perf.data** qui sera analysé sur un autre appareil.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.
- Le paquetage kernel **debuginfo** est installé. Pour plus d'informations, voir Obtenir des paquets d'informations de débogage pour une application ou une bibliothèque à l'aide de GDB.

Procédure

1. Capturez les données de performance que vous souhaitez étudier plus en détail :

```
# perf record -a --call-graph fp sleep seconds
```

Cet exemple génère une adresse **perf.data** sur l'ensemble du système pendant une période de quelques secondes, comme indiqué par la commande ***seconds*** secondes, comme indiqué par l'utilisation de la commande **sleep**. Il capture également les données du graphique d'appel à l'aide de la méthode du pointeur de trame.

2. Générer un fichier d'archive contenant les symboles de débogage des données enregistrées :

```
# perf archive
```

Verification steps

- Vérifiez que le fichier d'archive a été généré dans votre répertoire actif actuel :

```
# ls perf.data*
```

La sortie affichera tous les fichiers de votre répertoire actuel qui commencent par **perf.data**. Le fichier d'archive sera nommé soit :

```
perf.data.tar.gz
```

ou

```
perf.data.tar.bz2
```

Ressources supplémentaires

- Enregistrement et analyse des profils de performance avec perf
- Capturer les données du graphique d'appel avec l'enregistrement des performances

19.9. ANALYSE D'UN FICHIER PERF.DATA CRÉÉ SUR UN AUTRE APPAREIL

Vous pouvez utiliser l'outil **perf** pour analyser un fichier **perf.data** généré sur un autre appareil.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.
- Un fichier **perf.data** et un fichier d'archive associé, générés sur un autre appareil, sont présents sur l'appareil actuellement utilisé.

Procédure

1. Copiez le fichier **perf.data** et le fichier d'archive dans votre répertoire actif actuel.
2. Extraire le fichier d'archive dans **~/.debug**:

```
# mkdir -p ~/.debug
# tar xf perf.data.tar.bz2 -C ~/.debug
```

NOTE

Le fichier d'archive peut également être nommé ***perf.data.tar.gz***.

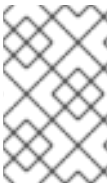
3. Ouvrez le fichier **perf.data** pour une analyse plus approfondie :

```
# perf report
```

19.10. POURQUOI PERF AFFICHE-T-IL CERTAINS NOMS DE FONCTIONS COMME DES ADRESSES DE FONCTIONS BRUTES ?

Pour les fonctions du noyau, **perf** utilise les informations du fichier **/proc/kallsyms** pour faire correspondre les échantillons à leurs noms de fonction ou symboles respectifs. Pour les fonctions exécutées dans l'espace utilisateur, cependant, vous pouvez voir des adresses de fonctions brutes parce que le binaire est dépouillé.

Le paquet **debuginfo** de l'exécutable doit être installé ou, si l'exécutable est une application développée localement, l'application doit être compilée avec les informations de débogage activées (l'option **-g** dans GCC) pour afficher les noms de fonction ou les symboles dans une telle situation.



NOTE

Il n'est pas nécessaire de réexécuter la commande **perf record** après avoir installé la commande **debuginfo** associée à un exécutable. Il suffit de réexécuter la commande **perf report**.

Ressources complémentaires

- Activation du débogage avec les informations de débogage

19.11. ACTIVATION DES DÉPÔTS DE DÉBOGAGE ET DE SOURCES

Une installation standard de Red Hat Enterprise Linux n'active pas les référentiels de débogage et de sources. Ces référentiels contiennent des informations nécessaires pour déboguer les composants du système et mesurer leurs performances.

Procédure

- Activez les canaux du paquet d'informations de source et de débogage : La partie **$(uname -i)** est automatiquement remplacée par une valeur correspondant à l'architecture de votre système :

Nom de l'architecture	Valeur
64-bit Intel et AMD	x86_64
aRM 64 bits	aarch64
IBM POWER	ppc64le
iBM Z 64 bits	s390x

19.12. OBTENIR LES PAQUETS D'INFORMATIONS DE DÉBOGAGE POUR UNE APPLICATION OU UNE BIBLIOTHÈQUE À L'AIDE DE GDB

Les informations de débogage sont nécessaires pour déboguer le code. Pour le code installé à partir d'un paquetage, le débogueur GNU (GDB) reconnaît automatiquement les informations de débogage manquantes, résout le nom du paquetage et fournit des conseils concrets sur la manière d'obtenir le paquetage.

Conditions préalables

- L'application ou la bibliothèque que vous souhaitez déboguer doit être installée sur le système.
- GDB et l'outil **debuginfo-install** doivent être installés sur le système. Pour plus de détails, voir Configuration pour le débogage d'applications .
- Les dépôts fournissant les paquets **debuginfo** et **debugsource** doivent être configurés et activés sur le système. Pour plus de détails, voir Activation des référentiels de débogage et de sources.

Procédure

1. Lancez GDB attaché à l'application ou à la bibliothèque que vous souhaitez déboguer. GDB reconnaît automatiquement les informations de débogage manquantes et suggère une commande à exécuter.

```
$ gdb -q /bin/ls
Reading symbols from /bin/ls...Reading symbols from .gnu_debugdata for /usr/bin/ls...(no debugging symbols found)...done.
(no debugging symbols found)...done.
Missing separate debuginfos, use: dnf debuginfo-install coreutils-8.30-6.el8.x86_64
(gdb)
```

2. Quittez GDB : tapez **q** et confirmez avec **Enter**.

```
(gdb) q
```

3. Exécutez la commande suggérée par GDB pour installer les paquets **debuginfo** requis :

```
# dnf debuginfo-install coreutils-8.30-6.el8.x86_64
```

L'outil de gestion des paquets **dnf** fournit un résumé des changements, demande une confirmation et, une fois que vous avez confirmé, télécharge et installe tous les fichiers nécessaires.

4. Si GDB n'est pas en mesure de suggérer le paquet **debuginfo**, suivez la procédure décrite dans Obtenir manuellement les paquets debuginfo pour une application ou une bibliothèque .

Ressources supplémentaires

- Guide d'utilisation de Red Hat Developer Toolset, section Installation des informations de débogage
- Comment puis-je télécharger ou installer des paquets debuginfo pour les systèmes RHEL ? - Solution de la base de connaissances de Red Hat

CHAPITRE 20. ÉTUDIER LES UNITÉS CENTRALES OCCUPÉES À L'AIDE DE LA PERF

Lorsque vous étudiez les problèmes de performance d'un système, vous pouvez utiliser l'outil **perf** pour identifier et surveiller les unités centrales les plus occupées afin de concentrer vos efforts.

20.1. AFFICHAGE DES ÉVÉNEMENTS DE L'UNITÉ CENTRALE QUI ONT ÉTÉ COMPTABILISÉS AVEC PERF STAT

Vous pouvez utiliser **perf stat** pour afficher les événements CPU qui ont été comptés en désactivant l'agrégation du comptage CPU. Vous devez compter les événements en mode système en utilisant l'indicateur **-a** afin d'utiliser cette fonctionnalité.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.

Procédure

- Compter les événements avec l'agrégation du nombre de CPU désactivée :

```
# perf stat -a -A sleep seconds
```

L'exemple précédent affiche les décomptes d'un ensemble par défaut d'événements matériels et logiciels courants enregistrés sur une période de ***seconds*** secondes, comme l'indique la commande **sleep**, sur chaque unité centrale par ordre croissant, en commençant par **CPU0**. Il peut donc être utile de spécifier un événement tel que les cycles :

```
# perf stat -a -A -e cycles sleep seconds
```

20.2. AFFICHAGE DE L'UNITÉ CENTRALE SUR LAQUELLE LES ÉCHANTILLONS ONT ÉTÉ PRÉLEVÉS AVEC LE RAPPORT DE PERF

La commande **perf record** échantillonne les données de performance et les stocke dans un fichier **perf.data** qui peut être lu avec la commande **perf report**. La commande **perf record** enregistre toujours l'unité centrale sur laquelle les échantillons ont été prélevés. Vous pouvez configurer **perf report** pour qu'il affiche ces informations.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.
- Un fichier **perf.data** a été créé avec **perf record** dans le répertoire actuel. Si le fichier **perf.data** a été créé avec un accès root, vous devez également exécuter **perf report** avec un accès root.

Procédure

- Affiche le contenu du fichier **perf.data** pour une analyse plus approfondie tout en le triant par CPU :

```
# perf report --sort cpu
```

- Vous pouvez trier par unité centrale et par commande pour afficher des informations plus détaillées sur l'utilisation du temps de l'unité centrale :

```
# perf report --sort cpu,comm
```

Cet exemple dresse la liste des commandes de toutes les unités centrales surveillées, par ordre décroissant d'utilisation des frais généraux, et identifie l'unité centrale sur laquelle la commande a été exécutée.

Ressources supplémentaires

- Enregistrement et analyse des profils de performance avec perf

20.3. AFFICHAGE D'UNITÉS CENTRALES SPÉCIFIQUES LORS DU PROFILAGE AVEC PERF TOP

Vous pouvez configurer **perf top** pour qu'il affiche des CPU spécifiques et leur utilisation relative lors du profilage de votre système en temps réel.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.

Procédure

- Lancez l'interface **perf top** tout en triant par CPU :

```
# perf top --sort cpu
```

Cet exemple dresse la liste des unités centrales et de leurs frais généraux respectifs dans l'ordre décroissant de l'utilisation des frais généraux en temps réel.

- Vous pouvez trier par unité centrale et par commande pour obtenir des informations plus détaillées sur l'utilisation du temps de l'unité centrale :

```
# perf top --sort cpu,comm
```

Cet exemple dresse la liste des commandes par ordre décroissant d'utilisation des frais généraux et identifie le processeur sur lequel la commande a été exécutée en temps réel.

20.4. SURVEILLANCE D'UNITÉS CENTRALES SPÉCIFIQUES GRÂCE À L'ENREGISTREMENT ET AU RAPPORT DE PERFORMANCE

Vous pouvez configurer **perf record** pour qu'il n'échantillonne que les processeurs spécifiques qui vous intéressent et analyser le fichier **perf.data** généré avec **perf report** pour une analyse plus approfondie.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.

Procédure

1. Échantillonner et enregistrer les données de performance dans les unités centrales spécifiques, en générant un fichier **perf.data**:

   - En utilisant une liste de CPU séparés par des virgules :

     ```
     # perf record -C 0,1 sleep seconds
     ```

     L'exemple précédent échantillonne et enregistre des données dans les unités centrales 0 et 1 pendant une période de ***seconds*** secondes, comme indiqué par l'utilisation de la commande **sleep**.

   - Utilisation d'une gamme d'unités centrales :

     ```
     # perf record -C 0-2 sleep seconds
     ```

     L'exemple précédent échantillonne et enregistre des données dans toutes les unités centrales, de l'unité centrale 0 à l'unité centrale 2, pendant une période de ***seconds*** secondes, comme indiqué par l'utilisation de la commande **sleep**.

2. Affiche le contenu du fichier **perf.data** pour une analyse plus approfondie :

   ```
   # perf report
   ```

   Cet exemple affiche le contenu de **perf.data**. Si vous surveillez plusieurs unités centrales et que vous souhaitez savoir sur quelle unité centrale les données ont été échantillonnées, reportez-vous à la section Afficher les échantillons de l'unité centrale avec le rapport perf.

CHAPITRE 21. CONTRÔLER LA PERFORMANCE DES APPLICATIONS AVEC PERF

Vous pouvez utiliser l'outil **perf** pour surveiller et analyser les performances des applications.

21.1. ATTACHER UNE FICHE DE PERF À UN PROCESSUS EN COURS

Vous pouvez attacher **perf record** à un processus en cours d'exécution. Cela demandera à **perf record** de n'échantillonner et d'enregistrer les données de performance que dans les processus spécifiés.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.

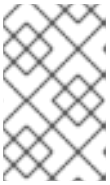
Procédure

- Attachez **perf record** à un processus en cours :

```
$ perf record -p ID1,ID2 sleep seconds
```

L'exemple précédent échantillonne et enregistre les données de performance des processus avec les identifiants de processus ***ID1*** et ***ID2*** pendant une période de ***seconds*** secondes, comme indiqué à l'aide de la commande **sleep**. Vous pouvez également configurer **perf** pour qu'il enregistre des événements dans des threads spécifiques :

```
$ perf record -t ID1,ID2 sleep seconds
```



NOTE

Lorsque vous utilisez l'option **-t** et que vous stipulez des identifiants de threads, **perf** désactive l'héritage par défaut. Vous pouvez activer l'héritage en ajoutant l'option **--inherit**.

21.2. CAPTURER LES DONNÉES DU GRAPHIQUE D'APPEL AVEC L'ENREGISTREMENT DES PERFORMANCES

Vous pouvez configurer l'outil **perf record** de manière à ce qu'il enregistre la fonction qui appelle d'autres fonctions dans le profil de performance. Cela permet d'identifier un goulot d'étranglement si plusieurs processus appellent la même fonction.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.

Procédure

- L'option **--call-graph** permet d'échantillonner et d'enregistrer les données de performance :

```
$ perf record --call-graph method command
```

- Remplacez ***command*** par la commande pendant laquelle vous souhaitez échantillonner les données. Si vous ne spécifiez pas de commande, **perf record** échantillonnera les données jusqu'à ce que vous l'arrêtiez manuellement en appuyant sur la touche **Ctrl**+**C**.
- Remplacer *method* par l'une des méthodes de déroulement suivantes :

  fp

  Utilise la méthode du pointeur de cadre. En fonction de l'optimisation du compilateur, comme avec les binaires compilés avec l'option GCC **--fomit-frame-pointer**, ceci peut ne pas être capable de dérouler la pile.

  dwarf

  Utilise les informations du cadre d'appel DWARF pour dérouler la pile.

  lbr

  Utilise le dernier enregistrement de branche sur les processeurs Intel.

Ressources supplémentaires

- **perf-record(1)** page de manuel

21.3. ANALYSE DE PERF.DATA AVEC PERF REPORT

Vous pouvez utiliser **perf report** pour afficher et analyser un fichier **perf.data**.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.
- Il existe un fichier **perf.data** dans le répertoire actuel.
- Si le fichier **perf.data** a été créé avec un accès root, vous devez également exécuter **perf report** avec un accès root.

Procédure

- Affiche le contenu du fichier **perf.data** pour une analyse plus approfondie :

```
# perf report
```

Cette commande affiche une sortie similaire à la suivante :

```
Samples: 2K of event 'cycles', Event count (approx.): 235462960
Overhead  Command          Shared Object                Symbol
  2.36%  kswapd0          [kernel.kallsyms]            [k] page_vma_mapped_walk
  2.13%  sssd_kcm         libc-2.28.so                 [.] memset_avx2_erms 2.13% perf
[kernel.kallsyms] [k] smp_call_function_single 1.53% gnome-shell libc-2.28.so [.]
strcmp_avx2
  1.17%  gnome-shell      libglib-2.0.so.0.5600.4      [.] g_hash_table_lookup
  0.93%  Xorg             libc-2.28.so                 [.] memmove_avx_unaligned_erms 0.89%
gnome-shell libgobject-2.0.so.0.5600.4 [.] g_object_unref 0.87% kswapd0 [kernel.kallsyms]
[k] page_referenced_one 0.86% gnome-shell libc-2.28.so [.] memmove_avx_unaligned_erms
  0.83%  Xorg             [kernel.kallsyms]            [k] alloc_vmap_area
  0.63%  gnome-shell      libglib-2.0.so.0.5600.4      [.] g_slice_alloc
  0.53%  gnome-shell      libgirepository-1.0.so.1.0.0 [.] g_base_info_unref
```

```
   0.53%  gnome-shell      ld-2.28.so                [.] _dl_find_dso_for_object
   0.49%  kswapd0          [kernel.kallsyms]         [k] vma_interval_tree_iter_next
   0.48%  gnome-shell      libpthread-2.28.so        [.] pthread_getspecific 0.47% gnome-
shell libgirepository-1.0.so.1.0.0 [.] 0x0000000000013b1d 0.45% gnome-shell libglib-
2.0.so.0.5600.4 [.] g_slice_free1 0.45% gnome-shell libgobject-2.0.so.0.5600.4 [.]
g_type_check_instance_is_fundamentally_a 0.44% gnome-shell libc-2.28.so [.] malloc 0.41%
swapper [kernel.kallsyms] [k] apic_timer_interrupt 0.40% gnome-shell ld-2.28.so [.]
_dl_lookup_symbol_x 0.39% kswapd0 [kernel.kallsyms] [k]
raw_callee_save___pv_queued_spin_unlock
```

Ressources supplémentaires

- **perf-report(1)** page de manuel

CHAPITRE 22. CRÉER DES ROBES DE CHAMBRE AVEC PERF

22.1. CRÉER DES ROBES DE CHAMBRE AU NIVEAU DE LA FONCTION AVEC PERF

Vous pouvez utiliser l'outil **perf** pour créer des points de référence dynamiques à des endroits arbitraires d'un processus ou d'une application. Ces tracepoints peuvent ensuite être utilisés conjointement avec d'autres outils **perf** tels que **perf stat** et **perf record** pour mieux comprendre le comportement du processus ou de l'application.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.

Procédure

1. Créez la sonde ascendante dans le processus ou l'application que vous souhaitez surveiller à un endroit intéressant du processus ou de l'application :

```
# perf probe -x /path/to/executable -a function
Added new event:
  probe_executable:function   (on function in /path/to/executable)

You can now use it in all perf tools, such as:

        perf record -e probe_executable:function -aR sleep 1
```

Ressources supplémentaires

- **perf-probe** page de manuel
- Enregistrement et analyse des profils de performance avec perf
- Compter les événements pendant l'exécution d'un processus avec perf stat

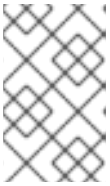
22.2. CRÉER DES UPROBES SUR DES LIGNES DANS UNE FONCTION AVEC PERF

Ces tracepoints peuvent ensuite être utilisés en conjonction avec d'autres outils **perf** tels que **perf stat** et **perf record** afin de mieux comprendre le processus ou le comportement des applications.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.
- Vous avez obtenu les symboles de débogage pour votre exécutable :

```
# objdump -t ./your_executable | head
```



NOTE

Pour ce faire, le paquet **debuginfo** de l'exécutable doit être installé ou, si l'exécutable est une application développée localement, l'application doit être compilée avec des informations de débogage, l'option **-g** dans GCC.

Procédure

1. Visualisez les lignes de fonction où vous pouvez placer une robe montante :

```
$ perf probe -x ./your_executable -L main
```

La sortie de cette commande ressemble à ce qui suit :

```
<main@/home/user/my_executable:0>
            0  int main(int argc, const char **argv)
            1  {
                    int err;
                    const char *cmd;
                    char sbuf[STRERR_BUFSIZE];

                    /* libsubcmd init */
            7       exec_cmd_init("perf", PREFIX, PERF_EXEC_PATH,
EXEC_PATH_ENVIRONMENT);
            8       pager_init(PERF_PAGER_ENVIRONMENT);
```

2. Créer la robe montante pour la ligne de fonction souhaitée :

```
# perf probe -x ./my_executable main:8
Added new event:
        probe_my_executable:main_L8    (on main:8 in /home/user/my_executable)

      You can now use it in all perf tools, such as:

            perf record -e probe_my_executable:main_L8 -aR sleep 1
```

22.3. SORTIE D'UN SCRIPT PERF DES DONNÉES ENREGISTRÉES AU COURS DES SONDES ASCENDANTES

Une méthode courante pour analyser les données collectées à l'aide de sondes ascendantes consiste à utiliser la commande **perf script** pour lire un fichier **perf.data** et afficher une trace détaillée de la charge de travail enregistrée.

In the perf script example output: * A uprobe is added to the function **isprime()** in a program called **my_prog** * **a** is a function argument added to the uprobe. Alternatively, **a** could be an arbitrary variable visible in the code scope of where you add your uprobe:

```
# perf script
   my_prog  1367 [007] 10802159.906593: probe_my_prog:isprime: (400551) a=2
   my_prog  1367 [007] 10802159.906623: probe_my_prog:isprime: (400551) a=3
   my_prog  1367 [007] 10802159.906625: probe_my_prog:isprime: (400551) a=4
   my_prog  1367 [007] 10802159.906627: probe_my_prog:isprime: (400551) a=5
   my_prog  1367 [007] 10802159.906629: probe_my_prog:isprime: (400551) a=6
   my_prog  1367 [007] 10802159.906631: probe_my_prog:isprime: (400551) a=7
```

```
my_prog  1367 [007] 10802159.906633: probe_my_prog:isprime: (400551) a=13
my_prog  1367 [007] 10802159.906635: probe_my_prog:isprime: (400551) a=17
my_prog  1367 [007] 10802159.906637: probe_my_prog:isprime: (400551) a=19
```

CHAPITRE 23. PROFILER LES ACCÈS À LA MÉMOIRE AVEC PERF MEM

Vous pouvez utiliser la commande **perf mem** pour échantillonner les accès à la mémoire de votre système.

23.1. L'OBJECTIF DE PERF MEM

La sous-commande **mem** de l'outil **perf** permet d'échantillonner les accès à la mémoire (chargements et stockages). La commande **perf mem** fournit des informations sur la latence de la mémoire, les types d'accès à la mémoire, les fonctions à l'origine des hits et des misses de la mémoire cache et, en enregistrant le symbole des données, les emplacements de la mémoire où ces hits et misses se produisent.

23.2. ÉCHANTILLONNAGE DE L'ACCÈS À LA MÉMOIRE AVEC PERF MEM

Cette procédure décrit comment utiliser la commande **perf mem** pour échantillonner les accès à la mémoire sur votre système. La commande prend les mêmes options que **perf record** et **perf report** ainsi que certaines options exclusives à la sous-commande **mem**. Les données enregistrées sont stockées dans un fichier **perf.data** dans le répertoire actuel en vue d'une analyse ultérieure.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.

Procédure

1. Échantillonner les accès à la mémoire :

   ```
   # perf mem record -a sleep seconds
   ```

   Cet exemple échantillonne les accès à la mémoire de toutes les unités centrales pendant une période de *seconds* secondes, comme indiqué par la commande **sleep**. Vous pouvez remplacer la commande **sleep** par n'importe quelle commande au cours de laquelle vous souhaitez échantillonner les données d'accès à la mémoire. Par défaut, **perf mem** échantillonne à la fois les chargements et les stockages de mémoire. Vous pouvez sélectionner une seule opération de mémoire en utilisant l'option **-t** et en spécifiant "load" ou "store" entre **perf mem** et **record**. Pour les chargements, les informations sur le niveau de hiérarchie de la mémoire, les accès à la mémoire TLB, les interrogations du bus et les verrouillages de la mémoire sont capturées.

2. Ouvrir le fichier **perf.data** pour analyse :

   ```
   # perf mem report
   ```

   Si vous avez utilisé les commandes de l'exemple, la sortie est la suivante :

   ```
   Available samples
   35k cpu/mem-loads,ldlat=30/P
   54k cpu/mem-stores/P
   ```

   La ligne **cpu/mem-loads,ldlat=30/P** indique les données collectées sur les chargements de

mémoire et la ligne **cpu/mem-stores/P** indique les données collectées sur les stockages de mémoire. Mettez en surbrillance la catégorie qui vous intéresse et appuyez sur la touche **Entrée** pour afficher les données :

```
Samples: 35K of event 'cpu/mem-loads,ldlat=30/P', Event count (approx.): 4067062
Overhead       Samples  Local Weight  Memory access            Symbol
Shared Object                Data Symbol                                  Data Object
Snoop       TLB access            Locked
  0.07%            29  98           L1 or L1 hit            [.] 0x000000000000a255
libspeexdsp.so.1.5.0         [.] 0x00007f697a3cd0f0                       anon
None        L1 or L2 hit          No
  0.06%            26  97           L1 or L1 hit            [.] 0x000000000000a255
libspeexdsp.so.1.5.0         [.] 0x00007f697a3cd0f0                       anon
None        L1 or L2 hit          No
  0.06%            25  96           L1 or L1 hit            [.] 0x000000000000a255
libspeexdsp.so.1.5.0         [.] 0x00007f697a3cd0f0                       anon
None        L1 or L2 hit          No
  0.06%             1  2325         Uncached or N/A hit     [k] pci_azx_readl
[kernel.kallsyms]            [k] 0xffffb092c06e9084                       [kernel.kallsyms]
None        L1 or L2 hit          No
  0.06%             1  2247         Uncached or N/A hit     [k] pci_azx_readl
[kernel.kallsyms]            [k] 0xffffb092c06e8164                       [kernel.kallsyms]
None        L1 or L2 hit          No
  0.05%             1  2166         L1 or L1 hit            [.] 0x00000000038140d6
libxul.so                    [.] 0x00007ffd7b84b4a8                       [stack]
None        L1 or L2 hit          No
  0.05%             1  2117         Uncached or N/A hit     [k] check_for_unclaimed_mmio
[kernel.kallsyms]            [k] 0xffffb092c1842300                       [kernel.kallsyms]
None        L1 or L2 hit          No
  0.05%            22  95           L1 or L1 hit            [.] 0x000000000000a255
libspeexdsp.so.1.5.0         [.] 0x00007f697a3cd0f0                       anon
None        L1 or L2 hit          No
  0.05%             1  1898         L1 or L1 hit            [.] 0x0000000002a30e07
libxul.so                    [.] 0x00007f610422e0e0                       anon
None        L1 or L2 hit          No
  0.05%             1  1878         Uncached or N/A hit     [k] pci_azx_readl
[kernel.kallsyms]            [k] 0xffffb092c06e8164                       [kernel.kallsyms]
None        L2 miss               No
  0.04%            18  94           L1 or L1 hit            [.] 0x000000000000a255
libspeexdsp.so.1.5.0         [.] 0x00007f697a3cd0f0                       anon
None        L1 or L2 hit          No
  0.04%             1  1593         Local RAM or RAM hit    [.] 0x00000000026f907d
libxul.so                    [.] 0x00007f3336d50a80                       anon
Hit         L2 miss               No
  0.03%             1  1399         L1 or L1 hit            [.] 0x00000000037cb5f1
libxul.so                    [.] 0x00007fbe81ef5d78                       libxul.so
None        L1 or L2 hit          No
  0.03%             1  1229         LFB or LFB hit          [.] 0x0000000002962aad
libxul.so                    [.] 0x00007fb6f1be2b28                       anon
None        L2 miss               No
  0.03%             1  1202         LFB or LFB hit          [.] __pthread_mutex_lock
libpthread-2.29.so           [.] 0x00007fb75583ef20                       anon
None        L1 or L2 hit          No
  0.03%             1  1193         Uncached or N/A hit     [k] pci_azx_readl
[kernel.kallsyms]            [k] 0xffffb092c06e9164                       [kernel.kallsyms]
None        L2 miss               No
```

```
  0.03%            1  1191          L1 or L1 hit              [k] azx_get_delay_from_lpib
[kernel.kallsyms]           [k] 0xffffb092ca7efcf0                                  [kernel.kallsyms]
None         L1 or L2 hit          No
```

Vous pouvez également trier vos résultats pour étudier différents aspects intéressants lors de l'affichage des données. Par exemple, pour trier les données sur les charges de mémoire par type d'accès à la mémoire survenant au cours de la période d'échantillonnage, dans l'ordre décroissant des frais généraux qu'ils représentent :

```
# perf mem -t load report --sort=mem
```

Par exemple, le résultat peut être :

```
Samples: 35K of event 'cpu/mem-loads,ldlat=30/P', Event count (approx.): 40670
Overhead       Samples  Memory access
  31.53%          9725  LFB or LFB hit
  29.70%         12201  L1 or L1 hit
  23.03%          9725  L3 or L3 hit
  12.91%          2316  Local RAM or RAM hit
   2.37%           743  L2 or L2 hit
   0.34%             9  Uncached or N/A hit
   0.10%            69  I/O or N/A hit
   0.02%           825  L3 miss
```

Ressources supplémentaires

- **perf-mem(1)** page de manuel.

23.3. INTERPRÉTATION DU RAPPORT DE PERF MEM

Le tableau affiché en exécutant la commande **perf mem report** sans aucun modificateur trie les données en plusieurs colonnes :

La colonne "Overhead" (frais généraux)

Indique le pourcentage de l'ensemble des échantillons collectés dans cette fonction particulière.

La colonne "Échantillons

Affiche le nombre d'échantillons pris en compte par cette ligne.

La colonne "Poids local

Affiche la latence d'accès en cycles de cœur de processeur.

La colonne "Accès à la mémoire

Affiche le type d'accès à la mémoire qui s'est produit.

La colonne "Symbole

Affiche le nom ou le symbole de la fonction.

La colonne "Objet partagé

Affiche le nom de l'image ELF d'où proviennent les échantillons (le nom [kernel.kallsyms] est utilisé lorsque les échantillons proviennent du noyau).

La colonne "Symbole de données

Affiche l'adresse de l'emplacement de mémoire ciblé par la ligne.

IMPORTANT

Souvent, en raison de l'allocation dynamique de la mémoire ou de l'accès à la mémoire de la pile, la colonne "Symbole de données" affichera une adresse brute.

La colonne "Snoop"

Affiche les transactions du bus.

La colonne "Accès TLB

Affiche les accès à la mémoire TLB.

La colonne "Verrouillé

Indique si une fonction était ou non verrouillée en mémoire.

En mode par défaut, les fonctions sont triées par ordre décroissant, celles dont les frais généraux sont les plus élevés étant affichées en premier.

CHAPITRE 24. DÉTECTER LES FAUX PARTAGES

Un faux partage se produit lorsqu'un cœur de processeur d'un système SMP (Symmetric Multi Processing) modifie des éléments de données sur la même ligne de cache utilisée par d'autres processeurs pour accéder à d'autres éléments de données qui ne sont pas partagés entre les processeurs.

Cette modification initiale exige que les autres processeurs utilisant la ligne de cache invalident leur copie et en demandent une mise à jour, bien que les processeurs n'aient pas besoin d'une version mise à jour de l'élément de données modifié, ni même nécessairement accès à cette version.

Vous pouvez utiliser la commande **perf c2c** pour détecter les faux partages.

24.1. L'OBJECTIF DE PERF C2C

La sous-commande **c2c** de l'outil **perf** permet l'analyse Cache-to-Cache (C2C) des données partagées. Vous pouvez utiliser la commande **perf c2c** pour inspecter la contention des lignes de cache afin de détecter le vrai et le faux partage.

La contention des lignes de cache se produit lorsqu'un cœur de processeur d'un système SMP (Symmetric Multi Processing) modifie des éléments de données sur la même ligne de cache qui est utilisée par d'autres processeurs. Tous les autres processeurs utilisant cette ligne de cache doivent alors invalider leur copie et demander une mise à jour. Cela peut entraîner une dégradation des performances.

La commande **perf c2c** fournit les informations suivantes :

- Lignes de cache pour lesquelles une contention a été détectée
- Processus de lecture et d'écriture des données
- Instructions à l'origine de la contestation
- Les nœuds NUMA (Non-Uniform Memory Access) impliqués dans la contestation

24.2. DÉTECTION DE LA CONTENTION DES LIGNES DE CACHE AVEC PERF C2C

Utilisez la commande **perf c2c** pour détecter la contention des lignes de cache dans un système.

La commande **perf c2c** prend en charge les mêmes options que **perf record** ainsi que certaines options exclusives à la sous-commande **c2c**. Les données enregistrées sont stockées dans un fichier **perf.data** dans le répertoire courant en vue d'une analyse ultérieure.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé. Pour plus d'informations, voir l' installation de perf.

Procédure

- Utilisez **perf c2c** pour détecter la contention des lignes de cache :

```
# perf c2c record -a sleep seconds
```

  Cet exemple échantillonne et enregistre les données de contention des lignes de cache sur tous

les processeurs pendant une période de ***seconds*** comme indiqué par la commande **sleep**. Vous pouvez remplacer la commande **sleep** par n'importe quelle commande pour laquelle vous souhaitez collecter des données sur la contention des lignes de cache.

Ressources supplémentaires

- **perf-c2c(1)** page de manuel

24.3. VISUALISATION D'UN FICHIER PERF.DATA ENREGISTRÉ AVEC PERF C2C RECORD

Cette procédure décrit comment visualiser le fichier **perf.data**, qui est enregistré à l'aide de la commande **perf c2c**.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé. Pour plus d'informations, voir Installation de perf.
- Un fichier **perf.data** enregistré à l'aide de la commande **perf c2c** est disponible dans le répertoire actuel. Pour plus d'informations, voir Détection de la contention des lignes de cache avec perf c2c.

Procédure

1. Ouvrez le fichier **perf.data** pour une analyse plus approfondie :

```
# perf c2c report --stdio
```

Cette commande permet de visualiser le fichier **perf.data** en plusieurs graphiques dans le terminal :

```
=================================================
        Trace Event Information
=================================================
 Total records                     :     329219
 Locked Load/Store Operations      :      14654
 Load Operations                   :      69679
 Loads - uncacheable               :          0
 Loads - IO                        :          0
 Loads - Miss                      :       3972
 Loads - no mapping                :          0
 Load Fill Buffer Hit              :      11958
 Load L1D hit                      :      17235
 Load L2D hit                      :         21
 Load LLC hit                      :      14219
 Load Local HITM                   :       3402
 Load Remote HITM                  :      12757
 Load Remote HIT                   :       5295
 Load Local DRAM                   :        976
 Load Remote DRAM                  :       3246
 Load MESI State Exclusive         :       4222
 Load MESI State Shared            :          0
 Load LLC Misses                   :      22274
 LLC Misses to Local DRAM          :       4.4%
```

```
 LLC Misses to Remote DRAM        :       14.6%
 LLC Misses to Remote cache (HIT)  :       23.8%
 LLC Misses to Remote cache (HITM) :       57.3%
 Store Operations                 :     259539
 Store - uncacheable              :          0
 Store - no mapping               :         11
 Store L1D Hit                    :     256696
 Store L1D Miss                   :       2832
 No Page Map Rejects              :       2376
 Unable to parse data source      :          1

=================================================
   Global Shared Cache Line Event Information
=================================================
 Total Shared Cache Lines         :         55
 Load HITs on shared lines        :      55454
 Fill Buffer Hits on shared lines  :      10635
 L1D hits on shared lines         :      16415
 L2D hits on shared lines         :          0
 LLC hits on shared lines         :       8501
 Locked Access on shared lines    :      14351
 Store HITs on shared lines       :     109953
 Store L1D hits on shared lines   :     109449
 Total Merged records             :     126112

=================================================
               c2c details
=================================================
 Events                           : cpu/mem-loads,ldlat=30/P
                                  : cpu/mem-stores/P
 Cachelines sort on               : Remote HITMs
 Cacheline data groupping         : offset,pid,iaddr

=================================================
    Shared Data Cache Line Table
=================================================
#
#                           Total      Rmt  ----- LLC Load Hitm -----  ---- Store Reference ----  --- Load
Dram ----      LLC    Total  ----- Core Load Hit -----  -- LLC Load Hit --
# Index           Cacheline  records     Hitm    Total      Lcl      Rmt    Total    L1Hit   L1Miss
Lcl       Rmt  Ld Miss    Loads       FB       L1       L2       Llc       Rmt
# .....  ..................  .......  .......  .......  .......  .......  .......  .......  .......  ........  ........  .......  .......  .......
.......  .......  ........  ........
#
     0           0x602180   149904   77.09%    12103     2269     9834   109504   109036      468
727      2657    13747    40400     5355    16154        0     2875       529
     1           0x602100    12128   22.20%     3951     1119     2832        0        0        0       65
200     3749    12128     5096      108        0     2056       652
     2  0xffff883ffb6a7e80      260    0.09%       15        3       12      161      161        0        1
1       15       99       25       50        0        6        1
     3  0xffffffff81aec000      157    0.07%        9        0        9        1        0        1        0        7
20      156       50       59        0       27        4
     4  0xffffffff81e3f540      179    0.06%        9        1        8      117       97       20        0       10
25       62       11        1        0       24        7

=================================================
```

```
      Shared Cache Line Distribution Pareto
================================================
#
#        ----- HITM -----  -- Store Refs --        Data address                          ---------- cycles --
--------       cpu                                     Shared
#   Num      Rmt      Lcl   L1 Hit  L1 Miss              Offset      Pid        Code address  rmt hitm  lcl
hitm      load       cnt              Symbol              Object                Source:Line  Node{cpu list}
# .....  .......  .......  .......  .......  ...................  .......  ...................  ........  ........  ........  ........
.....................  .....................  .............................  ....
#
  ------------------------------------------------------------
     0     9834     2269   109036      468           0x602180
  ------------------------------------------------------------
        65.51%   55.88%   75.20%    0.00%                0x0    14604           0x400b4f     27161
26039     26017         9  [.] read_write_func  no_false_sharing.exe
false_sharing_example.c:144   0{0-1,4}  1{24-25,120}  2{48,54}  3{169}
    0.41%    0.35%    0.00%    0.00%                0x0    14604           0x400b56     18088
12601     26671         9  [.] read_write_func  no_false_sharing.exe
false_sharing_example.c:145   0{0-1,4}  1{24-25,120}  2{48,54}  3{169}
    0.00%    0.00%   24.80%  100.00%                0x0    14604           0x400b61         0         0
0         9  [.] read_write_func  no_false_sharing.exe  false_sharing_example.c:145   0{0-1,4}
1{24-25,120}  2{48,54}  3{169}
    7.50%    9.92%    0.00%    0.00%               0x20    14604           0x400ba7      2470
1729      1897         2  [.] read_write_func  no_false_sharing.exe
false_sharing_example.c:154   1{122}  2{144}
   17.61%   20.89%    0.00%    0.00%               0x28    14604           0x400bc1      2294
1575      1649         2  [.] read_write_func  no_false_sharing.exe
false_sharing_example.c:158   2{53}  3{170}
    8.97%   12.96%    0.00%    0.00%               0x30    14604           0x400bdb      2325
1897      1828         2  [.] read_write_func  no_false_sharing.exe
false_sharing_example.c:162   0{96}  3{171}

  ------------------------------------------------------------
     1     2832     1119        0        0           0x602100
  ------------------------------------------------------------
   29.13%   36.19%    0.00%    0.00%               0x20    14604           0x400bb3      1964
1230      1788         2  [.] read_write_func  no_false_sharing.exe
false_sharing_example.c:155   1{122}  2{144}
   43.68%   34.41%    0.00%    0.00%               0x28    14604           0x400bcd      2274
1566      1793         2  [.] read_write_func  no_false_sharing.exe
false_sharing_example.c:159   2{53}  3{170}
   27.19%   29.40%    0.00%    0.00%               0x30    14604           0x400be7      2045
1247      2011         2  [.] read_write_func  no_false_sharing.exe
false_sharing_example.c:163   0{96}  3{171}
```

24.4. INTERPRÉTATION DU RAPPORT DE PERF C2C

La visualisation affichée en exécutant la commande **perf c2c report --stdio** trie les données en plusieurs tableaux :

Trace Events Information

Ce tableau fournit un résumé de haut niveau de tous les échantillons de chargement et de stockage, qui sont collectés par la commande **perf c2c record**.

Global Shared Cache Line Event Information

Ce tableau fournit des statistiques sur les lignes de cache partagées.

c2c Details

Ce tableau fournit des informations sur les événements échantillonnés et sur la manière dont les données **perf c2c report** sont organisées dans la visualisation.

Shared Data Cache Line Table

Ce tableau fournit un résumé d'une ligne pour les lignes de cache les plus chaudes où un faux partage est détecté et est trié par ordre décroissant par la quantité de **Hitm** distants détectés par ligne de cache par défaut.

Shared Cache Line Distribution Pareto

Ce tableau fournit une série d'informations sur chaque ligne de cache faisant l'objet d'une contention :

- Les lignes de cache sont numérotées dans la colonne **NUM**, à partir de **0**.
- L'adresse virtuelle de chaque ligne de cache figure dans la colonne **Data address Offset** et est suivie par le décalage dans la ligne de cache où les différents accès ont eu lieu.
- La colonne **Pid** contient l'identifiant du processus.
- La colonne **Code Address** contient l'adresse du code du pointeur d'instruction.
- Les colonnes sous l'étiquette **cycles** indiquent les temps de latence moyens.
- La colonne **cpu cnt** indique combien de CPU différents ont fourni les échantillons (en fait, combien de CPU différents attendaient les données indexées à cet emplacement donné).
- La colonne **Symbol** affiche le nom ou le symbole de la fonction.
- La colonne **Shared Object** affiche le nom de l'image ELF d'où proviennent les échantillons (le nom [**kernel.kallsyms**] est utilisé lorsque les échantillons proviennent du noyau).
- La colonne **Source:Line** affiche le fichier source et le numéro de ligne.
- La colonne **Node{cpu list}** indique les unités centrales spécifiques dont proviennent les échantillons pour chaque nœud.

24.5. DÉTECTION DES FAUX PARTAGES AVEC PERF C2C

Cette procédure décrit comment détecter les faux partages à l'aide de la commande **perf c2c**.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé. Pour plus d'informations, voir l' installation de perf.
- Un fichier **perf.data** enregistré à l'aide de la commande **perf c2c** est disponible dans le répertoire actuel. Pour plus d'informations, voir Détection de la contention des lignes de cache avec perf c2c.

Procédure

1. Ouvrez le fichier **perf.data** pour une analyse plus approfondie :

```
# perf c2c report --stdio
```

Cette opération ouvre le fichier **perf.data** dans le terminal.

2. Dans le tableau "Trace Event Information", localisez la ligne contenant les valeurs de **LLC Misses to Remote Cache (HITM)**:
   Le pourcentage dans la colonne de valeur de la ligne **LLC Misses to Remote Cache (HITM)** représente le pourcentage d'échecs LLC survenus entre les nœuds NUMA dans les lignes de cache modifiées et constitue un indicateur clé de l'existence d'un faux partage.

```
=================================================
            Trace Event Information
=================================================
  Total records                     :     329219
  Locked Load/Store Operations      :      14654
  Load Operations                   :      69679
  Loads - uncacheable               :          0
  Loads - IO                        :          0
  Loads - Miss                      :       3972
  Loads - no mapping                :          0
  Load Fill Buffer Hit              :      11958
  Load L1D hit                      :      17235
  Load L2D hit                      :         21
  Load LLC hit                      :      14219
  Load Local HITM                   :       3402
  Load Remote HITM                  :      12757
  Load Remote HIT                   :       5295
  Load Local DRAM                   :        976
  Load Remote DRAM                  :       3246
  Load MESI State Exclusive         :       4222
  Load MESI State Shared            :          0
  Load LLC Misses                   :      22274
  LLC Misses to Local DRAM          :       4.4%
  LLC Misses to Remote DRAM         :      14.6%
  LLC Misses to Remote cache (HIT)  :      23.8%
  LLC Misses to Remote cache (HITM) : 57.3%
  Store Operations                  :     259539
  Store - uncacheable               :          0
  Store - no mapping                :         11
  Store L1D Hit                     :     256696
  Store L1D Miss                    :       2832
  No Page Map Rejects               :       2376
  Unable to parse data source       :          1
```

3. Inspecter la colonne **Rmt** du champ **LLC Load Hitm** du champ **Shared Data Cache Line Table**

```
  =================================================
            Shared Data Cache Line Table
  =================================================
  #
  #                        Total      Rmt  ----- LLC Load Hitm -----  ---- Store Reference ----  ---
Load Dram ----      LLC    Total  ----- Core Load Hit -----  -- LLC Load Hit --
  # Index           Cacheline  records     Hitm    Total      Lcl      Rmt    Total    L1Hit    L1Miss
Lcl       Rmt  Ld Miss    Loads       FB       L1        L2       Llc       Rmt
  # ..... ..................  .......  .......  .......  .......  .......  .......  .......  .......  ........  ........  .......  .......  .......
.......  .......  ........  ........
  #
```

```
      0          0x602180   149904   77.09%    12103     2269     9834   109504   109036
468       727      2657    13747    40400     5355    16154        0     2875       529
      1          0x602100    12128   22.20%     3951     1119     2832        0        0        0       65
200     3749    12128     5096      108        0     2056      652
      2  0xffff883ffb6a7e80      260    0.09%       15        3       12      161      161        0        1
1       15       99       25       50        0        6        1
      3  0xffffffff81aec000      157    0.07%        9        0        9        1        0        1        0        7
20      156       50       59        0       27        4
      4  0xffffffff81e3f540      179    0.06%        9        1        8      117       97       20        0
10       25       62       11        1        0       24        7
```

Ce tableau est classé par ordre décroissant en fonction du nombre de **Hitm** distants détectés par ligne de cache. Un nombre élevé dans la colonne **Rmt** de la section **LLC Load Hitm** indique un faux partage et nécessite une inspection plus approfondie de la ligne de cache sur laquelle il s'est produit afin de déboguer la fausse activité de partage.

CHAPITRE 25. DÉMARRER AVEC FLAMEGRAPHS

En tant qu'administrateur système, vous pouvez utiliser **flamegraphs** pour créer des visualisations des données de performance du système enregistrées avec l'outil **perf**. En tant que développeur de logiciels, vous pouvez utiliser **flamegraphs** pour créer des visualisations des données de performance des applications enregistrées avec l'outil **perf**.

L'échantillonnage des traces de pile est une technique courante pour établir le profil des performances du processeur à l'aide de l'outil **perf**. Malheureusement, les résultats du profilage des traces de pile avec **perf** peuvent être extrêmement verbeux et leur analyse laborieuse. **flamegraphs** sont des visualisations créées à partir des données enregistrées avec **perf** afin d'identifier plus rapidement et plus facilement les chemins de code chauds.

25.1. INSTALLATION DE FLAMEGRAPHS

Pour commencer à utiliser **flamegraphs**, installez le paquetage requis.

Procédure

- Installez le paquetage **flamegraphs**:

```
# dnf install js-d3-flame-graph
```

25.2. CRÉATION DE GRAPHES DE FLAMME SUR L'ENSEMBLE DU SYSTÈME

Cette procédure décrit comment visualiser les données de performance enregistrées sur l'ensemble d'un système à l'aide de **flamegraphs**.

Conditions préalables

- **flamegraphs** sont installés comme décrit dans l' installation de flamegraphs.
- L'outil **perf** est installé comme décrit dans l' installation de perf.

Procédure

- Enregistrez les données et créez la visualisation :

```
# perf script flamegraph -a -F 99 sleep 60
```

Cette commande échantillonne et enregistre les données de performance sur l'ensemble du système pendant 60 secondes, comme stipulé par l'utilisation de la commande **sleep**, et construit ensuite la visualisation qui sera stockée dans le répertoire actif actuel sous le nom de **flamegraph.html**. La commande échantillonne par défaut les données du call-graph et prend les mêmes arguments que l'outil **perf**, dans ce cas particulier :

-a
Stipule d'enregistrer les données sur l'ensemble du système.

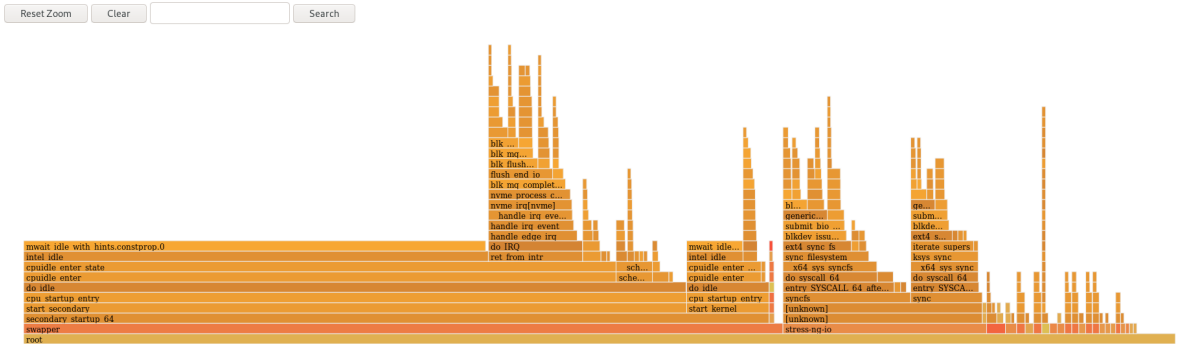
-F
Pour régler la fréquence d'échantillonnage par seconde.

Verification steps

- Pour l'analyse, voir la visualisation générée :

```
# xdg-open flamegraph.html
```

Cette commande ouvre la visualisation dans le navigateur par défaut :



25.3. CRÉATION DE GRAPHES DE FLAMME SUR DES PROCESSUS SPÉCIFIQUES

Vous pouvez utiliser **flamegraphs** pour visualiser les données de performance enregistrées sur des processus spécifiques en cours d'exécution.

Conditions préalables

- **flamegraphs** sont installés comme décrit dans l' installation de flamegraphs.
- L'outil **perf** est installé comme décrit dans l' installation de perf.

Procédure

- Enregistrez les données et créez la visualisation :

```
# perf script flamegraph -a -F 99 -p ID1,ID2 sleep 60
```

Cette commande échantillonne et enregistre les données de performance des processus avec les ID de processus ***ID1*** et ***ID2*** pendant 60 secondes, comme stipulé par l'utilisation de la commande **sleep**, et construit ensuite la visualisation qui sera stockée dans le répertoire actif actuel sous le nom de **flamegraph.html**. La commande échantillonne les données du call-graph par défaut et prend les mêmes arguments que l'outil **perf**, dans ce cas particulier :

-a

Stipule d'enregistrer les données sur l'ensemble du système.

-F

Pour régler la fréquence d'échantillonnage par seconde.

-p

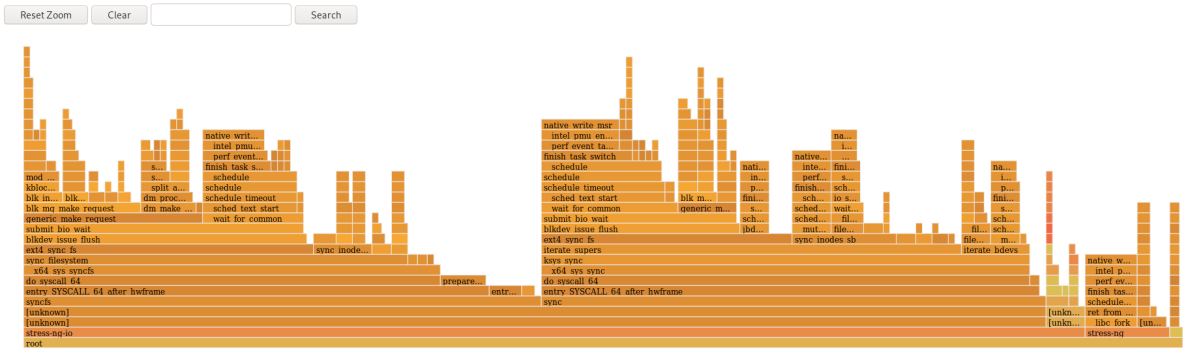
Pour stipuler des identifiants de processus spécifiques pour l'échantillonnage et l'enregistrement des données.

Verification steps

- Pour l'analyse, voir la visualisation générée :

```
# xdg-open flamegraph.html
```

Cette commande ouvre la visualisation dans le navigateur par défaut :



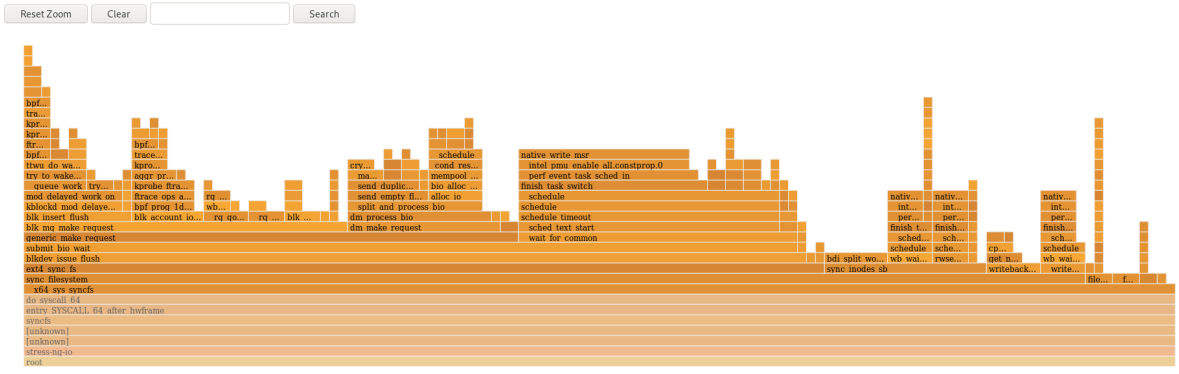
25.4. INTERPRÉTATION DES DIAGRAMMES DE FLAMME

Chaque case du diagramme de flamme représente une fonction différente de la pile. L'axe des y indique la profondeur de la pile, la case la plus haute de chaque pile étant la fonction qui était effectivement sur le CPU et toutes les cases inférieures étant des ancêtres. L'axe des abscisses indique la population des données échantillonnées du graphique d'appel.

Les enfants d'une pile dans une ligne donnée sont affichés en fonction du nombre d'échantillons prélevés pour chaque fonction respective dans l'ordre décroissant le long de l'axe des x ; l'axe des x ne représente pas le passage du temps. Plus une case individuelle est large, plus elle était fréquente sur le CPU ou faisait partie d'une ascendance sur le CPU au moment où les données ont été échantillonnées.

Procédure

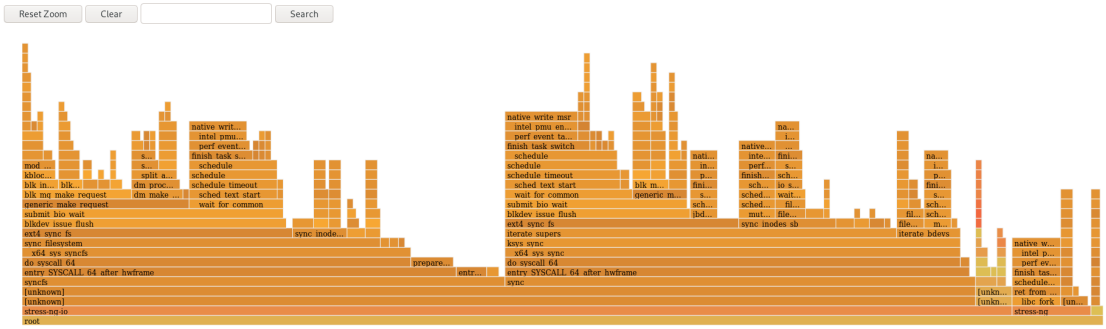
- Pour révéler les noms des fonctions qui n'ont pas été affichées précédemment et approfondir les données, cliquez sur une case du diagramme de flamme pour zoomer sur la pile à l'endroit donné :



- Pour revenir à l'affichage par défaut du diagramme de flamme, cliquez sur **Réinitialiser le zoom**.

IMPORTANT

Les cases représentant des fonctions de l'espace utilisateur peuvent être étiquetées comme **Unknown** dans **flamegraphs** parce que le binaire de la fonction est supprimé. Le paquet **debuginfo** de l'exécutable doit être installé ou, si l'exécutable est une application développée localement, l'application doit être compilée avec des informations de débogage. Utilisez l'option **-g** dans GCC, pour afficher les noms de fonctions ou les symboles dans une telle situation.



Ressources supplémentaires

- Pourquoi perf affiche-t-il certains noms de fonctions comme des adresses de fonctions brutes ?
- Activation du débogage avec les informations de débogage

CHAPITRE 26. SURVEILLANCE DES PROCESSUS POUR DÉTECTER LES GOULETS D'ÉTRANGLEMENT AU NIVEAU DES PERFORMANCES À L'AIDE DES TAMPONS CIRCULAIRES DE PERF

Vous pouvez créer des tampons circulaires qui prennent des instantanés de données spécifiques à un événement avec l'outil **perf** afin de surveiller les goulets d'étranglement des performances dans des processus spécifiques ou des parties d'applications fonctionnant sur votre système. Dans ce cas, **perf** n'écrit des données dans un fichier **perf.data** en vue d'une analyse ultérieure que si un événement spécifique est détecté.

26.1. TAMPONS CIRCULAIRES ET INSTANTANÉS SPÉCIFIQUES À UN ÉVÉNEMENT AVEC PERF

Lorsque vous étudiez les problèmes de performance d'un processus ou d'une application à l'aide de **perf**, il peut ne pas être abordable ou approprié d'enregistrer des données pendant les heures précédant l'apparition d'un événement spécifique intéressant. Dans ce cas, vous pouvez utiliser **perf record** pour créer des tampons circulaires personnalisés qui prennent des instantanés après des événements spécifiques.

L'option **--overwrite** permet à **perf record** de stocker toutes les données dans un tampon circulaire écrasable. Lorsque la mémoire tampon est pleine, **perf record** écrase automatiquement les enregistrements les plus anciens qui, par conséquent, ne sont jamais écrits dans un fichier **perf.data**.

L'utilisation conjointe des options **--overwrite** et **--switch-output-event** permet de configurer un tampon circulaire qui enregistre et déverse des données en continu jusqu'à ce qu'il détecte l'événement déclencheur **--switch-output-event**. L'événement déclencheur signale à **perf record** que quelque chose d'intéressant pour l'utilisateur s'est produit et qu'il faut écrire les données du tampon circulaire dans un fichier **perf.data**. Cela permet de collecter les données spécifiques qui vous intéressent tout en réduisant la charge de travail du processus **perf** en cours d'exécution en n'écrivant pas les données que vous ne voulez pas dans un fichier **perf.data**.

26.2. COLLECTE DE DONNÉES SPÉCIFIQUES POUR SURVEILLER LES GOULETS D'ÉTRANGLEMENT AU NIVEAU DES PERFORMANCES À L'AIDE DES TAMPONS CIRCULAIRES DE PERF

L'outil **perf** vous permet de créer des tampons circulaires déclenchés par des événements que vous spécifiez afin de ne collecter que les données qui vous intéressent. Pour créer des tampons circulaires qui collectent des données spécifiques à un événement, utilisez les options **--overwrite** et **--switch-output-event** pour **perf**.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.
- Vous avez placé une sonde ascendante dans le processus ou l'application que vous souhaitez surveiller, à un endroit qui vous intéresse dans le processus ou l'application :

```
# perf probe -x /path/to/executable -a function
Added new event:
  probe_executable:function   (on function in /path/to/executable)
```

> You can now use it in all perf tools, such as:
>
> perf record -e *probe_executable:function* -aR sleep 1

Procédure

- Créez la mémoire tampon circulaire avec la sonde ascendante comme événement déclencheur :

```
# perf record --overwrite -e cycles --switch-output-event probe_executable:function
./executable
[ perf record: dump data: Woken up 1 times ]
[ perf record: Dump perf.data.2021021012231959 ]
[ perf record: dump data: Woken up 1 times ]
[ perf record: Dump perf.data.2021021012232008 ]
^C[ perf record: dump data: Woken up 1 times ]
[ perf record: Dump perf.data.2021021012232082 ]
[ perf record: Captured and wrote 5.621 MB perf.data.<timestamp> ]
```

Cet exemple lance l'exécutable et collecte les cycles du processeur, spécifiés après l'option **-e**, jusqu'à ce que **perf** détecte l'uprobe, l'événement déclencheur spécifié après l'option **--switch-output-event**. À ce moment-là, **perf** prend un instantané de toutes les données du tampon circulaire et le stocke dans un fichier unique **perf.data** identifié par l'horodatage. Cet exemple a produit un total de 2 instantanés, le dernier fichier **perf.data** a été forcé en appuyant sur **Ctrl c.**

CHAPITRE 27. AJOUTER ET SUPPRIMER DES TRACEPOINTS D'UN COLLECTEUR DE PERF EN COURS D'EXÉCUTION SANS ARRÊTER OU REDÉMARRER PERF

En utilisant l'interface du tuyau de contrôle pour activer et désactiver différents points de traçage dans un collecteur **perf** en cours d'exécution, vous pouvez ajuster dynamiquement les données que vous collectez sans avoir à arrêter ou à redémarrer **perf**. Vous êtes ainsi assuré de ne pas perdre les données de performance qui auraient été enregistrées pendant le processus d'arrêt ou de redémarrage.

27.1. AJOUTER DES TRACEPOINTS À UN COLLECTEUR PERF EN COURS D'EXÉCUTION SANS ARRÊTER OU REDÉMARRER PERF

Ajoutez des tracepoints à un collecteur **perf** en cours d'exécution à l'aide de l'interface du tuyau de contrôle pour ajuster les données que vous enregistrez sans avoir à arrêter **perf** et à perdre des données de performance.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.

Procédure

1. Configurer l'interface de la conduite de contrôle :

```
# mkfifo control ack perf.pipe
```

2. Lancez **perf record** avec la configuration du fichier de contrôle et les événements que vous souhaitez activer :

```
# perf record --control=fifo:control,ack -D -1 --no-buffering -e 'sched:*' -o - > perf.pipe
```

Dans cet exemple, la déclaration de **'sched:*'** après l'option **-e** lance **perf record** avec les événements de l'ordonnanceur.

3. Dans un second terminal, démarrez le côté lecture du tuyau de contrôle :

```
# cat perf.pipe | perf --no-pager script -i -
```

Le démarrage de la partie lecture du tube de contrôle déclenche le message suivant dans le premier terminal :

```
Events disabled
```

4. Dans un troisième terminal, activez un point de contrôle à l'aide du fichier de contrôle :

```
# echo 'enable sched:sched_process_fork' > control
```

Cette commande déclenche l'analyse par **perf** de la liste des événements en cours dans le fichier de contrôle, à la recherche de l'événement déclaré. Si l'événement est présent, le point de contrôle est activé et le message suivant apparaît dans le premier terminal :

```
event sched:sched_process_fork enabled
```

Une fois le point de contrôle activé, le second terminal affiche la sortie de **perf** détectant le point de contrôle :

```
bash 33349 [034] 149587.674295: sched:sched_process_fork: comm=bash pid=33349
child_comm=bash child_pid=34056
```

27.2. SUPPRIMER LES TRACEPOINTS D'UN COLLECTEUR DE PERF EN COURS D'EXÉCUTION SANS ARRÊTER OU REDÉMARRER PERF

Supprimez les tracepoints d'un collecteur **perf** en cours d'exécution à l'aide de l'interface control pipe afin de réduire la portée des données collectées sans devoir arrêter **perf** et perdre des données de performance.

Conditions préalables

- L'outil de l'espace utilisateur **perf** est installé comme décrit dans la section Installation de perf.
- Vous avez ajouté des points de contrôle à un collecteur **perf** en cours d'exécution via l'interface control pipe. Pour plus d'informations, voir Ajouter des tracepoints à un collecteur perf en cours d'exécution sans arrêter ou redémarrer perf.

Procédure

- Retirer le point de traçage :

```
# echo 'disable sched:sched_process_fork' > control
```

NOTE

Cet exemple suppose que vous avez préalablement chargé les événements de l'ordonnanceur dans le fichier de contrôle et activé le point de contrôle **sched:sched_process_fork**.

Cette commande déclenche l'analyse par **perf** de la liste des événements en cours dans le fichier de contrôle, à la recherche de l'événement déclaré. Si l'événement est présent, le point de contrôle est désactivé et le message suivant apparaît dans le terminal utilisé pour configurer la conduite de contrôle :

```
event sched:sched_process_fork disabled
```

CHAPITRE 28. PROFILER L'ALLOCATION DE MÉMOIRE AVEC NUMASTAT

L'outil **numastat** permet d'afficher des statistiques sur les allocations de mémoire dans un système.

L'outil **numastat** affiche les données pour chaque nœud NUMA séparément. Vous pouvez utiliser ces informations pour étudier les performances de la mémoire de votre système ou l'efficacité de différentes stratégies de mémoire sur votre système.

28.1. STATISTIQUES NUMASTAT PAR DÉFAUT

Par défaut, l'outil **numastat** affiche des statistiques sur ces catégories de données pour chaque nœud NUMA :

numa_hit

Nombre de pages allouées avec succès à ce nœud.

numa_miss

Le nombre de pages qui ont été allouées sur ce nœud en raison d'une mémoire insuffisante sur le nœud prévu. Chaque événement **numa_miss** a un événement **numa_foreign** correspondant sur un autre nœud.

numa_foreign

Le nombre de pages initialement prévues pour ce nœud qui ont été allouées à un autre nœud à la place. Chaque événement **numa_foreign** a un événement **numa_miss** correspondant sur un autre nœud.

interleave_hit

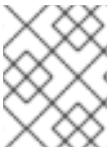
Nombre de pages de politique d'entrelacement allouées avec succès à ce nœud.

local_node

Nombre de pages allouées avec succès par un processus sur ce nœud.

other_node

Le nombre de pages allouées sur ce nœud par un processus sur un autre nœud.



NOTE

Les valeurs élevées de **numa_hit** et les valeurs faibles de **numa_miss** (l'une par rapport à l'autre) indiquent une performance optimale.

28.2. VISUALISATION DE L'ALLOCATION DE MÉMOIRE AVEC NUMASTAT

Vous pouvez visualiser l'allocation de la mémoire du système en utilisant l'outil **numastat**.

Conditions préalables

- Installez le paquetage **numactl**:

```
# dnf install numactl
```

Procédure

- Affichez l'allocation de mémoire de votre système :

```
$ numastat
                           node0          node1
numa_hit                76557759       92126519
numa_miss               30772308       30827638
numa_foreign            30827638       30772308
interleave_hit            106507         103832
local_node              76502227       92086995
other_node              30827840       30867162
```

Ressources supplémentaires

- **numastat(8)** page de manuel

CHAPITRE 29. CONFIGURATION D'UN SYSTÈME D'EXPLOITATION POUR OPTIMISER L'UTILISATION DE L'UNITÉ CENTRALE

Vous pouvez configurer le système d'exploitation pour optimiser l'utilisation de l'unité centrale en fonction des charges de travail.

29.1. OUTILS DE SURVEILLANCE ET DE DIAGNOSTIC DES PROBLÈMES LIÉS AU PROCESSEUR

Les outils suivants sont disponibles dans Red Hat Enterprise Linux 9 pour surveiller et diagnostiquer les problèmes de performance liés au processeur :

- **turbostat** imprime les résultats des compteurs à des intervalles spécifiés afin d'aider les administrateurs à identifier les comportements inattendus des serveurs, tels qu'une consommation d'énergie excessive, l'impossibilité d'entrer en état de veille profonde ou la création inutile d'interruptions de gestion du système (SMI).

- **numactl** fournit un certain nombre d'options pour gérer l'affinité entre le processeur et la mémoire. Le paquetage **numactl** comprend la bibliothèque **libnuma** qui offre une interface de programmation simple pour la politique NUMA supportée par le noyau, et peut être utilisée pour un réglage plus fin que l'application **numactl**.

- **numastat** affiche des statistiques sur la mémoire du système d'exploitation et de ses processus par nœud NUMA et indique aux administrateurs si la mémoire des processus est répartie sur l'ensemble du système ou si elle est centralisée sur des nœuds spécifiques. Cet outil est fourni par le paquetage **numactl**.

- **numad** est un démon de gestion automatique des affinités NUMA. Il surveille la topologie NUMA et l'utilisation des ressources au sein d'un système afin d'améliorer dynamiquement l'allocation et la gestion des ressources NUMA.

- /**proc**/**interrupts** affiche le numéro de la demande d'interruption (IRQ), le nombre de demandes d'interruption similaires traitées par chaque processeur du système, le type d'interruption envoyée et une liste séparée par des virgules des périphériques qui répondent à la demande d'interruption répertoriée.

- **pqos** est disponible dans le paquetage **intel-cmt-cat**. Il surveille le cache du processeur et la bande passante de la mémoire sur les processeurs Intel récents. Il surveille :

  - Les instructions par cycle (IPC).
  - Le nombre d'échecs du dernier niveau de cache.
  - Taille en kilo-octets que le programme s'exécutant dans une unité centrale donnée occupe dans le LLC.
  - La bande passante de la mémoire locale (MBL).
  - La bande passante vers la mémoire distante (MBR).

- **x86_energy_perf_policy** permet aux administrateurs de définir l'importance relative des performances et de l'efficacité énergétique. Ces informations peuvent ensuite être utilisées pour influencer les processeurs qui prennent en charge cette fonctionnalité lorsqu'ils sélectionnent des options qui mettent en balance les performances et l'efficacité énergétique.

- **taskset** est fourni par le paquetage **util-linux**. Il permet aux administrateurs de récupérer et de définir l'affinité processeur d'un processus en cours d'exécution, ou de lancer un processus avec une affinité processeur spécifiée.

Ressources supplémentaires

- **turbostat(8)**, **numactl(8)**, **numastat(8)**, **numa(7)**, **numad(8)**, **pqos(8)**, **x86_energy_perf_policy(8)**, et **taskset(1)** pages de manuel

29.2. TYPES DE TOPOLOGIE DE SYSTÈME

Dans l'informatique moderne, l'idée d'une unité centrale est trompeuse, car la plupart des systèmes modernes sont dotés de plusieurs processeurs. La topologie du système est la manière dont ces processeurs sont connectés les uns aux autres et aux autres ressources du système. Cela peut affecter les performances du système et de l'application, ainsi que les considérations de réglage d'un système.

Les deux principaux types de topologie utilisés dans l'informatique moderne sont les suivants :

Symmetric Multi-Processor (SMP) topology

La topologie SMP permet à tous les processeurs d'accéder à la mémoire dans le même laps de temps. Toutefois, comme l'accès partagé et égal à la mémoire oblige intrinsèquement tous les processeurs à effectuer des accès sérialisés à la mémoire, les contraintes de mise à l'échelle des systèmes SMP sont aujourd'hui généralement considérées comme inacceptables. C'est pourquoi pratiquement tous les systèmes de serveurs modernes sont des machines NUMA.

Non-Uniform Memory Access (NUMA) topology

La topologie NUMA a été développée plus récemment que la topologie SMP. Dans un système NUMA, plusieurs processeurs sont physiquement regroupés sur un socket. Chaque socket dispose d'une zone de mémoire dédiée et de processeurs qui ont un accès local à cette mémoire. Les processeurs d'un même nœud ont un accès rapide à la banque de mémoire de ce nœud et un accès plus lent aux banques de mémoire qui ne se trouvent pas sur leur nœud.
Par conséquent, l'accès à la mémoire non locale entraîne une pénalité en termes de performances. Ainsi, les applications sensibles aux performances sur un système à topologie NUMA devraient accéder à la mémoire qui se trouve sur le même nœud que le processeur qui exécute l'application, et devraient éviter d'accéder à la mémoire distante dans la mesure du possible.

Les applications multithreads sensibles aux performances peuvent bénéficier d'une configuration leur permettant de s'exécuter sur un nœud NUMA spécifique plutôt que sur un processeur spécifique. La pertinence de cette configuration dépend de votre système et des exigences de votre application. Si plusieurs threads d'application accèdent aux mêmes données mises en cache, il peut être judicieux de configurer ces threads pour qu'ils s'exécutent sur le même processeur. Toutefois, si plusieurs threads qui accèdent à des données différentes et les mettent en cache s'exécutent sur le même processeur, chaque thread peut évincer des données mises en cache auxquelles un thread précédent a accédé. Cela signifie que chaque thread "manque" le cache et perd du temps d'exécution en allant chercher les données dans la mémoire et en les replaçant dans le cache. Utilisez l'outil **perf** pour vérifier si le nombre de manques dans le cache est excessif.

29.2.1. Affichage des topologies de systèmes

Un certain nombre de commandes permettent de comprendre la topologie d'un système. Cette procédure décrit comment déterminer la topologie du système.

Procédure

- Pour afficher une vue d'ensemble de la topologie de votre système :

```
$ numactl --hardware
available: 4 nodes (0-3)
node 0 cpus: 0 4 8 12 16 20 24 28 32 36
node 0 size: 65415 MB
node 0 free: 43971 MB
[...]
```

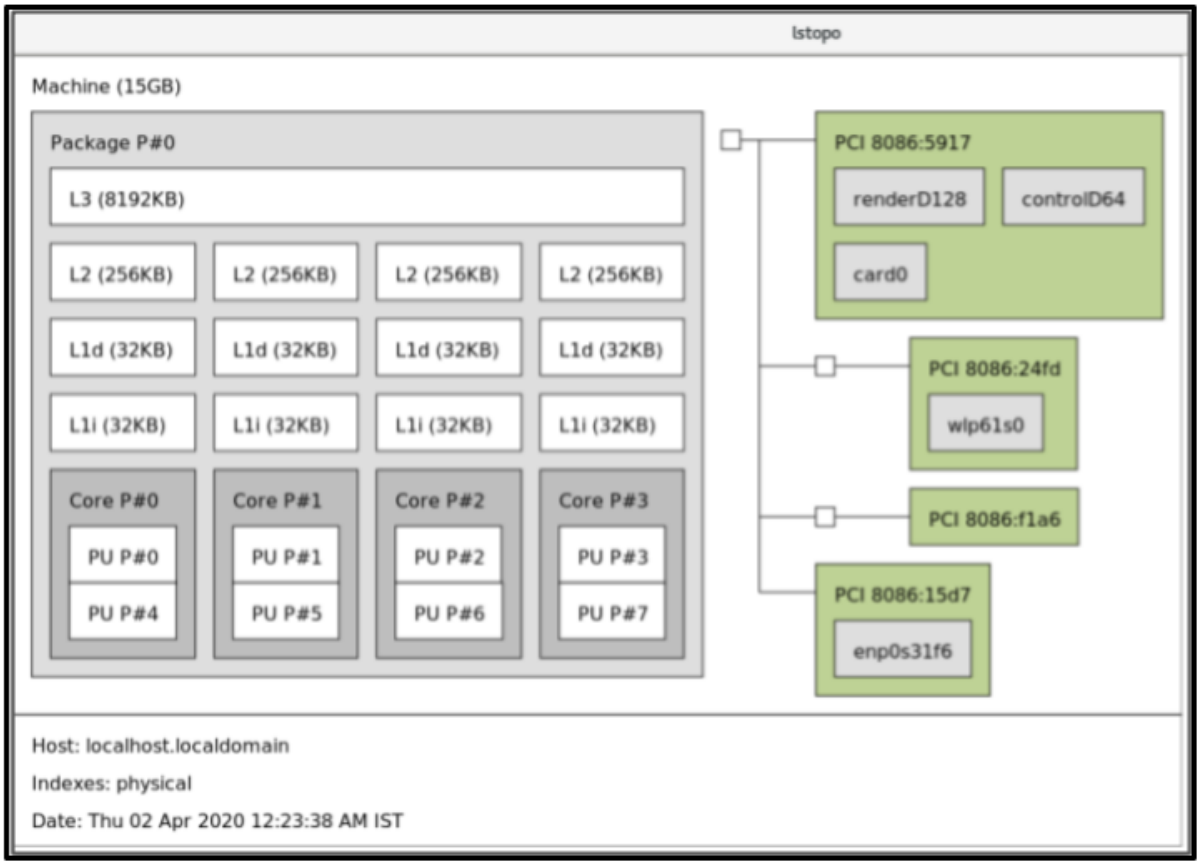
- Rassembler les informations sur l'architecture de l'unité centrale, telles que le nombre d'unités centrales, de threads, de cœurs, de sockets et de nœuds NUMA :

```
$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                40
On-line CPU(s) list:   0-39
Thread(s) per core:    1
Core(s) per socket:    10
Socket(s):             4
NUMA node(s):          4
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 47
Model name:            Intel(R) Xeon(R) CPU E7- 4870  @ 2.40GHz
Stepping:              2
CPU MHz:               2394.204
BogoMIPS:              4787.85
Virtualization:        VT-x
L1d cache:             32K
L1i cache:             32K
L2 cache:              256K
L3 cache:              30720K
NUMA node0 CPU(s):     0,4,8,12,16,20,24,28,32,36
NUMA node1 CPU(s):     2,6,10,14,18,22,26,30,34,38
NUMA node2 CPU(s):     1,5,9,13,17,21,25,29,33,37
NUMA node3 CPU(s):     3,7,11,15,19,23,27,31,35,39
```

- Pour afficher une représentation graphique de votre système :

```
# dnf install hwloc-gui
# lstopo
```

Figure 29.1. La sortie **lstopo**



- Pour afficher le texte détaillé :

```
# dnf install hwloc
# lstopo-no-graphics
Machine (15GB)
  Package L#0 + L3 L#0 (8192KB)
    L2 L#0 (256KB) + L1d L#0 (32KB) + L1i L#0 (32KB) + Core L#0
        PU L#0 (P#0)
        PU L#1 (P#4)
      HostBridge L#0
    PCI 8086:5917
        GPU L#0 "renderD128"
        GPU L#1 "controlD64"
        GPU L#2 "card0"
    PCIBridge
        PCI 8086:24fd
          Net L#3 "wlp61s0"
    PCIBridge
        PCI 8086:f1a6
    PCI 8086:15d7
        Net L#4 "enp0s31f6"
```

Ressources supplémentaires

- **numactl(8)**, **lscpu(1)**, et **lstopo(1)** pages de manuel

29.3. CONFIGURATION DU TEMPS DE RÉPONSE DU NOYAU

Par défaut, Red Hat Enterprise Linux 9 utilise un noyau "tickless", qui n'interrompt pas les processeurs en veille afin de réduire la consommation d'énergie et de permettre aux nouveaux processeurs de tirer parti des états de sommeil profond.

Red Hat Enterprise Linux 9 propose également une option tickless dynamique, qui est utile pour les charges de travail sensibles à la latence, telles que l'informatique à haute performance ou l'informatique en temps réel. Par défaut, l'option tickless dynamique est désactivée. Red Hat recommande d'utiliser le profil **cpu-partitioning TuneD** pour activer l'option tickless dynamique pour les noyaux spécifiés comme **isolated_cores**.

Cette procédure décrit comment activer manuellement et de manière persistante le comportement dynamique sans tic-tac.

Procédure

1. Pour activer le comportement dynamique sans tic-tac dans certains cœurs, spécifiez ces cœurs sur la ligne de commande du noyau avec le paramètre **nohz_full**. Sur un système à 16 cœurs, activez l'option de noyau **nohz_full=1-15**:

   ```
   # grubby --update-kernel=ALL --args="nohz_full=1-15"
   ```

   Cela permet un comportement dynamique sans tic-tac sur les cœurs **1** à **15**, en déplaçant tout le chronométrage vers le seul cœur non spécifié (cœur **0**).

2. Lorsque le système démarre, déplacez manuellement les threads **rcu** vers le cœur non sensible à la latence, en l'occurrence le cœur **0**:

   ```
   # for i in `pgrep rcu[^c]` ; do taskset -pc 0 $i ; done
   ```

3. Facultatif : Utilisez le paramètre **isolcpus** sur la ligne de commande du noyau pour isoler certains cœurs des tâches de l'espace utilisateur.

4. Facultatif : Définissez l'affinité du CPU pour les threads du noyau **write-back bdi-flush** sur le noyau de gestion :

   ```
   echo 1 > /sys/bus/workqueue/devices/writeback/cpumask
   ```

Verification steps

- Une fois le système redémarré, vérifiez si **dynticks** est activé :

  ```
  # journalctl -xe | grep dynticks
  Mar 15 18:34:54 rhel-server kernel: NO_HZ: Full dynticks CPUs: 1-15.
  ```

- Vérifiez que la configuration dynamique sans tic-tac fonctionne correctement :

  ```
  # perf stat -C 1 -e irq_vectors:local_timer_entry taskset -c 1 sleep 3
  ```

  Cette commande mesure les tics de l'unité centrale 1 tout en demandant à l'unité centrale 1 de dormir pendant 3 secondes.

- La configuration par défaut de la minuterie du noyau affiche environ 3100 tics sur un processeur normal :

```
# perf stat -C 0 -e irq_vectors:local_timer_entry taskset -c 0 sleep 3

 Performance counter stats for 'CPU(s) 0':

             3,107      irq_vectors:local_timer_entry

       3.001342790 seconds time elapsed
```

- Avec le noyau dynamique sans tic-tac configuré, vous devriez voir environ 4 tics à la place :

```
# perf stat -C 1 -e irq_vectors:local_timer_entry taskset -c 1 sleep 3

 Performance counter stats for 'CPU(s) 1':

                 4      irq_vectors:local_timer_entry

       3.001544078 seconds time elapsed
```

Ressources supplémentaires

- **perf(1)** et **cpuset(7)** pages de manuel
- Tout sur le paramètre nohz_full du noyau Article de la base de connaissances de Red Hat
- Comment vérifier la liste des informations sur les processeurs \N "isolated" et \N "nohz_full" à partir de sysfs ? À partir de sysfs ? Article de la base de connaissances de Red Hat

29.4. APERÇU D'UNE DEMANDE D'INTERRUPTION

Une demande d'interruption ou IRQ est un signal d'attention immédiate envoyé par un matériel à un processeur. Chaque périphérique d'un système se voit attribuer un ou plusieurs numéros d'IRQ qui lui permettent d'envoyer des interruptions uniques. Lorsque les interruptions sont activées, un processeur qui reçoit une demande d'interruption interrompt immédiatement l'exécution de l'application en cours afin de répondre à la demande d'interruption.

Étant donné que les interruptions interrompent le fonctionnement normal, des taux d'interruption élevés peuvent gravement dégrader les performances du système. Il est possible de réduire le temps pris par les interruptions en configurant l'affinité d'interruption ou en envoyant un certain nombre d'interruptions de moindre priorité dans un lot (coalescence d'un certain nombre d'interruptions).

Les demandes d'interruption sont associées à une propriété d'affinité, **smp_affinity**, qui définit les processeurs qui traitent la demande d'interruption. Pour améliorer les performances de l'application, attribuez l'affinité d'interruption et l'affinité de processus au même processeur ou aux processeurs du même cœur. Cela permet aux threads d'interruption et d'application spécifiés de partager des lignes de cache.

Sur les systèmes qui prennent en charge le pilotage des interruptions, la modification de la propriété **smp_affinity** d'une demande d'interruption configure le matériel de telle sorte que la décision de gérer une interruption avec un processeur particulier est prise au niveau du matériel, sans intervention du noyau.

29.4.1. Équilibrer les interruptions manuellement

Si votre BIOS exporte sa topologie NUMA, le service **irqbalance** peut automatiquement servir les demandes d'interruption sur le nœud local du matériel demandant le service.

Procédure

1. Vérifiez quels appareils correspondent aux demandes d'interruption que vous souhaitez configurer.

2. Recherchez les spécifications matérielles de votre plate-forme. Vérifiez si le chipset de votre système prend en charge la distribution des interruptions.

   a. Si c'est le cas, vous pouvez configurer la distribution des interruptions comme décrit dans les étapes suivantes. En outre, vérifiez l'algorithme utilisé par votre chipset pour équilibrer les interruptions. Certains BIOS disposent d'options permettant de configurer l'acheminement des interruptions.

   b. Si ce n'est pas le cas, votre chipset achemine toujours toutes les interruptions vers une seule unité centrale statique. Vous ne pouvez pas configurer l'unité centrale utilisée.

3. Vérifiez le mode APIC (Advanced Programmable Interrupt Controller) utilisé sur votre système :

   ```
   $ journalctl --dmesg | grep APIC
   ```

   Here,

   - Si votre système utilise un mode autre que **flat**, vous pouvez voir une ligne similaire à **Setting APIC routing to physical flat**.

   - Si vous ne voyez aucun message de ce type, votre système utilise le mode **flat**.
     Si votre système utilise le mode **x2apic**, vous pouvez le désactiver en ajoutant l'option **nox2apic** à la ligne de commande du noyau dans la configuration **bootloader**.

     Seul le mode plat non physique (**flat**) permet de distribuer les interruptions à plusieurs CPU. Ce mode n'est disponible que pour les systèmes comportant jusqu'à **8** CPU.

4. Calculez le **smp_affinity mask**. Pour plus d'informations sur la manière de calculer le **smp_affinity mask**, voir Définition du masque d'affinité smp_affinity.

Ressources supplémentaires

- **journalctl(1)** et **taskset(1)** pages de manuel

29.4.2. Définition du masque d'affinité smp_affinity

La valeur **smp_affinity** est stockée sous la forme d'un masque de bits hexadécimaux représentant tous les processeurs du système. Chaque bit configure un processeur différent. Le bit le moins significatif correspond à l'unité centrale 0.

La valeur par défaut du masque est **f**, ce qui signifie qu'une demande d'interruption peut être traitée par n'importe quel processeur du système. La valeur 1 signifie que seul le processeur 0 peut gérer l'interruption.

Procédure

1. En binaire, utilisez la valeur 1 pour les unités centrales qui gèrent les interruptions. Par exemple, pour configurer l'unité centrale 0 et l'unité centrale 7 afin qu'elles gèrent les interruptions, utilisez **0000000010000001** comme code binaire :

   Tableau 29.1. Bits binaires pour les CPU

UNITÉ CENTRALE	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Binaire	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1

2. Convertir le code binaire en hexadécimal :
   Par exemple, pour convertir le code binaire en utilisant Python :

```
>>> hex(int('0000000010000001', 2))

'0x81'
```

   Sur les systèmes comportant plus de 32 processeurs, vous devez délimiter les valeurs **smp_affinity** pour des groupes discrets de 32 bits. Par exemple, si vous souhaitez que seuls les 32 premiers processeurs d'un système à 64 processeurs répondent à une demande d'interruption, utilisez **0xffffffff,00000000**.

3. La valeur d'affinité d'interruption pour une demande d'interruption particulière est stockée dans le fichier **/proc/irq/irq_number/smp_affinity** associé. Définissez le masque **smp_affinity** dans ce fichier :

```
# echo mask > /proc/irq/irq_number/smp_affinity
```

Ressources supplémentaires

- **journalctl(1)**, **irqbalance(1)**, et **taskset(1)** pages de manuel

CHAPITRE 30. OPTIMISATION DE LA POLITIQUE D'ORDONNANCEMENT

Dans Red Hat Enterprise Linux, la plus petite unité d'exécution d'un processus s'appelle un thread. Le planificateur du système détermine quel processeur exécute un thread, et pendant combien de temps le thread s'exécute. Cependant, comme la principale préoccupation du planificateur est de maintenir le système occupé, il se peut qu'il ne planifie pas les threads de manière optimale pour la performance de l'application.

Par exemple, supposons qu'une application sur un système NUMA s'exécute sur le nœud A lorsqu'un processeur sur le nœud B devient disponible. Pour occuper le processeur du nœud B, l'ordonnanceur déplace l'un des threads de l'application vers le nœud B. Cependant, le thread de l'application doit toujours accéder à la mémoire du nœud A. Mais l'accès à cette mémoire sera plus long car le thread s'exécute maintenant sur le nœud B et la mémoire du nœud A n'est plus locale pour le thread. Ainsi, l'exécution du thread sur le nœud B peut prendre plus de temps qu'il n'en aurait fallu pour attendre qu'un processeur soit disponible sur le nœud A, puis pour exécuter le thread sur le nœud d'origine avec un accès local à la mémoire.

30.1. CATÉGORIES DE POLITIQUES D'ORDONNANCEMENT

Les applications sensibles aux performances bénéficient souvent du fait que le concepteur ou l'administrateur détermine où les threads sont exécutés. L'ordonnanceur Linux met en œuvre un certain nombre de politiques d'ordonnancement qui déterminent où et pendant combien de temps un thread s'exécute.

Les deux grandes catégories de politiques d'ordonnancement sont les suivantes :

Normal policies

Les threads normaux sont utilisés pour des tâches de priorité normale.

Realtime policies

Les politiques de temps réel sont utilisées pour les tâches sensibles au temps qui doivent être exécutées sans interruption. Les threads en temps réel ne sont pas soumis au découpage temporel. Cela signifie qu'ils s'exécutent jusqu'à ce qu'ils se bloquent, sortent, cèdent volontairement leur place ou soient préemptés par un thread de priorité supérieure.
Le thread temps réel ayant la priorité la plus basse est ordonnancé avant tout thread ayant une politique normale. Pour plus d'informations, voir Ordonnancement statique par priorité avec SCHED_FIFO et Ordonnancement par priorité à la ronde avec SCHED_RR .

Ressources supplémentaires

- **sched(7)**, **sched_setaffinity(2)**, **sched_getaffinity(2)**, **sched_setscheduler(2)**, et **sched_getscheduler(2)** pages de manuel

30.2. ORDONNANCEMENT STATIQUE DES PRIORITÉS AVEC SCHED_FIFO

Le **SCHED_FIFO**, également appelé ordonnancement statique des priorités, est une politique en temps réel qui définit une priorité fixe pour chaque thread. Cette politique permet aux administrateurs d'améliorer le temps de réponse aux événements et de réduire la latence. Il est recommandé de ne pas exécuter cette politique pendant une période prolongée pour les tâches sensibles au temps.

Lorsque **SCHED_FIFO** est en cours d'utilisation, l'ordonnanceur parcourt la liste de tous les threads

SCHED_FIFO par ordre de priorité et programme le thread de priorité la plus élevée qui est prêt à s'exécuter. Le niveau de priorité d'un thread **SCHED_FIFO** peut être n'importe quel nombre entier entre **1** et **99**, où **99** est considéré comme la priorité la plus élevée. Red Hat recommande de commencer par un nombre inférieur et d'augmenter la priorité uniquement lorsque vous identifiez des problèmes de latence.

AVERTISSEMENT

Étant donné que les threads en temps réel ne sont pas soumis au découpage temporel, Red Hat ne recommande pas de définir une priorité de 99. Cela maintient votre processus au même niveau de priorité que les threads de migration et de chien de garde ; si votre thread entre dans une boucle de calcul et que ces threads sont bloqués, ils ne pourront pas s'exécuter. Les systèmes dotés d'un seul processeur finiront par se bloquer dans cette situation.

Les administrateurs peuvent limiter la bande passante de **SCHED_FIFO** pour empêcher les programmeurs d'applications en temps réel de lancer des tâches en temps réel qui monopolisent le processeur.

Voici quelques-uns des paramètres utilisés dans cette politique :

/proc/sys/kernel/sched_rt_period_us

Ce paramètre définit la période de temps, en microsecondes, qui est considérée comme correspondant à cent pour cent de la bande passante du processeur. La valeur par défaut est **1000000 µs** ou **1 second**.

/proc/sys/kernel/sched_rt_runtime_us

Ce paramètre définit la période de temps, en microsecondes, consacrée à l'exécution des threads en temps réel. La valeur par défaut est **950000 µs**, ou **0.95 seconds**.

30.3. ORDONNANCEMENT PRIORITAIRE À LA RONDE AVEC SCHED_RR

La politique **SCHED_RR** est une variante round-robin de la politique **SCHED_FIFO**. Cette politique est utile lorsque plusieurs threads doivent s'exécuter au même niveau de priorité.

Comme **SCHED_FIFO**, **SCHED_RR** est une politique en temps réel qui définit une priorité fixe pour chaque thread. L'ordonnanceur parcourt la liste de tous les threads SCHED_RR par ordre de priorité et programme le thread de priorité la plus élevée qui est prêt à être exécuté. Cependant, contrairement à **SCHED_FIFO**, les threads qui ont la même priorité sont programmés dans un style round-robin dans une certaine tranche de temps.

Vous pouvez définir la valeur de cette tranche de temps en millisecondes à l'aide du paramètre du noyau **sched_rr_timeslice_ms** dans le fichier **/proc/sys/kernel/sched_rr_timeslice_ms**. La valeur la plus basse est **1 millisecond**.

30.4. ORDONNANCEMENT NORMAL AVEC SCHED_OTHER

SCHED_OTHER est la politique d'ordonnancement par défaut de Red Hat Enterprise Linux 9. Cette politique utilise l'Ordonnanceur complètement équitable (CFS) pour permettre un accès équitable au

processeur à tous les threads ordonnancés avec cette politique. Cette politique est particulièrement utile lorsqu'il y a un grand nombre de threads ou lorsque le débit des données est une priorité, car elle permet une planification plus efficace des threads dans le temps.

Lorsque cette stratégie est utilisée, l'ordonnanceur crée une liste de priorité dynamique basée en partie sur la valeur de gentillesse de chaque processus. Les administrateurs peuvent modifier la valeur d'agrément d'un processus, mais ne peuvent pas modifier directement la liste de priorité dynamique de l'ordonnanceur.

30.5. DÉFINITION DES RÈGLES DE L'ORDONNANCEUR

L'outil en ligne de commande **chrt** permet de vérifier et d'ajuster les politiques et les priorités de l'ordonnanceur. Il permet de lancer de nouveaux processus avec les propriétés souhaitées ou de modifier les propriétés d'un processus en cours d'exécution. Il peut également être utilisé pour définir la politique au moment de l'exécution.

Procédure

1. Affiche l'ID du processus (PID) des processus actifs :

```
# ps
```

   Utilisez l'option **--pid** ou **-p** avec la commande **ps** pour afficher les détails du PID en question.

2. Vérifier la politique d'ordonnancement, le PID et la priorité d'un processus particulier :

```
# chrt -p 468
pid 468's current scheduling policy: SCHED_FIFO
pid 468's current scheduling priority: 85

# chrt -p 476
pid 476's current scheduling policy: SCHED_OTHER
pid 476's current scheduling priority: 0
```

   Ici, *468* et *476* sont les PID d'un processus.

3. Définir la politique d'ordonnancement d'un processus :

   a. Par exemple, pour définir le processus avec le PID *1000* à *SCHED_FIFO*, avec une priorité de *50*:

```
# chrt -f -p 50 1000
```

   b. Par exemple, pour définir le processus avec le PID *1000* à *SCHED_OTHER*, avec une priorité de *0*:

```
# chrt -o -p 0 1000
```

   c. Par exemple, pour définir le processus avec le PID *1000* à *SCHED_RR*, avec une priorité de *10*:

```
# chrt -r -p 10 1000
```

d. Pour lancer une nouvelle application avec une politique et une priorité particulières, indiquez le nom de l'application :

```
# chrt -f 36 /bin/my-app
```

Ressources supplémentaires

- **chrt(1)** page de manuel
- Options de politique pour la commande chrt
- Modifier la priorité des services pendant le processus de démarrage

30.6. OPTIONS DE POLITIQUE POUR LA COMMANDE CHRT

La commande **chrt** permet de visualiser et de définir la politique d'ordonnancement d'un processus.

Le tableau suivant décrit les options de politique appropriées, qui peuvent être utilisées pour définir la politique d'ordonnancement d'un processus.

Tableau 30.1. Options de politique pour la commande chrt

Option courte	Option longue	Description
-f	**--fifo**	Régler l'horaire sur **SCHED_FIFO**
-o	**--other**	Régler l'horaire sur **SCHED_OTHER**
-r	**--rr**	Régler l'horaire sur **SCHED_RR**

30.7. MODIFIER LA PRIORITÉ DES SERVICES PENDANT LE PROCESSUS DE DÉMARRAGE

En utilisant le service **systemd**, il est possible de définir des priorités en temps réel pour les services lancés pendant le processus de démarrage. Le site *unit configuration directives* permet de modifier la priorité d'un service pendant le processus de démarrage.

Le changement de priorité du processus de démarrage s'effectue en utilisant les directives suivantes dans la section service :

CPUSchedulingPolicy=

Définit la politique d'ordonnancement du processeur pour les processus exécutés. Elle est utilisée pour définir les politiques **other**, **fifo** et **rr**.

CPUSchedulingPriority=

Définit la priorité d'ordonnancement du CPU pour les processus exécutés. La plage de priorité disponible dépend de la politique d'ordonnancement du CPU sélectionnée. Pour les politiques d'ordonnancement en temps réel, un nombre entier entre **1** (priorité la plus faible) et **99** (priorité la plus élevée) peut être utilisé.

La procédure suivante décrit comment modifier la priorité d'un service, pendant le processus de démarrage, à l'aide du service **mcelog**.

Conditions préalables

1. Installez le paquet TuneD :

```
# dnf install tuned
```

2. Activez et démarrez le service TuneD :

```
# systemctl enable --now tuned
```

Procédure

1. Afficher les priorités d'ordonnancement des threads en cours d'exécution :

```
# tuna --show_threads
                    thread       ctxt_switches
   pid SCHED_ rtpri affinity voluntary nonvoluntary            cmd
  1     OTHER    0    0xff      3181          292         systemd
  2     OTHER    0    0xff       254            0        kthreadd
  3     OTHER    0    0xff         2            0         rcu_gp
  4     OTHER    0    0xff         2            0      rcu_par_gp
  6     OTHER    0       0         9            0 kworker/0:0H-kblockd
  7     OTHER    0    0xff      1301            1 kworker/u16:0-events_unbound
  8     OTHER    0    0xff         2            0    mm_percpu_wq
  9     OTHER    0       0       266            0     ksoftirqd/0
[...]
```

2. Créez un fichier de répertoire de configuration du service supplémentaire **mcelog** et insérez le nom et la priorité de la politique dans ce fichier :

```
# cat << EOF > /etc/systemd/system/mcelog.system.d/priority.conf

[SERVICE]
CPUSchedulingPolicy=fifo
CPUSchedulingPriority=20
EOF
```

3. Recharger la configuration des scripts **systemd**:

```
# systemctl daemon-reload
```

4. Redémarrez le service **mcelog**:

```
# systemctl restart mcelog
```

Verification steps

- Afficher la priorité **mcelog** fixée par **systemd** issue :

```
# tuna -t mcelog -P
```

```
thread       ctxt_switches
  pid SCHED_ rtpri affinity voluntary nonvoluntary            cmd
826     FIFO    20  0,1,2,3        13             0          mcelog
```

Ressources supplémentaires

- **systemd(1)** et **tuna(8)** pages de manuel
- Description de la fourchette de priorité

30.8. CARTE DES PRIORITÉS

Les priorités sont définies par groupes, certains groupes étant dédiés à certaines fonctions du noyau. Pour les politiques d'ordonnancement en temps réel, un nombre entier compris entre **1** (priorité la plus faible) et **99** (priorité la plus élevée) peut être utilisé.

Le tableau suivant décrit la plage de priorités qui peut être utilisée pour définir la politique d'ordonnancement d'un processus.

Tableau 30.2. Description de la fourchette de priorité

Priorité	Fils	Description
1	Fils du noyau à faible priorité	Cette priorité est généralement réservée aux tâches qui doivent se situer juste au-dessus de **SCHED_OTHER**.
2 - 49	Disponible pour utilisation	La plage utilisée pour les priorités d'application typiques.
50	Valeur hard-IRQ par défaut	
51 - 98	Fils hautement prioritaires	Utilisez cette plage pour les threads qui s'exécutent périodiquement et qui doivent avoir des temps de réponse rapides. N'utilisez pas cette plage pour les threads liés au processeur, car vous risquez d'interrompre les interruptions.
99	Chiens de garde et migration	Les threads du système qui doivent être exécutés avec la priorité la plus élevée.

30.9. PROFIL DE PARTITIONNEMENT DU PROCESSEUR TUNED

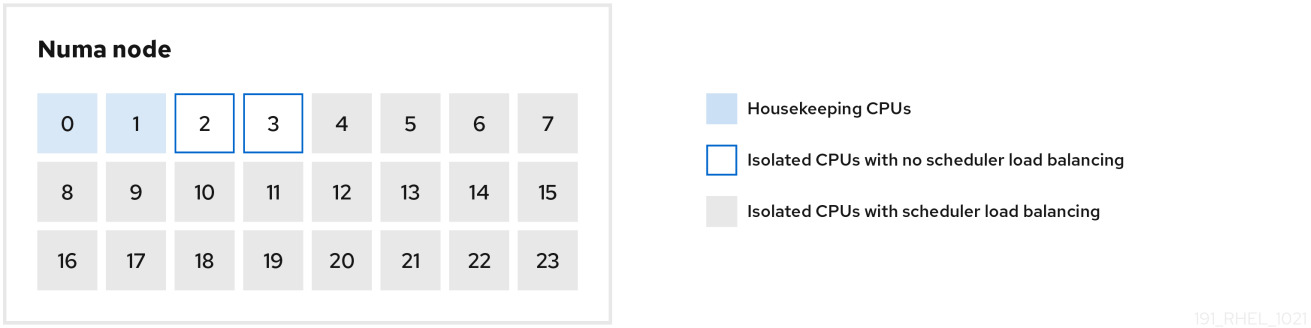
Pour régler Red Hat Enterprise Linux 9 pour les charges de travail sensibles à la latence, Red Hat recommande d'utiliser le profil **cpu-partitioning** TuneD.

Avant Red Hat Enterprise Linux 9, la documentation Red Hat sur les faibles temps de latence décrivait

les nombreuses étapes de bas niveau nécessaires à l'obtention d'un réglage des faibles temps de latence. Dans Red Hat Enterprise Linux 9, vous pouvez effectuer un réglage de faible latence plus efficacement en utilisant le profil **cpu-partitioning** TuneD. Ce profil est facilement personnalisable en fonction des exigences des applications individuelles à faible latence.

La figure suivante est un exemple d'utilisation du profil **cpu-partitioning**. Cet exemple utilise la disposition de l'unité centrale et des nœuds.

Figure 30.1. Figure partitionnement du processeur



Vous pouvez configurer le profil de partitionnement du processeur dans le fichier **/etc/tuned/cpu-partitioning-variables.conf** à l'aide des options de configuration suivantes :

CPU isolés avec répartition de la charge

Dans la figure de partitionnement des processeurs, les blocs numérotés de 4 à 23 sont les processeurs isolés par défaut. L'équilibrage de la charge des processus de l'ordonnanceur du noyau est activé sur ces CPU. Il est conçu pour les processus à faible latence avec plusieurs threads qui ont besoin de l'équilibrage de la charge du planificateur du noyau.
Vous pouvez configurer le profil de partitionnement des processeurs dans le fichier **/etc/tuned/cpu-partitioning-variables.conf** à l'aide de l'option **isolated_cores=cpu-list**, qui répertorie les processeurs à isoler qui utiliseront l'équilibrage de charge de l'ordonnanceur du noyau.

La liste des unités centrales isolées est séparée par des virgules ou vous pouvez spécifier une plage à l'aide d'un tiret, comme **3-5**. Cette option est obligatoire. Toute unité centrale absente de cette liste est automatiquement considérée comme une unité centrale de maintenance.

CPU isolés sans répartition de la charge

Dans la figure de partitionnement des CPU, les blocs numérotés 2 et 3 sont les CPU isolés qui ne fournissent pas d'équilibrage supplémentaire de la charge des processus de l'ordonnanceur du noyau.
Vous pouvez configurer le profil de partitionnement des processeurs dans le fichier **/etc/tuned/cpu-partitioning-variables.conf** à l'aide de l'option **no_balance_cores=cpu-list**, qui répertorie les processeurs à isoler qui n'utiliseront pas l'équilibrage de charge de l'ordonnanceur du noyau.

La spécification de l'option **no_balance_cores** est facultative, mais tous les processeurs de cette liste doivent être un sous-ensemble des processeurs figurant dans la liste **isolated_cores**.

Les threads d'application qui utilisent ces CPU doivent être épinglés individuellement à chaque CPU.

Unité centrale d'entretien

Toute unité centrale qui n'est pas isolée dans le fichier **cpu-partitioning-variables.conf** est automatiquement considérée comme une unité centrale de maintenance. Sur ces unités centrales, tous les services, démons, processus utilisateur, threads mobiles du noyau, gestionnaires d'interruption et temporisateurs du noyau sont autorisés à s'exécuter.

Ressources supplémentaires

- **tuned-profiles-cpu-partitioning(7)** page de manuel

30.10. UTILISATION DU PROFIL DE PARTITIONNEMENT DU PROCESSEUR TUNED POUR UN RÉGLAGE À FAIBLE LATENCE

Cette procédure décrit comment régler un système pour une faible latence en utilisant le profil **cpu-partitioning** de TuneD. Elle utilise l'exemple d'une application à faible latence qui peut utiliser **cpu-partitioning** et la disposition du processeur comme indiqué dans la figure de partitionnement du processeur.

Dans ce cas, l'application utilise :

- Un thread de lecture dédié, qui lit les données du réseau, sera placé sur l'unité centrale 2.
- Un grand nombre de threads qui traitent ces données réseau seront épinglés sur les CPU 4-23.
- Un thread d'écriture dédié qui écrit les données traitées sur le réseau sera placé sur l'unité centrale 3.

Conditions préalables

- Vous avez installé le profil TuneD **cpu-partitioning** en utilisant la commande **dnf install tuned-profiles-cpu-partitioning** en tant que root.

Procédure

1. Modifiez le fichier **/etc/tuned/cpu-partitioning-variables.conf** et ajoutez les informations suivantes :

```
# Isolated CPUs with the kernel's scheduler load balancing:
isolated_cores=2-23
# Isolated CPUs without the kernel's scheduler load balancing:
no_balance_cores=2,3
```

2. Définir le profil **cpu-partitioning** TuneD :

```
# tuned-adm profile cpu-partitioning
```

3. Reboot
   Après le redémarrage, le système est réglé pour une faible latence, conformément à l'isolation dans la figure de partitionnement des processeurs. L'application peut utiliser taskset pour affecter les threads de lecture et d'écriture aux CPU 2 et 3, et les threads d'application restants aux CPU 4 à 23.

Ressources supplémentaires

- **tuned-profiles-cpu-partitioning(7)** page de manuel

30.11. PERSONNALISATION DU PROFIL TUNED DE PARTITIONNEMENT DU PROCESSEUR

Vous pouvez étendre le profil TuneD pour apporter des modifications supplémentaires à l'accord.

Par exemple, le profil **cpu-partitioning** configure les unités centrales pour qu'elles utilisent **cstate=1**. Afin d'utiliser le profil **cpu-partitioning** mais de changer en plus l'état c du CPU de cstate1 à cstate0, la procédure suivante décrit un nouveau profil TuneD nommé *my_profile*, qui hérite du profil **cpu-partitioning** et définit ensuite l'état C-0.

Procédure

1. Créez le répertoire **/etc/tuned/my_profile**:

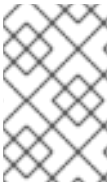
```
# mkdir /etc/tuned/my_profile
```

2. Créez un fichier **tuned.conf** dans ce répertoire et ajoutez-y le contenu suivant :

```
# vi /etc/tuned/my_profile/tuned.conf
[main]
summary=Customized tuning on top of cpu-partitioning
include=cpu-partitioning
[cpu]
force_latency=cstate.id:0|1
```

3. Utiliser le nouveau profil :

```
# tuned-adm profile my_profile
```



NOTE

Dans l'exemple partagé, un redémarrage n'est pas nécessaire. Toutefois, si les modifications apportées au profil *my_profile* nécessitent un redémarrage pour être prises en compte, redémarrez votre machine.

Ressources supplémentaires

- **tuned-profiles-cpu-partitioning(7)** page de manuel

CHAPITRE 31. OPTIMISER LES PERFORMANCES DU RÉSEAU

Le réglage des paramètres du réseau est un processus complexe qui doit prendre en compte de nombreux facteurs. Par exemple, cela inclut l'architecture CPU-mémoire, le nombre de cœurs de CPU, etc. Red Hat Enterprise Linux utilise des paramètres par défaut qui sont optimisés pour la plupart des scénarios. Cependant, dans certains cas, il peut être nécessaire d'ajuster les paramètres du réseau afin d'augmenter le débit ou la latence ou de résoudre des problèmes, tels que des chutes de paquets.

31.1. RÉGLAGE DES PARAMÈTRES DE LA CARTE RÉSEAU

Dans les réseaux à haut débit de 40 Gbps et plus, certaines valeurs par défaut des paramètres du noyau liés à la carte réseau peuvent être à l'origine de chutes de paquets et d'une dégradation des performances. L'ajustement de ces paramètres peut éviter de tels problèmes.

31.1.1. Augmentation des tampons de l'anneau pour réduire un taux élevé de perte de paquets

Les tampons d'anneau de réception sont partagés entre le pilote du périphérique et le contrôleur d'interface réseau (NIC). La carte attribue un tampon circulaire de transmission (TX) et de réception (RX). Comme son nom l'indique, le ring buffer est un tampon circulaire dans lequel un débordement écrase les données existantes. Il existe deux façons de transférer des données de la carte d'interface réseau au noyau : les interruptions matérielles et les interruptions logicielles, également appelées SoftIRQ.

Le noyau utilise le tampon de l'anneau RX pour stocker les paquets entrants jusqu'à ce qu'ils puissent être traités par le pilote de périphérique. Le pilote de périphérique vide l'anneau RX, généralement à l'aide de SoftIRQ, ce qui place les paquets entrants dans une structure de données du noyau appelée **sk_buff** ou **skb** pour commencer son voyage à travers le noyau et jusqu'à l'application qui possède la prise correspondante.

Le noyau utilise le tampon circulaire TX pour contenir les paquets sortants destinés au réseau. Ces tampons en anneau se trouvent au bas de la pile et constituent un point crucial où des paquets peuvent être abandonnés, ce qui affecte négativement les performances du réseau.

Augmentez la taille des tampons en anneau d'un périphérique Ethernet si le taux de perte de paquets entraîne des pertes de données, des dépassements de délai ou d'autres problèmes pour les applications.

Procédure

1. Affiche les statistiques de chute de paquets de l'interface :

```
# ethtool -S enp1s0
    ...
    rx_queue_0_drops: 97326
    rx_queue_1_drops: 63783
    ...
```

   Notez que la sortie de la commande dépend de la carte réseau et du pilote.

   Des valeurs élevées dans les compteurs **discard** ou **drop** indiquent que le tampon disponible se remplit plus rapidement que le noyau ne peut traiter les paquets. L'augmentation des tampons de l'anneau peut aider à éviter de telles pertes.

2. Affiche la taille maximale des tampons de l'anneau :

```
# ethtool -g enp1s0
Ring parameters for enp1s0:
Pre-set maximums:
RX:             4096
RX Mini:        0
RX Jumbo:       16320
TX:             4096
Current hardware settings:
RX:             255
RX Mini:        0
RX Jumbo:       0
TX:             255
```

Si les valeurs de la section **Pre-set maximums** sont plus élevées que celles de la section **Current hardware settings**, vous pouvez modifier les paramètres dans les étapes suivantes.

3. Identifier le profil de connexion NetworkManager qui utilise l'interface :

```
# nmcli connection show
NAME                UUID                                  TYPE      DEVICE
Example-Connection  a5eb6490-cc20-3668-81f8-0314a27f3f75  ethernet  enp1s0
```

4. Mettez à jour le profil de connexion et augmentez les tampons de l'anneau :

   - Pour augmenter la mémoire tampon de l'anneau RX, entrez :

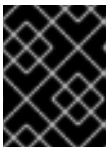
     ```
     # nmcli connection modify Example-Connection ethtool.ring-rx 4096
     ```

   - Pour augmenter la mémoire tampon de l'anneau TX, entrez :

     ```
     # nmcli connection modify Example-Connection ethtool.ring-tx 4096
     ```

5. Recharger la connexion au NetworkManager :

```
# nmcli connection up Example-Connection
```



IMPORTANT

Selon le pilote utilisé par votre carte d'interface réseau, un changement dans la mémoire tampon de l'anneau peut interrompre brièvement la connexion réseau.

Ressources supplémentaires

- les commandes ifconfig et ip signalent les chutes de paquets
- Dois-je m'inquiéter d'un taux d'abandon de paquets de 0,05 % ?
- **ethtool(8)** page de manuel

31.1.2. Optimisation de la file d'attente des périphériques de réseau pour éviter les chutes de paquets

Lorsqu'une carte réseau reçoit des paquets et avant que la pile de protocoles du noyau ne les traite, le noyau stocke ces paquets dans des files d'attente. Le noyau maintient une file d'attente distincte pour chaque cœur de processeur.

Si la file d'attente d'un noyau est pleine, le noyau abandonne tous les autres paquets entrants que la fonction du noyau **netif_receive_skb()** assigne à cette file. Si le serveur contient une carte réseau de 10 Gbps ou plus, ou plusieurs cartes de 1 Gbps, réglez la taille de la file d'attente pour éviter ce problème.

Conditions préalables

- Un adaptateur réseau de 10 Gbps ou plus rapide ou plusieurs adaptateurs réseau de 1 Gbps

Procédure

1. Déterminer s'il est nécessaire d'ajuster la file d'attente de l'arriéré, afficher les compteurs dans le fichier **/proc/net/softnet_stat**:

```
# awk '{for (i=1; i<=NF; i++) printf strtonum("0x" $i) (i==NF?"\n":" ")}'
/proc/net/softnet_stat | column -t
221951548  0      0  0  0  0  0  0  0  0  0  0  0
192058677  18862  0  0  0  0  0  0  0  0  0  0  1
455324886  0      0  0  0  0  0  0  0  0  0  0  2
...
```

   La commande **awk** convertit les valeurs de **/proc/net/softnet_stat** du format hexadécimal au format décimal et les affiche sous forme de tableau. Chaque ligne représente un cœur de CPU en commençant par le cœur 0.

   Les colonnes concernées sont les suivantes :

   - Première colonne : Nombre total de trames reçues
   - Deuxième colonne : Nombre de trames abandonnées en raison d'une file d'attente pleine
   - Dernière colonne : Le numéro du cœur de l'unité centrale

2. Si les valeurs de la deuxième colonne du fichier **/proc/net/softnet_stat** augmentent avec le temps, augmentez la taille de la file d'attente :

   a. Affiche la taille actuelle de la file d'attente de l'arriéré :

```
# sysctl net.core.netdev_max_backlog
net.core.netdev_max_backlog = 1000
```

   b. Créez le fichier **/etc/sysctl.d/10-netdev_max_backlog.conf** avec le contenu suivant :

```
net.core.netdev_max_backlog = 2000
```

      Fixer le paramètre **net.core.netdev_max_backlog** au double de la valeur actuelle.

   c. Charger les paramètres du fichier **/etc/sysctl.d/10-netdev_max_backlog.conf**:

```
# sysctl -p /etc/sysctl.d/10-netdev_max_backlog.conf
```

Vérification

- Contrôler la deuxième colonne du fichier **/proc/net/softnet_stat**:

```
# awk '{for (i=1; i<=NF; i ) printf strtonum("0x" $i) (i==NF?"\n":" ")}'
/proc/net/softnet_stat | column -t
```

Si les valeurs continuent d'augmenter, doublez à nouveau la valeur de **net.core.netdev_max_backlog**. Répétez ce processus jusqu'à ce que les compteurs de chutes de paquets n'augmentent plus.

31.1.3. Augmenter la longueur de la file d'attente de transmission d'un NIC pour réduire le nombre d'erreurs de transmission

Le noyau stocke les paquets dans une file d'attente avant de les transmettre. La longueur par défaut (1000 paquets) est généralement suffisante pour les réseaux à 10 Gbps et souvent aussi pour les réseaux à 40 Gbps. Cependant, dans les réseaux plus rapides, ou si vous rencontrez un nombre croissant d'erreurs de transmission sur un adaptateur, augmentez la longueur de la file d'attente.

Procédure

1. Affiche la longueur de la file d'attente d'émission actuelle :

```
# ip -s link show enp1s0
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
mode DEFAULT group default qlen 1000
...
```

Dans cet exemple, la longueur de la file d'attente de transmission (**qlen**) de l'interface **enp1s0** est **1000**.

2. Surveillez le compteur de paquets abandonnés de la file d'attente de transmission logicielle d'une interface réseau :

```
# tc -s qdisc show dev enp1s0
qdisc fq_codel 0: root refcnt 2 limit 10240p flows 1024 quantum 1514 target 5ms interval
100ms memory_limit 32Mb ecn drop_batch 64
 Sent 16889923 bytes 426862765 pkt (dropped 191980, overlimits 0 requeues 2)
 ...
```

3. Si le nombre d'erreurs de transmission est élevé ou en augmentation, définissez une longueur de file d'attente de transmission plus élevée :

   a. Identifier le profil de connexion NetworkManager qui utilise cette interface :

```
# nmcli connection show
NAME               UUID                                  TYPE      DEVICE
Example-Connection  a5eb6490-cc20-3668-81f8-0314a27f3f75  ethernet  enp1s0
```

   b. Créez le script **/etc/NetworkManager/dispatcher.d/99-set-tx-queue-length-up** NetworkManager dispatcher avec le contenu suivant :

```
#!/bin/bash
# Set TX queue length on enp1s0 to 2000
```

```
if [ "$1" == "enp1s0" ] && [ "$2" == "up" ] ; then
    ip link set dev enp1s0 txqueuelen 2000
fi
```

c. Définir le bit exécutable sur le fichier **/etc/NetworkManager/dispatcher.d/99-set-tx-queue-length-up**:

```
# chmod x /etc/NetworkManager/dispatcher.d/99-set-tx-queue-length-up
```

d. Recharger la connexion au NetworkManager :

```
# nmcli connection up Example-Connection
```

Vérification

1. Affiche la longueur de la file d'attente d'émission :

```
# ip -s link show enp1s0
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
mode DEFAULT group default qlen 2000
...
```

2. Surveiller le compteur de paquets abandonnés :

```
# tc -s qdisc show dev enp1s0
```

Si le compteur **dropped** continue d'augmenter, doublez à nouveau la longueur de la file d'attente d'émission. Répétez ce processus jusqu'à ce que le compteur n'augmente plus.

31.2. RÉGLAGE DE L'ÉQUILIBRAGE DES IRQ

Sur les hôtes multicœurs, vous pouvez augmenter les performances en vous assurant que Red Hat Enterprise Linux équilibre les files d'attente d'interruption (IRQ) afin de distribuer les interruptions entre les cœurs du CPU.

31.2.1. Interruptions et gestionnaires d'interruptions

Lorsqu'un contrôleur d'interface réseau (NIC) reçoit des données, il les copie dans les tampons du noyau en utilisant l'accès direct à la mémoire (DMA). La carte d'interface réseau informe ensuite le noyau de ces données en déclenchant une interruption matérielle. Ces interruptions sont traitées par des gestionnaires d'interruption qui effectuent un travail minimal, car ils ont déjà interrompu une autre tâche et les gestionnaires ne peuvent pas s'interrompre eux-mêmes. Les interruptions matérielles peuvent être coûteuses en termes d'utilisation du processeur, en particulier si elles utilisent des verrous du noyau.

Le gestionnaire d'interruptions matérielles laisse alors la majorité de la réception des paquets à un processus de demande d'interruption logicielle (SoftIRQ). Le noyau peut programmer ces processus de manière plus équitable.

Exemple 31.1. Affichage des interruptions matérielles

Le noyau stocke les compteurs d'interruptions dans le fichier **/proc/interrupts**. Pour afficher les compteurs d'un NIC spécifique, tel que **enp1s0**, entrez :

```
# egrep "CPU|enp1s0" /proc/interrupts
       CPU0    CPU1    CPU2    CPU3    CPU4   CPU5
105:  141606      0       0       0       0      0  IR-PCI-MSI-edge      enp1s0-rx-0
106:       0  141091       0       0       0      0  IR-PCI-MSI-edge      enp1s0-rx-1
107:       2       0  163785       0       0      0  IR-PCI-MSI-edge      enp1s0-rx-2
108:       3       0       0  194370       0      0  IR-PCI-MSI-edge      enp1s0-rx-3
109:       0       0       0       0       0      0  IR-PCI-MSI-edge      enp1s0-tx
```

Un vecteur d'interruption est attribué à chaque file d'attente dans la première colonne. Le noyau initialise ces vecteurs lorsque le système démarre ou lorsqu'un utilisateur charge le module du pilote NIC. Chaque file d'attente de réception (**RX**) et de transmission (**TX**) se voit attribuer un vecteur unique qui indique au gestionnaire d'interruption de quelle carte ou file d'attente l'interruption provient. Les colonnes représentent le nombre d'interruptions entrantes pour chaque cœur de CPU.

31.2.2. Demandes d'interruption logicielle

Les demandes d'interruption logicielle (SoftIRQ) effacent les tampons de l'anneau de réception des adaptateurs réseau. Le noyau planifie l'exécution des routines SoftIRQ à un moment où les autres tâches ne seront pas interrompues. Sur Red Hat Enterprise Linux, les processus nommés **ksoftirqd/*cpu-number*** exécutent ces routines et appellent des fonctions de code spécifiques au pilote.

Pour surveiller les compteurs SoftIRQ pour chaque cœur de CPU, entrez :

```
# watch -n1 'egrep "CPU|NET_RX|NET_TX" /proc/softirqs'
                CPU0       CPU1   CPU2       CPU3 CPU4    CPU5       CPU6  CPU7
    NET_TX:    49672      52610  28175      97288      12633   19843      18746     220689
    NET_RX:       96       1615       789        46        31    1735       1315     470798
```

La commande met à jour dynamiquement la sortie. Appuyer sur **Ctrl**+**C** pour interrompre la sortie.

31.2.3. NAPI Polling

La nouvelle API (NAPI) est une extension du cadre de traitement des paquets du pilote de périphérique visant à améliorer l'efficacité des paquets réseau entrants. Les interruptions matérielles sont coûteuses car elles entraînent généralement un changement de contexte de l'espace du noyau à l'espace utilisateur et vice-versa, et ne peuvent pas s'interrompre elles-mêmes. Même avec la coalescence des interruptions, le gestionnaire d'interruptions monopolise complètement un cœur de processeur. Avec la NAPI, le pilote peut utiliser un mode d'interrogation au lieu d'être interrompu par le noyau pour chaque paquet reçu.

En fonctionnement normal, le noyau émet une première interruption dure, suivie d'un gestionnaire de demande d'interruption douce (SoftIRQ) qui interroge la carte réseau à l'aide de routines NAPI. Pour éviter que les SoftIRQ ne monopolisent un cœur de processeur, la routine d'interrogation dispose d'un budget qui détermine le temps de processeur que la SoftIRQ peut consommer. Une fois la routine d'interrogation SoftIRQ terminée, le noyau quitte la routine et la programme pour qu'elle s'exécute à nouveau ultérieurement afin de répéter le processus de réception des paquets de la carte réseau.

31.2.4. Le service irqbalance

Sur les systèmes avec ou sans architecture NUMA (Non-Uniform Memory Access), le service **irqbalance** équilibre efficacement les interruptions entre les cœurs de l'unité centrale, en fonction des conditions du système. Le service **irqbalance** s'exécute en arrière-plan et surveille la charge du

processeur toutes les 10 secondes. Il déplace les interruptions vers d'autres cœurs de processeur lorsque la charge d'un processeur est trop élevée. En conséquence, le système fonctionne bien et gère la charge plus efficacement.

Si **irqbalance** n'est pas en cours d'exécution, c'est généralement le noyau 0 de l'unité centrale qui gère la plupart des interruptions. Même en cas de charge modérée, ce cœur d'unité centrale peut devenir très occupé en essayant de gérer la charge de travail de tout le matériel du système. Par conséquent, les interruptions ou les travaux basés sur les interruptions peuvent être manqués ou retardés. Cela peut entraîner une baisse des performances du réseau et du stockage, une perte de paquets et, éventuellement, d'autres problèmes.

IMPORTANT

La désactivation de **irqbalance** peut avoir un impact négatif sur le débit du réseau.

Sur les systèmes dotés d'un seul cœur de processeur, le service **irqbalance** n'apporte aucun avantage et s'arrête de lui-même.

Par défaut, le service **irqbalance** est activé et fonctionne sous Red Hat Enterprise Linux. Pour réactiver le service si vous l'avez désactivé, entrez :

```
# systemctl enable --now irqbalance
```

Ressources supplémentaires

- Avons-nous besoin d'irqbalance ? solution
- Comment dois-je configurer les canaux IRQ de l'interface réseau ? solution

31.2.5. Augmentation de la durée d'exécution des SoftIRQ sur l'unité centrale

Si les SoftIRQ ne durent pas assez longtemps, le taux de données entrantes peut dépasser la capacité du noyau à vider la mémoire tampon assez rapidement. Par conséquent, les tampons du contrôleur d'interface réseau (NIC) débordent et des paquets sont perdus.

Si les processus **softirqd** n'ont pas pu récupérer tous les paquets des interfaces en un cycle d'interrogation de l'interface NAPI, cela indique que les SoftIRQ n'ont pas assez de temps CPU. Cela peut être le cas sur des hôtes avec des cartes d'interface réseau rapides, telles que 10 Gbps ou plus. Si vous augmentez les valeurs des paramètres du noyau **net.core.netdev_budget** et **net.core.netdev_budget_usecs**, vous pouvez contrôler le temps et le nombre de paquets que **softirqd** peut traiter dans un cycle d'interrogation.

Procédure

1. Pour déterminer s'il est nécessaire d'ajuster le paramètre **net.core.netdev_budget**, affichez les compteurs du fichier **/proc/net/softnet_stat**:

```
# awk '{for (i=1; i<=NF; i++) printf strtonum("0x" $i) (i==NF?"\n":" ")}'
/proc/net/softnet_stat | column -t
221951548  0  0      0  0  0  0  0  0  0  0  0  0
192058677  0  20380  0  0  0  0  0  0  0  0  0  1
455324886  0  0      0  0  0  0  0  0  0  0  0  2
...
```

La commande **awk** convertit les valeurs de **/proc/net/softnet_stat** du format hexadécimal au format décimal et les affiche sous forme de tableau. Chaque ligne représente un cœur de CPU en commençant par le cœur 0.

Les colonnes concernées sont les suivantes :

- Première colonne : Nombre total de trames reçues.
- Troisième colonne : Nombre de fois où les processus **softirqd** n'ont pas pu récupérer tous les paquets des interfaces au cours d'un cycle d'interrogation de la NAPI.
- Dernière colonne : Le numéro du cœur de l'unité centrale.

2. Si les compteurs de la troisième colonne du fichier **/proc/net/softnet_stat** augmentent avec le temps, réglez le système :

   a. Affiche les valeurs actuelles des paramètres **net.core.netdev_budget_usecs** et **net.core.netdev_budget**:

   ```
   # sysctl net.core.netdev_budget_usecs net.core.netdev_budget
   net.core.netdev_budget_usecs = 2000
   net.core.netdev_budget = 300
   ```

   Avec ces paramètres, les processus **softirqd** ont jusqu'à 2000 microsecondes pour traiter jusqu'à 300 messages du NIC dans un cycle d'interrogation. L'interrogation se termine en fonction de la condition qui est remplie en premier.

   b. Créez le fichier **/etc/sysctl.d/10-netdev_budget.conf** avec le contenu suivant :

   ```
   net.core.netdev_budget = 600
   net.core.netdev_budget_usecs = 4000
   ```

   Fixe les paramètres à un double de leurs valeurs actuelles.

   c. Charger les paramètres du fichier **/etc/sysctl.d/10-netdev_budget.conf**:

   ```
   # sysctl -p /etc/sysctl.d/10-netdev_budget.conf
   ```

Vérification

- Contrôler la troisième colonne du fichier **/proc/net/softnet_stat**:

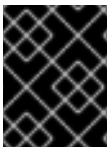
  ```
  # awk '{for (i=1; i<=NF; i ) printf strtonum("0x" $i) (i==NF?"\n":" ")}'
  /proc/net/softnet_stat | column -t
  ```

  Si les valeurs continuent d'augmenter, réglez **net.core.netdev_budget_usecs** et **net.core.netdev_budget** à des valeurs plus élevées. Répétez ce processus jusqu'à ce que les compteurs n'augmentent plus.

31.3. AMÉLIORER LA LATENCE DU RÉSEAU

Les fonctions de gestion de l'alimentation du processeur peuvent entraîner des retards indésirables dans le traitement des applications sensibles au facteur temps. Vous pouvez désactiver tout ou partie de ces fonctions de gestion de l'alimentation pour améliorer la latence du réseau.

Par exemple, si la latence est plus élevée lorsque le serveur est inactif que lorsqu'il est fortement sollicité, les paramètres de gestion de l'alimentation du processeur peuvent influer sur la latence.



IMPORTANT

La désactivation des fonctions de gestion de l'alimentation du processeur peut entraîner une augmentation de la consommation d'énergie et une perte de chaleur.

31.3.1. Comment les états de puissance de l'unité centrale influencent la latence du réseau

Les états de consommation (C-states) des CPU optimisent et réduisent la consommation d'énergie des ordinateurs. Les états C sont numérotés, en commençant par C0. En C0, le processeur est entièrement alimenté et en cours d'exécution. En C1, le processeur est entièrement alimenté mais n'est pas en cours d'exécution. Plus le numéro de l'état C est élevé, plus le processeur éteint de composants.

Chaque fois qu'un cœur de processeur est inactif, la logique d'économie d'énergie intégrée intervient et tente de faire passer le cœur de l'état C actuel à un état supérieur en éteignant divers composants du processeur. Si le cœur du processeur doit traiter des données, Red Hat Enterprise Linux (RHEL) envoie une interruption au processeur pour réveiller le cœur et ramener son état C à C0.

Sortir des états C profonds pour revenir à C0 prend du temps en raison de la remise sous tension des différents composants du processeur. Sur les systèmes multicœurs, il peut également arriver que de nombreux cœurs soient simultanément inactifs et, par conséquent, dans des états C profonds. Si RHEL tente de les réveiller en même temps, le noyau peut générer un grand nombre d'interruptions inter-processeurs (IPI) pendant que tous les cœurs reviennent d'états C profonds. En raison du verrouillage requis lors du traitement des interruptions, le système peut alors se bloquer pendant un certain temps pour traiter toutes les interruptions. Cela peut entraîner des retards importants dans la réponse de l'application aux événements.

Exemple 31.2. Affichage des temps dans l'état C par cœur

La page **Idle Stats** de l'application PowerTOP affiche le temps que les cœurs de l'unité centrale passent dans chaque état C :

```
        Pkg(HW)  |          Core(HW) |          CPU(OS) 0   CPU(OS) 4
                 |                   | C0 active   2.5%       2.2%
                 |                   | POLL        0.0%    0.0 ms  0.0%    0.1 ms
                 |                   | C1          0.1%    0.2 ms  0.0%    0.1 ms
C2 (pc2)   63.7%    |                 |
C3 (pc3)    0.0%    | C3 (cc3)    0.1%    | C3          0.1%    0.1 ms  0.1%    0.1 ms
C6 (pc6)    0.0%    | C6 (cc6)    8.3%    | C6          5.2%    0.6 ms  6.0%    0.6 ms
C7 (pc7)    0.0%    | C7 (cc7)   76.6%    | C7s         0.0%    0.0 ms  0.0%    0.0 ms
C8 (pc8)    0.0%    |                   | C8          6.3%    0.9 ms  5.8%    0.8 ms
C9 (pc9)    0.0%    |                   | C9          0.4%    3.7 ms  2.2%    2.2 ms
C10 (pc10)  0.0%    |                   |
                 |                   | C10        80.8%    3.7 ms 79.4%    4.4 ms
                 |                   | C1E         0.1%    0.1 ms  0.1%    0.1 ms
...
```

Ressources supplémentaires

- Gérer la consommation d'énergie avec PowerTOP

31.3.2. Paramètres de l'état C dans le micrologiciel EFI

Dans la plupart des systèmes dotés d'un microprogramme EFI, vous pouvez activer et désactiver les différents états de consommation (C-states). Cependant, sur Red Hat Enterprise Linux (RHEL), le pilote de veille détermine si le noyau utilise les paramètres du microprogramme :

- **intel_idle**: Il s'agit du pilote par défaut sur les hôtes dotés d'un processeur Intel, qui ignore les paramètres d'état C du micrologiciel EFI.
- **acpi_idle**: RHEL utilise ce pilote sur les hôtes équipés de processeurs de fournisseurs autres qu'Intel et si **intel_idle** est désactivé. Par défaut, le pilote **acpi_idle** utilise les paramètres d'état C du micrologiciel EFI.

Ressources supplémentaires

- **/usr/share/doc/kernel-doc-*<version>*/Documentation/admin-guide/pm/cpuidle.rst** fournie par le paquet **kernel-doc**

31.3.3. Désactivation des états C par l'utilisation d'un profil TuneD personnalisé

Le service TuneD utilise l'interface Power Management Quality of Service (**PMQOS**) du noyau pour définir le verrouillage des états de consommation (C-states). Le pilote d'inactivité du noyau peut communiquer avec cette interface pour limiter dynamiquement les états C. Cela évite aux administrateurs de devoir coder en dur une valeur maximale d'état C en utilisant le pilote d'inactivité du noyau. Cela évite aux administrateurs de devoir coder en dur une valeur maximale d'état C en utilisant les paramètres de la ligne de commande du noyau.

Conditions préalables

- Le paquet **tuned** est installé.
- Le service **tuned** est activé et fonctionne.

Procédure

1. Afficher le profil actif :

```
# tuned-adm active
Current active profile: network-latency
```

2. Créez un répertoire pour le profil TuneD personnalisé :

```
# mkdir /etc/tuned/network-latency-custom/
```

3. Créez le fichier **/etc/tuned/*network-latency-custom*/tuned.conf** avec le contenu suivant :

```
[main]
include=network-latency

[cpu]
force_latency=cstate.id:1|2
```

Ce profil personnalisé hérite de tous les paramètres du profil **network-latency**. Le paramètre **force_latency** TuneD spécifie la latence en microsecondes (µs). Si la latence de l'état C est

supérieure à la valeur spécifiée, le pilote de ralenti de Red Hat Enterprise Linux empêche le CPU de passer à un état C supérieur. Avec **force_latency=cstate.id:1|2**, TuneD vérifie d'abord si le répertoire **/sys/devices/system/cpu/cpu_<number>_/cpuidle/state_<cstate.id>_/** existe. Dans ce cas, TuneD lit la valeur de latence à partir du fichier **latency** dans ce répertoire. Si le répertoire n'existe pas, TuneD utilise 2 microsecondes comme valeur de repli.

4. Activer le ***network-latency-custom*** profil :

```
# tuned-adm profile network-latency-custom
```

Ressources supplémentaires

- Démarrer avec TuneD
- Personnalisation des profils TuneD

31.3.4. Désactivation des états C à l'aide d'une option de la ligne de commande du noyau

Les paramètres de ligne de commande du noyau **processor.max_cstate** et **intel_idle.max_cstat** configurent les états de consommation maximum (C-state) que les cœurs du CPU peuvent utiliser. Par exemple, en réglant les paramètres sur **1**, on s'assure que le processeur ne demandera jamais un état C inférieur à C1.

Utilisez cette méthode pour tester si la latence des applications sur un hôte est affectée par les états C. Pour ne pas coder en dur un état spécifique, envisagez d'utiliser une solution plus dynamique. Voir Désactiver les états C en utilisant un profil TuneD personnalisé .

Conditions préalables

- Le service **tuned** n'est pas en cours d'exécution ou est configuré pour ne pas mettre à jour les paramètres de l'état C.

Procédure

1. Affiche le pilote d'inactivité utilisé par le système :

```
# cat /sys/devices/system/cpu/cpuidle/current_driver
intel_idle
```

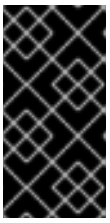
2. Si l'hôte utilise le pilote **intel_idle**, définissez le paramètre du noyau **intel_idle.max_cstate** pour définir l'état C le plus élevé que les cœurs de CPU doivent pouvoir utiliser :

```
# grubby --update-kernel=ALL --args="intel_idle.max_cstate=0"
```

Le réglage de **intel_idle.max_cstate=0** désactive le pilote **intel_idle**. Par conséquent, le noyau utilise le pilote **acpi_idle** qui utilise les valeurs d'état C définies dans le microprogramme EFI. Pour cette raison, définissez également **processor.max_cstate** pour remplacer ces paramètres d'état C.

3. Sur chaque hôte, indépendamment du fournisseur de l'unité centrale, définir l'état C le plus élevé que les cœurs de l'unité centrale devraient être en mesure d'utiliser :

```
# grubby --update-kernel=ALL --args="processor.max_cstate=0"
```



IMPORTANT

Si vous définissez **processor.max_cstate=0** en plus de **intel_idle.max_cstate=0**, le pilote **acpi_idle** remplace la valeur de **processor.max_cstate** et la fixe à **1**. Par conséquent, avec **processor.max_cstate=0 intel_idle.max_cstate=0**, l'état C le plus élevé que le noyau utilisera est C1, et non C0.

4. Redémarrez l'hôte pour que les modifications soient prises en compte :

```
# reboot
```

Vérification

1. Afficher l'état C maximal :

```
# cat /sys/module/processor/parameters/max_cstate
1
```

2. Si l'hôte utilise le pilote **intel_idle**, affichez l'état C maximum :

```
# cat /sys/module/intel_idle/parameters/max_cstate
0
```

Ressources supplémentaires

- Que sont les "C-states" du processeur et comment les désactiver si nécessaire ?
- **/usr/share/doc/kernel-doc-*<version>*/Documentation/admin-guide/pm/cpuidle.rst** fournie par le paquet **kernel-doc**

31.4. AMÉLIORER LE DÉBIT DE GRANDES QUANTITÉS DE FLUX DE DONNÉES CONTIGUS

Selon la norme IEEE 802.3, une trame Ethernet par défaut sans balise VLAN (Virtual Local Area Network) a une taille maximale de 1518 octets. Chacune de ces trames comprend un en-tête de 18 octets, ce qui laisse 1500 octets pour la charge utile. Par conséquent, pour chaque 1500 octets de données que le serveur transmet sur le réseau, 18 octets (1,2 %) d'en-tête de trame Ethernet sont surchargés et transmis également. Les en-têtes des protocoles des couches 3 et 4 augmentent encore le surdébit par paquet.

Envisagez d'utiliser des trames jumbo pour réduire la charge de travail si les hôtes de votre réseau envoient souvent de nombreux flux de données contigus, tels que les serveurs de sauvegarde ou les serveurs de fichiers hébergeant de nombreux fichiers volumineux. Les trames jumbo sont des trames non standardisées dont l'unité de transmission maximale (MTU) est supérieure à la taille de la charge utile Ethernet standard de 1 500 octets. Par exemple, si vous configurez des trames jumbo avec l'UTM maximale autorisée de 9000 octets de charge utile, la surcharge de chaque trame est réduite à 0,2 %.

En fonction du réseau et des services, il peut être intéressant de n'activer les trames jumbo que dans des parties spécifiques d'un réseau, comme le backend de stockage d'un cluster. Cela permet d'éviter la fragmentation des paquets.

31.4.1. Considérations à prendre en compte avant de configurer des trames jumbo

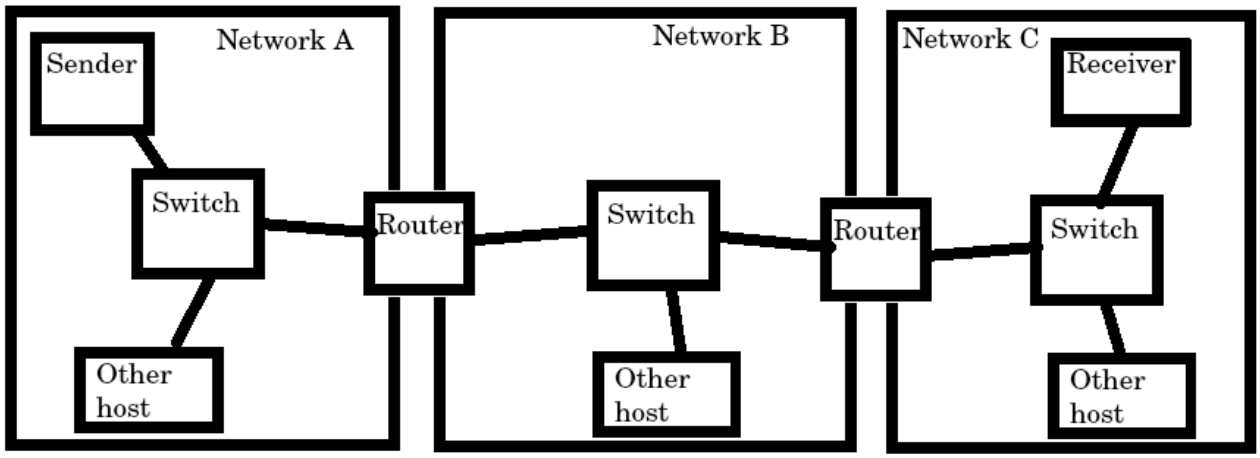
Selon le matériel, les applications et les services de votre réseau, les trames jumbo peuvent avoir différents impacts. Décidez avec soin si l'activation des trames jumbo présente un avantage dans votre scénario.

Conditions préalables

Tous les périphériques du réseau sur le chemin de transmission doivent prendre en charge les trames jumbo et utiliser la même taille d'unité de transmission maximale (MTU). Dans le cas contraire, vous risquez de rencontrer les problèmes suivants :

- Paquets abandonnés.
- Temps de latence plus élevé en raison de la fragmentation des paquets.
- Risque accru de perte de paquets causé par la fragmentation. Par exemple, si un routeur fragmente une trame unique de 9000 octets en six trames de 1500 octets et que l'une de ces trames de 1500 octets est perdue, la trame entière est perdue car elle ne peut pas être réassemblée.

Dans le diagramme suivant, tous les hôtes des trois sous-réseaux doivent utiliser le même MTU si un hôte du réseau A envoie un paquet à un hôte du réseau C :



Avantages des trames jumbo

- Débit plus élevé : Chaque trame contient plus de données utilisateur alors que la surcharge du protocole est fixe.
- Réduction de l'utilisation de l'unité centrale : Les trames Jumbo provoquent moins d'interruptions et permettent donc d'économiser des cycles de CPU.

Inconvénients des trames jumbo

- Temps de latence plus élevé : Les trames plus grandes retardent les paquets qui les suivent.
- Augmentation de l'utilisation de la mémoire tampon : Les trames plus grandes peuvent remplir plus rapidement la mémoire de la file d'attente.

31.4.2. Configuration du MTU dans un profil de connexion NetworkManager existant

Si votre réseau requiert une unité de transmission maximale (MTU) différente de celle par défaut, vous pouvez configurer ce paramètre dans le profil de connexion NetworkManager correspondant.

Les trames Jumbo sont des paquets réseau dont la charge utile est comprise entre 1500 et 9000 octets. Tous les appareils du même domaine de diffusion doivent prendre en charge ces trames.

Conditions préalables

- Tous les appareils du domaine de diffusion utilisent le même MTU.
- Vous connaissez le MTU du réseau.
- Vous avez déjà configuré un profil de connexion pour le réseau avec le MTU divergent.

Procédure

1. Facultatif : Affiche le MTU actuel :

```
# ip link show
...
3: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
mode DEFAULT group default qlen 1000
    link/ether 52:54:00:74:79:56 brd ff:ff:ff:ff:ff:ff
...
```

2. Optionnel : Afficher les profils de connexion de NetworkManager :

```
# nmcli connection show
NAME     UUID                                  TYPE      DEVICE
Example  f2f33f29-bb5c-3a07-9069-be72eaec3ecf  ethernet  enp1s0
...
```

3. Définissez le MTU dans le profil qui gère la connexion au réseau avec le MTU divergent :

```
# nmcli connection modify Example mtu 9000
```

4. Réactiver la connexion :

```
# nmcli connection up Example
```

Vérification

1. Affichez le paramètre MTU :

```
# ip link show
...
3: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc fq_codel state UP
mode DEFAULT group default qlen 1000
    link/ether 52:54:00:74:79:56 brd ff:ff:ff:ff:ff:ff
...
```

2. Vérifiez qu'aucun hôte sur les chemins de transmission ne fragmente les paquets :
   - Du côté du récepteur, afficher les statistiques de réassemblage IP du noyau :

```
# nstat -az IpReasm*
```

```
#kernel
IpReasmTimeout 0 0.0
IpReasmReqds 0 0.0
IpReasmOKs 0 0.0
IpReasmFails 0 0.0
```

Si les compteurs retournent **0**, les paquets n'ont pas été réassemblés.

- Du côté de l'expéditeur, transmettre une requête ICMP avec le bit d'interdiction de fragmentation :

```
# ping -c1 -Mdo -s 8972 destination_host
```

Si la commande réussit, le paquet n'a pas été fragmenté.

Calculez la valeur de l'option **-s** packet size comme suit : Taille MTU - 8 octets En-tête ICMP - 20 octets En-tête IPv4 = taille du paquet

31.5. OPTIMISATION DES CONNEXIONS TCP POUR UN DÉBIT ÉLEVÉ

Réglez les paramètres relatifs à TCP sur Red Hat Enterprise Linux afin d'augmenter le débit, de réduire la latence ou d'éviter les problèmes, tels que la perte de paquets.

31.5.1. Test du débit TCP à l'aide de iperf3

L'utilitaire **iperf3** propose un mode serveur et un mode client pour effectuer des tests de débit réseau entre deux hôtes.

NOTE

Le débit des applications dépend de nombreux facteurs, tels que la taille des tampons utilisés par l'application. Par conséquent, les résultats mesurés à l'aide d'utilitaires de test, tels que **iperf3**, peuvent être sensiblement différents de ceux des applications sur un serveur soumis à une charge de travail de production.

Conditions préalables

- Le paquet **iperf3** est installé à la fois sur le client et sur le serveur.
- Aucun autre service sur l'un ou l'autre hôte ne provoque un trafic réseau qui affecte substantiellement le résultat du test.
- Pour les connexions à 40 Gbps et plus, la carte réseau prend en charge la fonction ARFS (Accelerated Receive Flow Steering) et cette fonction est activée sur l'interface.

Procédure

1. Facultatif : Affichez la vitesse réseau maximale du contrôleur d'interface réseau (NIC) sur le serveur et le client :

```
# ethtool enp1s0 | grep "Speed"
  Speed: 100000Mb/s
```

2. Sur le serveur :

a. Ouvrez temporairement le port TCP 5201 par défaut de **iperf3** dans le service **firewalld**:

```
# firewall-cmd --add-port=5201/tcp
# firewall-cmd --reload
```

b. Démarrer **iperf3** en mode serveur :

```
# iperf3 --server
```

Le service attend maintenant les connexions entrantes des clients.

3. Sur le client :

a. Commencez à mesurer le débit :

```
# iperf3 --time 60 --zerocopy --client 192.0.2.1
```

- **--time *<seconds>***: Définit le temps en secondes pendant lequel le client arrête la transmission.
  Réglez ce paramètre à une valeur qui devrait fonctionner et augmentez-la lors des mesures ultérieures. Si le serveur envoie des paquets à une vitesse supérieure à celle que les dispositifs sur le chemin de transmission ou le client peuvent traiter, les paquets peuvent être abandonnés.

- **--zerocopy**: Active une méthode de copie zéro au lieu d'utiliser l'appel système **write()**. Cette option n'est nécessaire que si vous souhaitez simuler une application à copie zéro ou atteindre 40 Gbps et plus sur un seul flux.

- **--client <server>**: Active le mode client et définit l'adresse IP ou le nom du serveur qui exécute le serveur **iperf3**.

4. Attendez que **iperf3** termine le test. Tant le serveur que le client affichent des statistiques toutes les secondes et un résumé à la fin. Par exemple, voici un résumé affiché sur un client :

```
[ ID] Interval           Transfer     Bitrate         Retr
[  5] 0.00-60.00  sec  101 GBytes   14.4 Gbits/sec   0   sender
[  5] 0.00-60.04  sec  101 GBytes   14.4 Gbits/sec       receiver
```

Dans cet exemple, le débit moyen était de 14,4 Gbps.

5. Sur le serveur :

a. Appuyer sur **Ctrl**+**C** pour arrêter le serveur **iperf3**.

b. Fermez le port TCP 5201 dans **firewalld**:

```
# firewall-cmd --remove-port=5201/tcp
# firewall-cmd --reload
```

Ressources supplémentaires

- **iperf3(1)** page de manuel

31.5.2. Paramètres de la mémoire tampon de la socket TCP à l'échelle du système

Les tampons de sockets stockent temporairement les données que le noyau a reçues ou doit envoyer :

- Le tampon de lecture de la socket contient les paquets que le noyau a reçus mais que l'application n'a pas encore lus.
- La mémoire tampon de la socket d'écriture contient les paquets qu'une application a écrits dans la mémoire tampon, mais que le noyau n'a pas encore transmis à la pile IP et au pilote de réseau.

Si un paquet TCP est trop volumineux et dépasse la taille de la mémoire tampon ou si les paquets sont envoyés ou reçus à un rythme trop rapide, le noyau abandonne tout nouveau paquet TCP entrant jusqu'à ce que les données soient retirées de la mémoire tampon. Dans ce cas, l'augmentation des tampons de la socket peut empêcher la perte de paquets.

Les paramètres du noyau de la mémoire tampon du socket **net.ipv4.tcp_rmem** (lecture) et **net.ipv4.tcp_wmem** (écriture) contiennent trois valeurs :

```
net.ipv4.tcp_rmem = 4096  131072  6291456
net.ipv4.tcp_wmem = 4096  16384   4194304
```

Les valeurs affichées sont en octets et Red Hat Enterprise Linux les utilise de la manière suivante :

- La première valeur est la taille minimale de la mémoire tampon. Les nouvelles sockets ne peuvent pas avoir une taille inférieure.
- La deuxième valeur est la taille de la mémoire tampon par défaut. Si une application ne définit pas de taille de tampon, il s'agit de la valeur par défaut.
- La troisième valeur est la taille maximale des tampons automatiquement réglés. L'utilisation de la fonction **setsockopt()** avec l'option de socket **SO_SNDBUF** dans une application désactive cette taille maximale de tampon.

Notez que les paramètres **net.ipv4.tcp_rmem** et **net.ipv4.tcp_wmem** définissent la taille des sockets pour les protocoles IPv4 et IPv6.

31.5.3. Augmentation des tampons de socket TCP au niveau du système

Les tampons de socket TCP à l'échelle du système stockent temporairement les données que le noyau a reçues ou doit envoyer. Les paramètres des tampons de socket **net.ipv4.tcp_rmem** (lecture) et **net.ipv4.tcp_wmem** (écriture) contiennent chacun trois paramètres : Une valeur minimale, une valeur par défaut et une valeur maximale.

IMPORTANT

La définition d'une taille de tampon trop importante entraîne un gaspillage de mémoire. Chaque socket peut être défini à la taille demandée par l'application, et le noyau double cette valeur. Par exemple, si une application demande une taille de tampon de socket de 256 KiB et ouvre 1 million de sockets, le système peut utiliser jusqu'à 512 Go de RAM (512 KiB x 1 million) uniquement pour l'espace tampon de socket potentiel.

En outre, une valeur trop élevée pour la taille maximale de la mémoire tampon peut augmenter la latence.

Conditions préalables

- Vous avez rencontré un taux important de paquets TCP abandonnés.

Procédure

1. Déterminez la latence de la connexion. Par exemple, faites un ping entre le client et le serveur pour mesurer le temps moyen d'aller-retour (RTT) :

```
# ping -c 10 server.example.com
...
--- server.example.com ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9014ms
rtt min/avg/max/mdev = 117.208/117.056/119.333/0.616 ms
```

   Dans cet exemple, la latence est de 117 ms.

2. Utilisez la formule suivante pour calculer le produit de retard de la bande passante (BDP) pour le trafic que vous souhaitez régler :

```
connection speed in bytes * latency in ms = BDP in bytes
```

   Par exemple, pour calculer le BDP d'une connexion de 10 Gbps ayant une latence de 117 ms :

```
(10 * 1000 * 1000 * 1000 / 8) * 117 = 10683760 bytes
```

3. Créez le fichier **/etc/sysctl.d/10-tcp-socket-buffers.conf** et définissez la taille maximale de la mémoire tampon en lecture ou en écriture, ou les deux, en fonction de vos besoins :

```
net.ipv4.tcp_rmem = 4096 262144 21367520
net.ipv4.tcp_wmem = 4096 24576 21367520
```

   Spécifiez les valeurs en octets. Utilisez la règle empirique suivante lorsque vous essayez d'identifier les valeurs optimales pour votre environnement :

   - Taille de la mémoire tampon par défaut (deuxième valeur) : N'augmentez que légèrement cette valeur ou fixez-la à **524288** (512 KiB) au maximum. Une taille de tampon par défaut trop élevée peut entraîner l'effondrement du tampon et, par conséquent, des pics de latence.

   - Taille maximale de la mémoire tampon (troisième valeur) : Une valeur double ou triple du BDP est souvent suffisante.

4. Charger les paramètres du fichier **/etc/sysctl.d/10-tcp-socket-buffers.conf**:

```
# sysctl -p /etc/sysctl.d/10-tcp-socket-buffers.conf
```

5. Configurez vos applications pour qu'elles utilisent une taille de tampon de socket plus importante. La troisième valeur des paramètres **net.ipv4.tcp_rmem** et **net.ipv4.tcp_wmem** définit la taille maximale de la mémoire tampon que la fonction **setsockopt()** d'une application peut demander.
   Pour plus de détails, consultez la documentation du langage de programmation de votre application. Si vous n'êtes pas le développeur de l'application, contactez-le.

6. Si vous avez modifié la deuxième valeur du paramètre **net.ipv4.tcp_rmem** ou **net.ipv4.tcp_wmem**, redémarrez les applications pour utiliser les nouvelles tailles de tampon TCP.
   Si vous n'avez modifié que la troisième valeur, il n'est pas nécessaire de redémarrer l'application car l'auto-tuning applique ces paramètres de manière dynamique.

Vérification

1. Facultatif : Testez le débit TCP à l'aide de iperf3 .
2. Surveillez les statistiques de chute de paquets en utilisant la même méthode que celle utilisée lorsque vous avez rencontré les chutes de paquets.
   Si les chutes de paquets se produisent toujours, mais à un taux plus faible, augmentez encore la taille des tampons.

Ressources supplémentaires

- Quelles sont les conséquences de la modification de la taille des tampons des sockets ? solution
- **tcp(7)** page de manuel
- **socket(7)** page de manuel

31.5.4. Mise à l'échelle des fenêtres TCP

La fonction TCP Window Scaling, qui est activée par défaut dans Red Hat Enterprise Linux, est une extension du protocole TCP qui améliore considérablement le débit.

Par exemple, sur une connexion de 1 Gbps avec un temps d'aller-retour (RTT) de 1,5 ms :

- Lorsque l'option TCP Window Scaling est activée, un débit d'environ 630 Mbps est réaliste.
- Lorsque la fonction TCP Window Scaling est désactivée, le débit descend à 380 Mbps.

L'une des caractéristiques du protocole TCP est le contrôle de flux. Avec le contrôle de flux, un expéditeur peut envoyer autant de données que le récepteur peut en recevoir, mais pas plus. Pour ce faire, le récepteur annonce une valeur **window**, qui est la quantité de données qu'un expéditeur peut envoyer.

À l'origine, le protocole TCP prenait en charge des fenêtres d'une taille maximale de 64 KiB, mais lorsque les produits de délai de bande passante (BDP) sont élevés, cette valeur devient une restriction car l'expéditeur ne peut pas envoyer plus de 64 KiB à la fois. Les connexions à haut débit peuvent transférer beaucoup plus de 64 KiB de données à la fois. Par exemple, une liaison de 10 Gbps avec une latence de 1 ms entre les systèmes peut avoir plus de 1 Mio de données en transit à un moment donné. Il serait inefficace qu'un hôte n'envoie que 64 KiB, puis fasse une pause jusqu'à ce que l'autre hôte reçoive ces 64 KiB.

Pour éliminer ce goulot d'étranglement, l'extension TCP Window Scaling permet de décaler arithmétiquement vers la gauche la valeur de la fenêtre TCP afin d'augmenter la taille de la fenêtre au-delà de 64 KiB. Par exemple, la plus grande valeur de fenêtre de **65535** est décalée de 7 places vers la gauche, ce qui permet d'obtenir une taille de fenêtre de près de 8 Mio. Cela permet de transférer beaucoup plus de données à un moment donné.

La mise à l'échelle des fenêtres TCP est négociée au cours de la poignée de main TCP à trois voies qui ouvre chaque connexion TCP. L'expéditeur et le destinataire doivent tous deux prendre en charge la mise à l'échelle de la fenêtre TCP pour que la fonction fonctionne. Si l'un ou l'autre des participants, ou les deux, n'annoncent pas la possibilité d'échelonner la fenêtre dans leur poignée de main, la connexion revient à la taille de fenêtre TCP originale de 16 bits.

Par défaut, TCP Window Scaling est activé dans Red Hat Enterprise Linux :

```
# sysctl net.ipv4.tcp_window_scaling
net.ipv4.tcp_window_scaling = 1
```

Si TCP Window Scaling est désactivé (**0**) sur votre serveur, rétablissez la configuration de la même manière que vous l'avez définie.

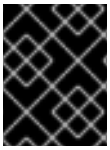
Ressources supplémentaires

- RFC 1323 : Extensions TCP pour de hautes performances
- Configuring kernel parameters at runtime

31.5.5. Comment TCP SACK réduit le taux d'abandon des paquets

La fonction TCP Selective Acknowledgment (TCP SACK), qui est activée par défaut dans Red Hat Enterprise Linux (RHEL), est une amélioration du protocole TCP et augmente l'efficacité des connexions TCP.

Dans les transmissions TCP, le récepteur envoie un paquet ACK à l'expéditeur pour chaque paquet qu'il reçoit. Par exemple, un client envoie les paquets TCP 1 à 10 au serveur, mais les paquets 5 et 6 sont perdus. Sans TCP SACK, le serveur laisse tomber les paquets 7 à 10 et le client doit retransmettre tous les paquets à partir du point de perte, ce qui est inefficace. Lorsque la fonction TCP SACK est activée sur les deux hôtes, le client ne doit retransmettre que les paquets 5 et 6 perdus.



IMPORTANT

La désactivation de TCP SACK réduit les performances et entraîne un taux d'abandon de paquets plus élevé du côté du destinataire dans une connexion TCP.

Par défaut, TCP SACK est activé dans RHEL. Pour vérifier :

```
# sysctl net.ipv4.tcp_sack
1
```

Si TCP SACK est désactivé (**0**) sur votre serveur, rétablissez la configuration de la même manière que vous l'avez définie.

Ressources supplémentaires

- RFC 2018 : Options d'accusé de réception sélectif de TCP
- Dois-je m'inquiéter d'un taux de perte de paquets de 0,05 % ? solution
- Configuring kernel parameters at runtime

31.6. OPTIMISATION DES CONNEXIONS UDP

Avant de commencer à régler Red Hat Enterprise Linux pour améliorer le débit du trafic UDP, il est important d'avoir des attentes réalistes. UDP est un protocole simple. Comparé à TCP, UDP ne contient pas de fonctions telles que le contrôle de flux, le contrôle de congestion et la fiabilité des données. Il est donc difficile d'obtenir une communication fiable via UDP avec un débit proche de la vitesse maximale du contrôleur d'interface réseau (NIC).

31.6.1. Détection des chutes de paquets

Il existe plusieurs niveaux dans la pile du réseau dans lesquels le noyau peut laisser tomber des paquets. Red Hat Enterprise Linux fournit différents utilitaires pour afficher les statistiques de ces niveaux. Utilisez-les pour identifier les problèmes potentiels.

Notez que vous pouvez ignorer un très faible taux de paquets abandonnés. Cependant, si vous rencontrez un taux significatif, envisagez des mesures de réglage.

NOTE

Le noyau laisse tomber les paquets réseau si la pile réseau ne peut pas gérer le trafic entrant.

Procédure

1. Identifier si le contrôleur d'interface réseau (NIC) laisse tomber des paquets :

   a. Affiche les statistiques spécifiques au NIC et au pilote :

   ```
   # ethtool -S enp1s0
   NIC statistics:
       ...
       rx_queue_0_drops: 17657
       ...
   ```

   La dénomination des statistiques et leur disponibilité dépendent de la carte réseau et du pilote.

   b. Affiche les statistiques de l'interface :

   ```
   # ip -s link show enp1s0
   2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state
   UP mode DEFAULT group default qlen 1000
       link/ether 52:54:00:74:79:56 brd ff:ff:ff:ff:ff:ff
       RX:   bytes  packets errors dropped  missed   mcast_
       84697611107 56866482      0   10904       0       0
       TX:   bytes  packets errors dropped carrier collsns_
        5540028184  3722234       0       0       0       0
   ```

   RX représente les statistiques des paquets reçus et **TX** des paquets transmis.

2. Identifier les chutes de paquets spécifiques au protocole UDP dues à des tampons de socket trop petits ou à la lenteur du traitement de l'application :

   ```
   # nstat -az UdpSndbufErrors UdpRcvbufErrors
   #kernel
   UdpSndbufErrors          4    0.0
   UdpRcvbufErrors    45716659    0.0
   ```

   La deuxième colonne de la sortie énumère les compteurs.

Ressources supplémentaires

- Solution pour l'interface réseau RHEL qui laisse tomber des paquets

- Dois-je m'inquiéter d'un taux de perte de paquets de 0,05 % ? solution

31.6.2. Test du débit UDP avec iperf3

L'utilitaire **iperf3** propose un mode serveur et un mode client pour effectuer des tests de débit réseau entre deux hôtes.

NOTE

Le débit des applications dépend de nombreux facteurs, tels que la taille des tampons utilisés par l'application. Par conséquent, les résultats mesurés à l'aide d'utilitaires de test, tels que **iperf3**, peuvent être sensiblement différents de ceux des applications sur un serveur soumis à une charge de travail de production.

Conditions préalables

- Le paquet **iperf3** est installé à la fois sur le client et sur le serveur.
- Aucun autre service sur les deux hôtes ne provoque un trafic réseau qui affecte substantiellement le résultat du test.
- Facultatif : vous avez augmenté la taille maximale des sockets UDP sur le serveur et le client. Pour plus d'informations, reportez-vous à la section Augmentation des tampons de socket UDP à l'échelle du système.

Procédure

1. Facultatif : Affichez la vitesse réseau maximale du contrôleur d'interface réseau (NIC) sur le serveur et le client :

```
# ethtool enp1s0 | grep "Speed"
  Speed: 10000Mb/s
```

2. Sur le serveur :

   a. Affichez la taille maximale du tampon de lecture de la socket UDP et notez la valeur :

```
# sysctl net.core.rmem_max
net.core.rmem_max = 16777216
```

   La valeur affichée est en octets.

   b. Ouvrez temporairement le port 5201 par défaut de **iperf3** dans le service **firewalld**:

```
# firewall-cmd --add-port=5201/tcp --add-port=5201/udp
# firewall-cmd --reload
```

   Notez que **iperf3** n'ouvre qu'un socket TCP sur le serveur. Si un client souhaite utiliser le protocole UDP, il se connecte d'abord à ce port TCP, puis le serveur ouvre un socket UDP sur le même numéro de port pour effectuer le test de débit du trafic UDP. Pour cette raison, vous devez ouvrir le port 5201 pour les protocoles TCP et UDP dans le pare-feu local.

   c. Démarrer **iperf3** en mode serveur :

```
# iperf3 --server
```

Le service attend maintenant les connexions entrantes des clients.

3. Sur le client :

   a. Affichez l'unité de transmission maximale (MTU) de l'interface que le client utilisera pour la connexion au serveur et notez la valeur :

   ```
   # ip link show enp1s0
   2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state
   UP mode DEFAULT group default qlen 1000
   ...
   ```

   b. Affichez la taille maximale du tampon d'écriture de la socket UDP et notez la valeur :

   ```
   # sysctl net.core.wmem_max
   net.core.wmem_max = 16777216
   ```

   La valeur affichée est en octets.

   c. Commencez à mesurer le débit :

   ```
   # iperf3 --udp --time 60 --window 16777216 --length 1472 --bitrate 2G --client
   192.0.2.1
   ```

   - **--udp**: Utiliser le protocole UDP pour le test.
   - **--time *<seconds>***: Définit le temps en secondes pendant lequel le client arrête la transmission.
   - **--window *<size>***: Définit la taille de la mémoire tampon de la socket UDP. Idéalement, les tailles sont les mêmes sur le client et le serveur. Si elles sont différentes, réglez ce paramètre sur la valeur la plus petite : **net.core.wmem_max** sur le client ou **net.core.rmem_max** sur le serveur.
   - **--length *<size>***: Définit la longueur de la mémoire tampon à lire et à écrire. Réglez cette option sur la plus grande charge utile non fragmentée. Calculez la valeur idéale comme suit : MTU - En-tête IP (20 octets pour IPv4 et 40 octets pour IPv6) - En-tête UDP de 8 octets.
   - **--bitrate *<rate>***: Limite le débit à la valeur spécifiée en bits par seconde. Vous pouvez spécifier des unités, telles que **2G** pour 2 Gbps.
     Réglez ce paramètre à une valeur qui devrait fonctionner et augmentez-la lors des mesures ultérieures. Si le serveur envoie des paquets à une vitesse supérieure à celle à laquelle les périphériques sur le chemin de transmission ou le client peuvent les traiter, les paquets peuvent être abandonnés.
   - **--client <server>**: Active le mode client et définit l'adresse IP ou le nom du serveur qui exécute le serveur **iperf3**.

4. Attendez que **iperf3** termine le test. Tant le serveur que le client affichent des statistiques toutes les secondes et un résumé à la fin. Par exemple, voici un résumé affiché sur un client :

```
[ ID] Interval       Transfer     Bitrate         Jitter    Lost/Total Datagrams
[ 5] 0.00-60.00 sec  14.0 GBytes   2.00 Gbits/sec  0.000 ms  0/10190216 (0%) sender
[ 5] 0.00-60.04 sec  14.0 GBytes   2.00 Gbits/sec  0.002 ms  0/10190216 (0%) receiver
```

Dans cet exemple, le débit moyen était de 2 Gbps et aucun paquet n'a été perdu.

5. Sur le serveur :

   a. Appuyer sur **Ctrl**+**C** pour arrêter le serveur **iperf3**.

   b. Fermer le port 5201 dans **firewalld**:

```
# firewall-cmd --remove-port=5201/tcp --remove-port=5201/udp
# firewall-cmd --reload
```

Ressources supplémentaires

- **iperf3(1)** page de manuel

31.6.3. Impact de la taille du MTU sur le débit du trafic UDP

Si votre application utilise des messages UDP de grande taille, l'utilisation de trames jumbo peut améliorer le débit. Selon la norme IEEE 802.3, une trame Ethernet par défaut sans balise VLAN (Virtual Local Area Network) a une taille maximale de 1518 octets. Chacune de ces trames comprend un en-tête de 18 octets, ce qui laisse 1500 octets pour la charge utile. Par conséquent, pour chaque 1500 octets de données que le serveur transmet sur le réseau, 18 octets (1,2 %) sont des frais généraux.

Les trames jumbo sont des trames non standardisées dont l'unité de transmission maximale (MTU) est supérieure à la taille de la charge utile Ethernet standard de 1500 octets. Par exemple, si vous configurez des trames jumbo avec le MTU maximum autorisé de 9000 octets de charge utile, l'overhead de chaque trame est réduit à 0,2 %.

IMPORTANT

Tous les dispositifs du réseau sur le chemin de transmission et les domaines de diffusion concernés doivent prendre en charge les trames jumbo et utiliser le même MTU. La fragmentation et le réassemblage des paquets dus à des paramètres MTU incohérents sur le chemin de transmission réduisent le débit du réseau.

Les différents types de connexion ont certaines limites MTU :

- Ethernet : le MTU est limité à 9000 octets.
- IP over InfiniBand (IPoIB) en mode datagramme : Le MTU est limité à 4 octets de moins que le MTU InfiniBand.
- Les réseaux en mémoire prennent généralement en charge des MTU plus importants. Pour plus de détails, voir la documentation correspondante.

31.6.4. Impact de la vitesse de l'unité centrale sur le débit du trafic UDP

Dans les transferts de masse, le protocole UDP est beaucoup moins efficace que le protocole TCP, principalement en raison de l'absence d'agrégation de paquets dans le protocole UDP. Par défaut, les fonctions Generic Receive Offload (GRO) et Transmit Segmentation Offload (TSO) ne sont pas

activées. Par conséquent, la fréquence de l'unité centrale peut limiter le débit UDP pour les transferts de masse sur les liaisons à haut débit.

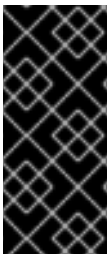
Par exemple, sur un hôte réglé avec une unité de transmission maximale (MTU) élevée et des tampons de socket importants, une unité centrale de 3 GHz peut traiter le trafic d'une carte d'interface réseau de 10 GBit qui envoie ou reçoit du trafic UDP à pleine vitesse. Cependant, vous pouvez vous attendre à une perte de vitesse de 1 à 2 Gbps pour chaque 100 MHz de vitesse de CPU en dessous de 3 GHz lorsque vous transmettez du trafic UDP. De plus, si une vitesse de CPU de 3 GHz peut atteindre 10 Gbps, le même CPU limite le trafic UDP sur une carte réseau de 40 GBit à environ 20-25 Gbps.

31.6.5. Augmentation des tampons des sockets UDP dans l'ensemble du système

Les tampons de sockets stockent temporairement les données que le noyau a reçues ou doit envoyer :

- Le tampon de lecture de la socket contient les paquets que le noyau a reçus mais que l'application n'a pas encore lus.
- La mémoire tampon de la socket d'écriture contient les paquets qu'une application a écrits dans la mémoire tampon, mais que le noyau n'a pas encore transmis à la pile IP et au pilote de réseau.

Si un paquet UDP est trop volumineux et dépasse la taille de la mémoire tampon ou si les paquets sont envoyés ou reçus à un rythme trop rapide, le noyau laisse tomber tout nouveau paquet UDP entrant jusqu'à ce que les données soient retirées de la mémoire tampon. Dans ce cas, l'augmentation des tampons de la socket peut empêcher la perte de paquets.



IMPORTANT

La définition d'une taille de tampon trop importante entraîne un gaspillage de mémoire. Chaque socket peut être réglé sur la taille demandée par l'application, et le noyau double cette valeur. Par exemple, si une application demande une taille de tampon de socket de 256 KiB et ouvre 1 million de sockets, le système a besoin de 512 Go de RAM (512 KiB x 1 million) uniquement pour l'espace tampon de socket potentiel.

Conditions préalables

- Vous avez rencontré un taux important de paquets UDP abandonnés.

Procédure

1. Créez le fichier **/etc/sysctl.d/10-udp-socket-buffers.conf** et définissez la taille maximale de la mémoire tampon en lecture ou en écriture, ou les deux, en fonction de vos besoins :

   ```
   net.core.rmem_max = 16777216
   net.core.wmem_max = 16777216
   ```

   Spécifiez les valeurs en octets. Les valeurs indiquées dans cet exemple fixent la taille maximale des tampons à 16 Mio. Les valeurs par défaut des deux paramètres sont **212992** bytes (208 KiB).

2. Charger les paramètres du fichier **/etc/sysctl.d/10-udp-socket-buffers.conf**:

   ```
   # sysctl -p /etc/sysctl.d/10-udp-socket-buffers.conf
   ```

3. Configurez vos applications pour qu'elles utilisent des tampons de socket plus grands.
   Les paramètres **net.core.rmem_max** et **net.core.wmem_max** définissent la taille maximale de

la mémoire tampon que la fonction **setsockopt()** d'une application peut demander. Notez que si vous configurez votre application pour ne pas utiliser la fonction **setsockopt()**, le noyau utilise les valeurs des paramètres **rmem_default** et **wmem_default**.

Pour plus de détails, consultez la documentation du langage de programmation de votre application. Si vous n'êtes pas le développeur de l'application, contactez-le.

4. Redémarrez les applications pour utiliser les nouvelles tailles de tampon UDP.

Vérification

- Surveillez les statistiques de chute de paquets en utilisant la même méthode que celle utilisée lorsque vous avez rencontré les chutes de paquets.
  Si les chutes de paquets se produisent toujours, mais à un taux plus faible, augmentez encore la taille des tampons.

Ressources supplémentaires

- Quelles sont les conséquences de la modification de la taille des tampons des sockets ? solution
- **udp(7)** page de manuel
- **socket(7)** page de manuel

31.7. IDENTIFIER LES GOULOTS D'ÉTRANGLEMENT DE LA MÉMOIRE TAMPON DE LA SOCKET DE LECTURE DE L'APPLICATION

Si les applications TCP n'effacent pas les tampons de lecture de la socket assez fréquemment, la performance peut en souffrir et des paquets peuvent être perdus. Red Hat Enterprise Linux fournit différents utilitaires pour identifier de tels problèmes.

31.7.1. Identification de l'effondrement et de l'élagage de la mémoire tampon de réception

Lorsque les données de la file d'attente de réception dépassent la taille du tampon de réception, la pile TCP tente de libérer de l'espace en supprimant les métadonnées inutiles du tampon de la socket. Cette étape est connue sous le nom de "collapsing".

Si l'effondrement ne parvient pas à libérer suffisamment d'espace pour le trafic supplémentaire, le noyau élague les nouvelles données qui arrivent. Cela signifie que le noyau supprime les données de la mémoire et que le paquet est perdu.

Pour éviter les opérations de réduction et d'élagage, surveillez si la réduction et l'élagage des tampons TCP se produisent sur votre serveur et, dans ce cas, réglez les tampons TCP.

Procédure

1. Utilisez l'utilitaire **nstat** pour interroger les compteurs **TcpExtTCPRcvCollapsed** et **TcpExtRcvPruned**:

```
# nstat -az TcpExtTCPRcvCollapsed TcpExtRcvPruned
#kernel
TcpExtRcvPruned            0        0.0
TcpExtTCPRcvCollapsed      612859   0.0
```

2. Attendez un peu et exécutez à nouveau la commande **nstat**:

```
# nstat -az TcpExtTCPRcvCollapsed TcpExtRcvPruned
#kernel
TcpExtRcvPruned            0         0.0
TcpExtTCPRcvCollapsed      620358    0.0
```

3. Si les valeurs des compteurs ont augmenté par rapport à la première exécution, un réglage est nécessaire :

   - Si l'application utilise l'appel **setsockopt(SO_RCVBUF)**, envisagez de le supprimer. Avec cet appel, l'application n'utilise que la taille du tampon de réception spécifiée dans l'appel et désactive la capacité de la socket à ajuster automatiquement sa taille.

   - Si l'application n'utilise pas l'appel **setsockopt(SO_RCVBUF)**, réglez les valeurs par défaut et maximale de la mémoire tampon de la socket TCP read.

4. Affichez la file d'attente des commandes en attente de réception (**Recv-Q**) :

```
# ss -nti
State   Recv-Q   Send-Q   Local Address:Port   Peer Address:Port   Process
ESTAB   0        0        192.0.2.1:443        192.0.2.125:41574
     :7,7 ... lastrcv:543 ...
ESTAB   78       0        192.0.2.1:443        192.0.2.56:42612
     :7,7 ... lastrcv:658 ...
ESTAB   88       0        192.0.2.1:443        192.0.2.97:40313
     :7,7 ... lastrcv:5764 ...
...
```

5. Exécutez la commande **ss -nt** plusieurs fois en attendant quelques secondes entre chaque exécution.
   Si la sortie ne mentionne qu'un seul cas de valeur élevée dans la colonne **Recv-Q**, l'application se trouvait entre deux opérations de réception. Toutefois, si les valeurs de **Recv-Q** restent constantes alors que **lastrcv** augmente continuellement ou que **Recv-Q** augmente continuellement au fil du temps, l'un des problèmes suivants peut en être la cause :

   - L'application ne vérifie pas ses tampons de socket assez souvent. Contactez le fournisseur de l'application pour savoir comment résoudre ce problème.

   - L'application n'a pas assez de temps d'utilisation du processeur. Pour déboguer davantage ce problème :

     i. Affichage des cœurs de l'unité centrale sur lesquels l'application s'exécute :

```
# ps -eo pid,tid,psr,pcpu,stat,wchan:20,comm
   PID     TID PSR %CPU STAT WCHAN               COMMAND
...
 44594   44594   5  0.0 Ss   do_select           httpd
 44595   44595   3  0.0 S    skb_wait_for_more_pa httpd
 44596   44596   5  0.0 Sl   pipe_read           httpd
 44597   44597   5  0.0 Sl   pipe_read           httpd
 44602   44602   5  0.0 Sl   pipe_read           httpd
...
```

     La colonne **PSR** affiche les cœurs de l'unité centrale auxquels le processus est actuellement affecté.

ii. Identifier les autres processus s'exécutant sur les mêmes cœurs et envisager de les affecter à d'autres cœurs.

Ressources supplémentaires

- Augmentation des tampons de socket TCP au niveau du système

31.8. OPTIMISATION DES APPLICATIONS AVEC UN GRAND NOMBRE DE REQUÊTES ENTRANTES

Si vous exécutez une application qui traite un grand nombre de requêtes entrantes, comme les serveurs web, il peut être nécessaire de régler Red Hat Enterprise Linux afin d'optimiser les performances.

31.8.1. Ajustement du backlog d'écoute TCP pour traiter un nombre élevé de tentatives de connexion TCP

Lorsqu'une application ouvre un socket TCP dans l'état **LISTEN**, le noyau limite le nombre de connexions client acceptées que ce socket peut gérer. Si les clients essaient d'établir plus de connexions que l'application ne peut en traiter, les nouvelles connexions sont perdues ou le noyau envoie des cookies SYN au client.

Si le système est soumis à une charge de travail normale et qu'un trop grand nombre de connexions de clients légitimes amène le noyau à envoyer des cookies SYN, réglez Red Hat Enterprise Linux (RHEL) pour les éviter.

Conditions préalables

- RHEL enregistre **possible SYN flooding on port *<ip_address>*:*<port_number>*** les messages d'erreur dans le journal Systemd.
- Le nombre élevé de tentatives de connexion provient de sources valables et n'est pas dû à une attaque.

Procédure

1. Pour vérifier si un réglage est nécessaire, affichez les statistiques du port concerné :

```
# ss -ntl '( sport = :443 )'
State    Recv-Q   Send-Q   Local Address:Port   Peer Address:Port  Process
LISTEN   650      500      192.0.2.1:443        0.0.0.0:*
```

   Si le nombre actuel de connexions dans l'arriéré (**Recv-Q**) est supérieur à l'arriéré de sockets (**Send-Q**), l'arriéré d'écoute n'est pas encore assez important et un réglage est nécessaire.

2. Facultatif : Affiche la limite actuelle de l'arriéré d'écoute TCP :

```
# sysctl net.core.somaxconn
net.core.somaxconn = 4096
```

3. Créez le fichier **/etc/sysctl.d/10-socket-backlog-limit.conf** et fixez une limite d'écoute plus élevée :

```
net.core.somaxconn = 8192
```

Il est à noter que les applications peuvent demander une réserve d'écoute plus importante que celle spécifiée dans le paramètre du noyau **net.core.somaxconn**, mais le noyau limite l'application au nombre que vous avez défini dans ce paramètre.

4. Charger le réglage à partir du fichier **/etc/sysctl.d/10-socket-backlog-limit.conf**:

```
# sysctl -p /etc/sysctl.d/10-socket-backlog-limit.conf
```

5. Reconfigurer l'application pour utiliser la nouvelle limite d'écoute :
   - Si l'application fournit une option de configuration pour la limite, mettez-la à jour. Par exemple, le serveur HTTP Apache fournit l'option de configuration **ListenBacklog** pour définir la limite de l'arriéré d'écoute pour ce service.
   - Si vous ne pouvez pas configurer la limite, recompilez l'application.
6. Restart the application.

Vérification

1. Surveillez le journal Systemd pour détecter d'autres messages d'erreur **possible SYN flooding on port *<port_number>*** messages d'erreur.
2. Surveillez le nombre actuel de connexions dans l'arriéré et comparez-le à l'arriéré de prises :

```
# ss -ntl '( sport = :443 )'
State    Recv-Q   Send-Q   Local Address:Port   Peer Address:Port  Process
LISTEN   0        500      192.0.2.1:443        0.0.0.0:*
```

   Si le nombre actuel de connexions dans l'arriéré (**Recv-Q**) est supérieur à l'arriéré de sockets (**Send-Q**), l'arriéré d'écoute n'est pas assez important et un réglage supplémentaire est nécessaire.

Ressources supplémentaires

- kernel : Inondation SYN possible sur le port #. Envoi d'une solution de cookies
- Le serveur TCP à l'écoute ignore la solution SYN ou ACK pour une nouvelle connexion
- **listen(2)** page de manuel

31.9. ÉVITER LES CONFLITS DE VERROUILLAGE DE LA FILE D'ATTENTE D'ÉCOUTE

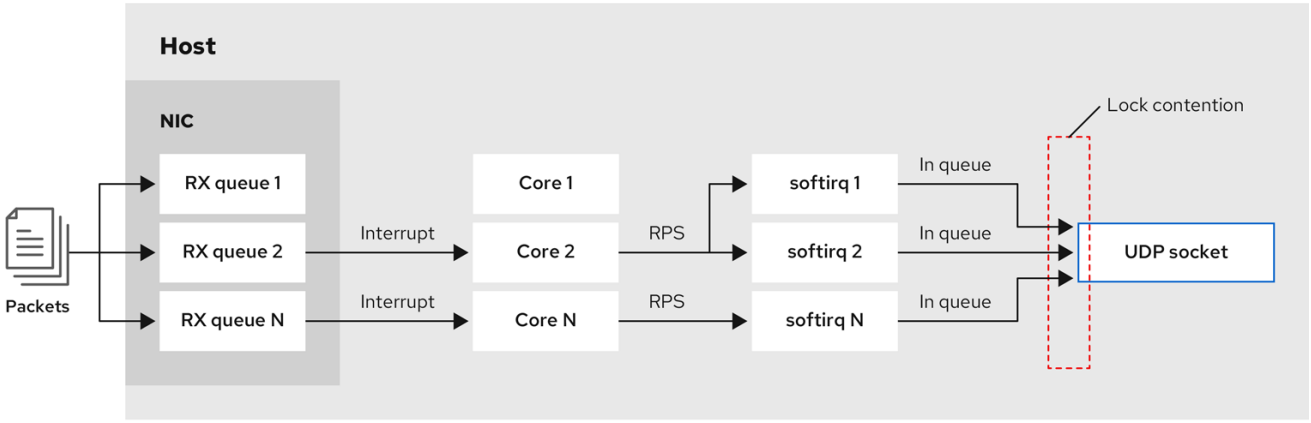
La contention du verrouillage de la file d'attente peut entraîner des chutes de paquets et une utilisation accrue de l'unité centrale et, par conséquent, une latence plus élevée. Vous pouvez éviter la contention de la file d'attente de réception (RX) et de transmission (TX) en réglant votre application et en utilisant le pilotage des paquets de transmission.

31.9.1. Éviter la contention du verrouillage de la file d'attente RX : Les options de socket SO_REUSEPORT et SO_REUSEPORT_BPF

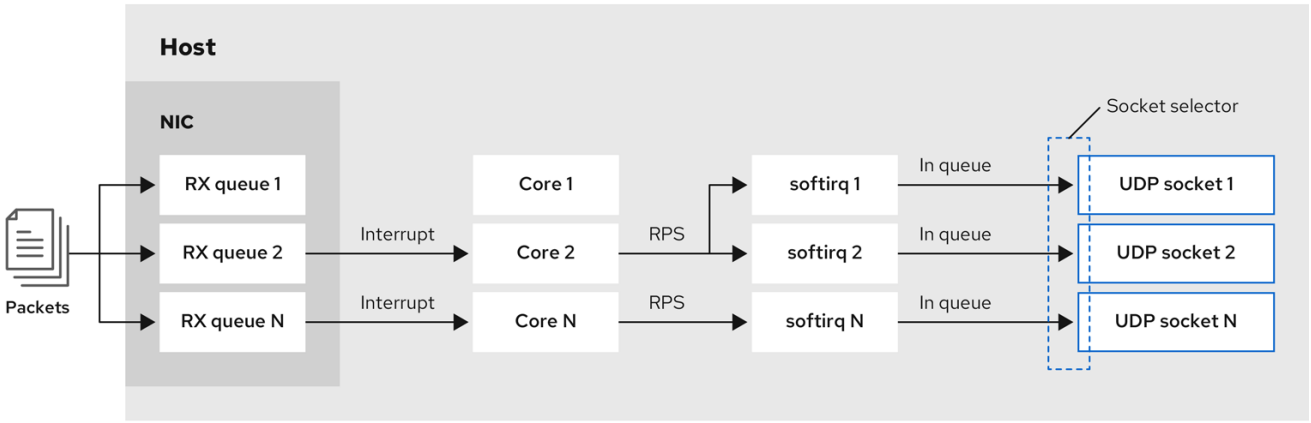
Sur un système multicœur, vous pouvez améliorer les performances des applications de serveur réseau multithread si l'application ouvre le port en utilisant l'option de socket **SO_REUSEPORT** ou

SO_REUSEPORT_BPF. Si l'application n'utilise pas l'une de ces options de socket, tous les threads sont obligés de partager un socket unique pour recevoir le trafic entrant. L'utilisation d'une seule socket provoque :

- Contrainte importante sur la mémoire tampon de réception, ce qui peut entraîner des pertes de paquets et une utilisation accrue de l'unité centrale.
- Augmentation significative de l'utilisation de l'unité centrale
- Possiblement des chutes de paquets



Avec l'option de socket **SO_REUSEPORT** ou **SO_REUSEPORT_BPF**, plusieurs sockets sur un hôte peuvent se lier au même port :



Red Hat Enterprise Linux fournit un exemple de code sur l'utilisation des options de socket **SO_REUSEPORT** dans les sources du noyau. Pour accéder à l'exemple de code :

1. Activer le référentiel **rhel-9-for-x86_64-baseos-debug-rpms**:

```
# subscription-manager repos --enable rhel-9-for-x86_64-baseos-debug-rpms
```

2. Installez le paquetage **kernel-debuginfo-common-x86_64**:

```
# dnf install kernel-debuginfo-common-x86_64
```

3. L'exemple de code est maintenant disponible dans le fichier **/usr/src/debug/kernel-*<version>*/linux-<version>/tools/testing/selftests/net/reuseport_bpf_cpu.c** fichier.

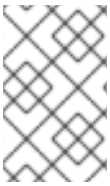
Ressources supplémentaires

- **socket(7)** page de manuel
- **/usr/src/debug/kernel-*<version>*/linux-<version>/tools/testing/selftests/net/reuseport_bpf_cpu.c**

31.9.2. Éviter la contention du verrouillage de la file d'attente TX : Pilotage des paquets d'émission

Dans les hôtes dotés d'un contrôleur d'interface réseau (NIC) qui prend en charge plusieurs files d'attente, la fonction XPS (transmit packet steering) répartit le traitement des paquets réseau sortants entre plusieurs files d'attente. Cela permet à plusieurs unités centrales de traiter le trafic réseau sortant et d'éviter les conflits de verrouillage des files d'attente de transmission et, par conséquent, les chutes de paquets.

Certains pilotes, tels que **ixgbe**, **i40e**, et **mlx5** configurent automatiquement XPS. Pour savoir si le pilote prend en charge cette fonctionnalité, consultez la documentation de votre pilote de carte réseau. Consultez la documentation de votre pilote NIC pour savoir s'il prend en charge cette fonctionnalité. Si le pilote ne prend pas en charge le réglage automatique de XPS, vous pouvez affecter manuellement des cœurs de CPU aux files d'attente de transmission.



NOTE

Red Hat Enterprise Linux n'offre pas d'option permettant d'assigner de manière permanente les files d'attente de transmission aux cœurs de CPU. Utilisez les commandes dans un script et exécutez-le lorsque le système démarre.

Conditions préalables

- Le NIC prend en charge plusieurs files d'attente.
- Le paquet **numactl** est installé.

Procédure

1. Affiche le nombre de files d'attente disponibles :

```
# ethtool -l enp1s0
Channel parameters for enp1s0:
Pre-set maximums:
RX:  0
TX:  0
Other:  0
Combined: 4
Current hardware settings:
RX:  0
TX:  0
Other:  0
Combined: 1
```

La section **Pre-set maximums** indique le nombre total de files d'attente et **Current hardware settings** le nombre de files d'attente actuellement affectées aux files d'attente de réception, de transmission, autres ou combinées.

2. Facultatif : Si vous avez besoin de files d'attente sur des canaux spécifiques, attribuez-les en conséquence. Par exemple, pour affecter les 4 files d'attente au canal **Combined**, entrez :

```
# ethtool -L enp1s0 combined 4
```

3. Affiche le nœud NUMA (Non-Uniform Memory Access) auquel la carte d'interface réseau est affectée :

```
# cat /sys/class/net/enp1s0/device/numa_node
0
```

Si le fichier est introuvable ou si la commande renvoie **-1**, l'hôte n'est pas un système NUMA.

4. Si l'hôte est un système NUMA, afficher quels CPU sont assignés à quel nœud NUMA :

```
# lscpu | grep NUMA
NUMA node(s):      2
NUMA node0 CPU(s):  0-3
NUMA node1 CPU(s):  4-7
```

5. Dans l'exemple ci-dessus, la carte d'interface réseau a 4 files d'attente et la carte d'interface réseau est assignée au nœud NUMA 0. Ce nœud utilise les cœurs de CPU 0-3. Par conséquent, mappez chaque file d'attente de transmission à l'un des cœurs de CPU de 0 à 3 :

```
# echo 1 > /sys/class/net/enp1s0/queues/tx-0/xps_cpus
# echo 2 > /sys/class/net/enp1s0/queues/tx-1/xps_cpus
# echo 4 > /sys/class/net/enp1s0/queues/tx-2/xps_cpus
# echo 8 > /sys/class/net/enp1s0/queues/tx-3/xps_cpus
```

Si le nombre de cœurs de CPU et de files d'attente de transmission (TX) est le même, utilisez un mappage 1 à 1 pour éviter toute forme de contention sur la file d'attente TX. Dans le cas contraire, si vous affectez plusieurs CPU à la même file d'attente TX, les opérations de transmission sur différents CPU entraîneront une contention du verrouillage de la file d'attente TX, ce qui aura un impact négatif sur le débit de transmission.

Notez que vous devez transmettre le bitmap, qui contient les numéros de cœur du processeur, aux files d'attente. Utilisez la commande suivante pour calculer le bitmap :

```
# printf %x $((1 << <core_number> ))
```

Vérification

1. Identifier les identifiants de processus (PID) des services qui envoient du trafic :

```
# pidof <process_name>
12345 98765
```

2. Attachez les PID aux cœurs qui utilisent XPS :

```
# numactl -C 0-3 12345 98765
```

3. Surveillez le compteur **requeues** pendant que le processus envoie du trafic :

```
# tc -s qdisc
qdisc fq_codel 0: dev enp10s0u1 root refcnt 2 limit 10240p flows 1024 quantum 1514 target
5ms interval 100ms memory_limit 32Mb ecn drop_batch 64
 Sent 125728849 bytes 1067587 pkt (dropped 0, overlimits 0 requeues 30)
 backlog 0b 0p requeues 30
 ...
```

Si le compteur **requeues** n'augmente plus à un rythme significatif, il n'y a plus de conflit de verrouillage de la file d'attente TX.

Ressources supplémentaires

- **/usr/share/doc/kernel-doc-_<version>/Documentation/networking/scaling.rst**

31.9.3. Désactivation de la fonction Generic Receive Offload sur les serveurs ayant un trafic UDP élevé

Les applications qui utilisent le transfert de masse UDP à grande vitesse doivent activer et utiliser la fonction UDP Generic Receive Offload (GRO) sur la socket UDP. Cependant, vous pouvez désactiver GRO pour augmenter le débit si les conditions suivantes s'appliquent :

- L'application ne prend pas en charge le GRO et la fonctionnalité ne peut pas être ajoutée.
- Le débit TCP n'est pas pertinent.

AVERTISSEMENT

La désactivation de GRO réduit considérablement le débit de réception du trafic TCP. Par conséquent, ne désactivez pas le GRO sur les hôtes où les performances TCP sont importantes.

Conditions préalables

- L'hôte traite principalement le trafic UDP.
- L'application n'utilise pas de GRO.
- L'hôte n'utilise pas de protocoles de tunnel UDP, tels que VXLAN.
- L'hôte n'exécute pas de machines virtuelles (VM) ni de conteneurs.

Procédure

1. Optionnel : Afficher les profils de connexion de NetworkManager :

```
# nmcli connection show
NAME     UUID                                  TYPE      DEVICE
example  f2f33f29-bb5c-3a07-9069-be72eaec3ecf  ethernet  enp1s0
```

2. Désactiver la prise en charge du GRO dans le profil de connexion :

```
# nmcli connection modify example ethtool.feature-gro off
```

3. Réactiver le profil de connexion :

```
# nmcli connection up example
```

Vérification

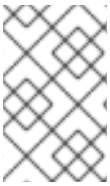
1. Vérifiez que le GRO est désactivé :

```
# ethtool -k enp1s0 | grep generic-receive-offload
generic-receive-offload: off
```

2. Surveillez le débit du serveur. Réactiver le GRO dans le profil NetworkManager si le paramètre a des effets secondaires négatifs sur d'autres applications sur l'hôte.

31.10. OPTIMISATION DU PILOTE DE PÉRIPHÉRIQUE ET DE LA CARTE D'INTERFACE RÉSEAU

Dans RHEL, les modules du noyau fournissent des pilotes pour les contrôleurs d'interface réseau (NIC). Ces modules prennent en charge des paramètres permettant de régler et d'optimiser le pilote de périphérique et la carte d'interface réseau. Par exemple, si le pilote prend en charge le retardement de la génération d'interruptions de réception, vous pouvez réduire la valeur du paramètre correspondant pour éviter de manquer de descripteurs de réception.

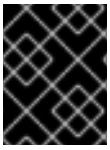


NOTE

Tous les modules ne prennent pas en charge les paramètres personnalisés, et les fonctionnalités dépendent du matériel, ainsi que de la version du pilote et du micrologiciel.

31.10.1. Configuration des paramètres personnalisés du pilote NIC

De nombreux modules du noyau permettent de définir des paramètres pour régler le pilote et le contrôleur d'interface réseau (NIC). Vous pouvez personnaliser les paramètres en fonction du matériel et du pilote.



IMPORTANT

Si vous définissez des paramètres sur un module du noyau, RHEL applique ces paramètres à tous les périphériques qui utilisent ce pilote.

Conditions préalables

- Un NIC est installé dans l'hôte.

- Le module du noyau qui fournit le pilote pour la carte d'interface réseau prend en charge la fonction de réglage requise.
- Vous êtes connecté localement ou à l'aide d'une interface réseau différente de celle qui utilise le pilote pour lequel vous souhaitez modifier les paramètres.

Procédure

1. Identifier le conducteur :

```
# ethtool -i enp0s31f6
driver: e1000e
version: ...
firmware-version: ...
...
```

Notez que certaines fonctions peuvent nécessiter une version spécifique du pilote et du micrologiciel.

2. Affiche les paramètres disponibles du module du noyau :

```
# modinfo -p e1000e
...
SmartPowerDownEnable:Enable PHY smart power down (array of int)
parm:RxIntDelay:Receive Interrupt Delay (array of int)
```

Pour plus de détails sur les paramètres, voir la documentation du module du noyau. Pour les modules de RHEL, voir la documentation du répertoire **/usr/share/doc/kernel-doc-*<version>*/Documentation/networking/device_drivers/** fournie par le paquetage **kernel-doc**.

3. Créez le fichier **/etc/modprobe.d/*nic-parameters.conf*** et spécifier les paramètres du module :

```
options <module_name> <parameter1> =<value> <parameter2> =<value>
```

Par exemple, pour activer le mécanisme d'économie d'énergie du port et définir la génération d'interruptions de réception à 4 unités, entrez :

```
options e1000e SmartPowerDownEnable=1 RxIntDelay=4
```

4. Décharger le module :

```
# modprobe -r e1000e
```

AVERTISSEMENT

Le déchargement d'un module utilisé par une interface réseau active met immédiatement fin à la connexion et vous pouvez vous bloquer sur le serveur.

5. Charger le module :

```
# modprobe e1000e
```

6. Réactiver les connexions réseau :

```
# nmcli connection up <profile_name>
```

Vérification

1. Affiche les messages du noyau :

```
# dmesg
...
[35309.225765] e1000e 0000:00:1f.6: Transmit Interrupt Delay set to 16
[35309.225769] e1000e 0000:00:1f.6: PHY Smart Power Down Enabled
...
```

   Notez que tous les modules n'enregistrent pas les paramètres dans le tampon circulaire du noyau.

2. Certains modules du noyau créent des fichiers pour chaque paramètre du module dans le répertoire **/sys/module/*<driver>*/parameters/** pour chaque paramètre du module. Chacun de ces fichiers contient la valeur actuelle de ce paramètre. Vous pouvez afficher ces fichiers pour vérifier un paramètre :

```
# cat /sys/module/<driver_name>/parameters/<parameter_name>
```

31.11. CONFIGURATION DES PARAMÈTRES DE DÉLESTAGE DE LA CARTE RÉSEAU

Pour réduire la charge du processeur, certains adaptateurs réseau utilisent des fonctions de délestage qui déplacent la charge de traitement du réseau vers le contrôleur d'interface réseau (NIC). Par exemple, avec la charge utile d'encapsulation de sécurité (ESP), la carte d'interface réseau effectue les opérations ESP pour accélérer les connexions IPsec et réduire la charge de l'unité centrale.

Par défaut, la plupart des fonctionnalités de délestage de Red Hat Enterprise Linux sont activées. Ne les désactivez que dans les cas suivants :

- Désactiver temporairement les fonctions de délestage à des fins de dépannage.
- Désactiver définitivement les fonctions de délestage lorsqu'une fonction spécifique a un impact négatif sur votre hôte.

Si une fonction de délestage liée aux performances n'est pas activée par défaut dans un pilote de réseau, vous pouvez l'activer manuellement.

31.11.1. Mise en place temporaire d'une fonction de délestage

Si vous pensez qu'une fonction de déchargement pose des problèmes ou réduit les performances de votre hôte, vous pouvez tenter d'en déterminer la cause en l'activant ou en la désactivant temporairement, en fonction de son état actuel.

Si vous activez ou désactivez temporairement une fonction de délestage, elle reprend sa valeur précédente au prochain redémarrage.

Conditions préalables

- La carte réseau prend en charge les fonctions de délestage.

Procédure

1. Affiche les fonctions de délestage disponibles de l'interface et leur état actuel :

```
# ethtool -k enp1s0
...
esp-hw-offload: on
ntuple-filters: off
rx-vlan-filter: off [fixed]
...
```

   Le résultat dépend des capacités du matériel et de son pilote. Notez que vous ne pouvez pas modifier l'état des fonctionnalités marquées par **[fixed]**.

2. Désactiver temporairement une fonction de délestage :

```
# ethtool -K <interface> <feature> [on|off]
```

   - Par exemple, pour désactiver temporairement le délestage IPsec Encapsulating Security Payload (ESP) sur l'interface **enp10s0u1**, entrez :

```
# ethtool -K enp10s0u1 esp-hw-offload off
```

   - Par exemple, pour activer temporairement le filtrage accéléré du flux de réception (aRFS) sur l'interface **enp10s0u1**, entrez :

```
# ethtool -K enp10s0u1 ntuple-filters on
```

Vérification

1. Affiche les états des fonctions de délestage :

```
# ethtool -k enp1s0
...
esp-hw-offload: off
ntuple-filters: on
...
```

2. Vérifiez si le problème que vous avez rencontré avant de modifier la fonction de délestage existe toujours.

   - Si le problème n'existe plus après la modification d'une fonction de délestage spécifique :

     i. Contactez le service d'assistance de Red Hat et signalez le problème.

     ii. Envisagez d'activer de façon permanente la fonction de délestage jusqu'à ce qu'un correctif soit disponible.

- Si le problème persiste après la désactivation d'une fonction de délestage spécifique :

  i. Réinitialiser le paramètre à son état précédent en utilisant la commande **ethtool -K *<interface> <feature> [on|off]*** en utilisant la commande

  ii. Activez ou désactivez une autre fonction de délestage pour circonscrire le problème.

Ressources supplémentaires

- **ethtool(8)** page de manuel

31.11.2. Paramétrage permanent d'une fonction de délestage

Si vous avez identifié une fonctionnalité de délestage spécifique qui limite les performances de votre hôte, vous pouvez l'activer ou la désactiver de façon permanente, en fonction de son état actuel.

Si vous activez ou désactivez de manière permanente une fonctionnalité de déchargement, NetworkManager s'assure que la fonctionnalité conserve cet état après un redémarrage.

Conditions préalables

- Vous avez identifié une fonction de délestage spécifique pour limiter les performances de votre hôte.

Procédure

1. Identifiez le profil de connexion qui utilise l'interface réseau sur laquelle vous souhaitez modifier l'état de la fonction de délestage :

```
# nmcli connection show
NAME     UUID                                  TYPE      DEVICE
Example  a5eb6490-cc20-3668-81f8-0314a27f3f75  ethernet  enp1ss0
...
```

2. Modifier de façon permanente l'état de la fonction de délestage :

```
# nmcli connection modify <connection_name> <feature> [on|off]
```

   - Par exemple, pour désactiver de façon permanente le délestage IPsec Encapsulating Security Payload (ESP) dans le profil de connexion **Example**, entrez :

```
# nmcli connection modify Example ethtool.feature-esp-hw-offload off
```

   - Par exemple, pour activer de façon permanente le filtrage accéléré du flux de réception (aRFS) dans le profil de connexion **Example**, entrez :

```
# nmcli connection modify Example ethtool.feature-ntuple on
```

3. Réactiver le profil de connexion :

```
# nmcli connection up Example
```

Vérification

- Affiche les états de sortie des fonctions de délestage :

```
# ethtool -k enp1s0
...
esp-hw-offload: off
ntuple-filters: on
...
```

Ressources supplémentaires

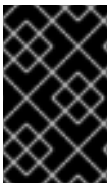
- Fonctionnalités d'offload supportées par NetworkManager

31.12. RÉGLAGE DE LA COALESCENCE D'INTERRUPTION

La coalescence des interruptions est un mécanisme permettant de réduire le nombre d'interruptions générées par une carte réseau. En règle générale, la réduction du nombre d'interruptions permet d'améliorer la latence et les performances globales de votre réseau.

Le réglage de la coalescence d'interruption implique l'ajustement des paramètres qui la contrôlent :

- Le nombre de paquets qui sont combinés en une seule interruption.
- Délai avant la génération d'une interruption.



IMPORTANT

Les paramètres optimaux de coalescence dépendent des conditions spécifiques du réseau et du matériel utilisé. Par conséquent, il faudra peut-être plusieurs tentatives pour trouver les paramètres qui conviennent le mieux à votre environnement et à vos besoins.

31.12.1. Optimisation de RHEL pour les services sensibles à la latence ou au débit

L'objectif du réglage de la coalescence est de minimiser le nombre d'interruptions nécessaires pour une charge de travail donnée. Dans les situations de haut débit, l'objectif est d'avoir le moins d'interruptions possible tout en maintenant un débit de données élevé. Dans les situations de faible latence, davantage d'interruptions peuvent être utilisées pour gérer rapidement le trafic.

Vous pouvez ajuster les paramètres de votre carte réseau pour augmenter ou diminuer le nombre de paquets qui sont combinés en une seule interruption. Vous pouvez ainsi améliorer le débit ou la latence de votre trafic.

Procédure

1. Identifiez l'interface réseau qui subit le goulot d'étranglement :

```
# ethtool -S enp1s0
NIC statistics:
    rx_packets: 1234
    tx_packets: 5678
    rx_bytes: 12345678
    tx_bytes: 87654321
    rx_errors: 0
    tx_errors: 0
    rx_missed: 0
```

```
    tx_dropped: 0
    coalesced_pkts: 0
    coalesced_events: 0
    coalesced_aborts: 0
```

Identifier les compteurs de paquets dont le nom contient "drop", "discard" ou "error". Ces statistiques particulières mesurent la perte réelle de paquets au niveau du tampon de paquets de la carte d'interface réseau (NIC), qui peut être causée par la coalescence de la NIC.

2. Surveillez les valeurs des compteurs de paquets que vous avez identifiés à l'étape précédente. Comparez-les aux valeurs attendues pour votre réseau afin de déterminer si une interface particulière présente un goulot d'étranglement. Les signes les plus courants d'un goulot d'étranglement dans le réseau sont, entre autres, les suivants

   - Nombreuses erreurs sur une interface réseau
   - Perte de paquets élevée
   - Utilisation intensive de l'interface réseau

NOTE

D'autres facteurs importants sont par exemple l'utilisation de l'unité centrale, l'utilisation de la mémoire et les entrées/sorties du disque lorsqu'il s'agit d'identifier un goulot d'étranglement dans le réseau.

3. Visualiser les paramètres de coalescence actuels :

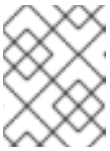
```
# ethtool enp1s0
Settings for enp1s0:
        Supported ports: [ TP ]
        Supported link modes:   10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Supported pause frame use: No
        Supports auto-negotiation: Yes
        Advertised link modes:  10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Advertised pause frame use: No
        Advertised auto-negotiation: Yes
        Speed: 1000Mb/s
        Duplex: Full
        Port: Twisted Pair
        PHYAD: 0
        Transceiver: internal
        Auto-negotiation: on
        MDI-X: Unknown
        Supports Wake-on: g
        Wake-on: g
        Current message level: 0x00000033 (51)
                               drv probe link
        Link detected: yes
```

Dans cette sortie, surveillez les champs **Speed** et **Duplex**. Ces champs affichent des informations sur le fonctionnement de l'interface réseau et indiquent si les valeurs attendues sont respectées.

4. Vérifier les paramètres actuels de coalescence d'interruption :

```
# ethtool -c enp1s0
Coalesce parameters for enp1s0:
        Adaptive RX: off
        Adaptive TX: off
        RX usecs: 100
        RX frames: 8
        RX usecs irq: 100
        RX frames irq: 8
        TX usecs: 100
        TX frames: 8
        TX usecs irq: 100
        TX frames irq: 8
```

- Les valeurs de **usecs** se réfèrent au nombre de microsecondes que le récepteur ou l'émetteur attend avant de générer une interruption.
- Les valeurs de **frames** se réfèrent au nombre de trames que le récepteur ou l'émetteur attend avant de générer une interruption.
- Les valeurs de **irq** sont utilisées pour configurer la modération d'interruption lorsque l'interface réseau gère déjà une interruption.



NOTE

Toutes les cartes d'interface réseau ne permettent pas de signaler et de modifier toutes les valeurs de l'exemple de sortie.

- La valeur **Adaptive RX/TX** représente le mécanisme adaptatif de coalescence des interruptions, qui ajuste les paramètres de coalescence des interruptions de manière dynamique. En fonction de l'état des paquets, le pilote du NIC calcule automatiquement les valeurs de coalescence lorsque **Adaptive RX/TX** est activé (l'algorithme diffère d'un pilote de NIC à l'autre).

5. Modifiez les paramètres de coalescence si nécessaire. Par exemple :

- Lorsque **ethtool.coalesce-adaptive-rx** est désactivé, configurez **ethtool.coalesce-rx-usecs** pour que le délai avant la génération d'une interruption soit de 100 microsecondes pour les paquets RX :

```
# nmcli connection modify enp1s0 ethtool.coalesce-rx-usecs 100
```

- Activer **ethtool.coalesce-adaptive-rx** alors que **ethtool.coalesce-rx-usecs** est réglé sur sa valeur par défaut :

```
# nmcli connection modify enp1s0 ethtool.coalesce-adaptive-rx on
```

Red Hat recommande de modifier le paramètre Adaptive-RX comme suit :

- Les utilisateurs soucieux d'une faible latence (inférieure à 50us) ne doivent pas activer **Adaptive-RX**.
- Les utilisateurs soucieux du débit peuvent probablement activer **Adaptive-RX** sans problème. S'ils ne veulent pas utiliser le mécanisme de coalescence adaptative des interruptions, ils peuvent essayer de fixer des valeurs importantes comme 100us ou 250us à **ethtool.coalesce-rx-usecs**.
- Les utilisateurs qui ne sont pas sûrs de leurs besoins ne devraient pas modifier ce paramètre jusqu'à ce qu'un problème survienne.

6. Réactiver la connexion :

```
# nmcli connection up enp1s0
```

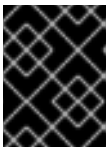
Verification steps

- Surveillez les performances du réseau et vérifiez si des paquets ont été perdus :

```
# ethtool -S enp1s0
NIC statistics:
    rx_packets: 1234
    tx_packets: 5678
    rx_bytes: 12345678
    tx_bytes: 87654321
    rx_errors: 0
    tx_errors: 0
    rx_missed: 0
    tx_dropped: 0
    coalesced_pkts: 12
    coalesced_events: 34
    coalesced_aborts: 56
...
```

La valeur des champs **rx_errors**, **rx_dropped**, **tx_errors**, et **tx_dropped** doit être égale à 0 ou proche de cette valeur (jusqu'à quelques centaines, en fonction du trafic réseau et des ressources du système). Une valeur élevée dans ces champs indique un problème de réseau. Vos compteurs peuvent avoir des noms différents. Surveillez de près les compteurs de paquets dont le nom contient "drop", "discard" ou "error".

Les valeurs de **rx_packets**, **tx_packets**, **rx_bytes**, et **tx_bytes** devraient augmenter avec le temps. Si les valeurs n'augmentent pas, il peut y avoir un problème de réseau. Les compteurs de paquets peuvent avoir des noms différents, selon le pilote de votre carte d'interface réseau.



IMPORTANT

La sortie de la commande **ethtool** peut varier en fonction du NIC et du pilote utilisés.

Les utilisateurs qui se concentrent sur une latence extrêmement faible peuvent utiliser des mesures au niveau de l'application ou l'API d'horodatage des paquets du noyau à des fins de surveillance.

Ressources supplémentaires

- Enquête initiale pour tout problème de performance
- Quels sont les paramètres du noyau disponibles pour le réglage du réseau ?
- Comment rendre les paramètres ethtool de la carte réseau persistants (appliqués automatiquement au démarrage) ?
- Horodatage

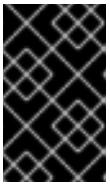
31.13. AVANTAGES DES HORODATAGES TCP

Les horodatages TCP sont des informations optionnelles dans l'en-tête TCP et une extension du protocole TCP. Par défaut, les horodatages TCP sont activés dans Red Hat Enterprise Linux, et le noyau utilise les horodatages TCP pour mieux estimer le temps d'aller-retour (RTT) dans les connexions TCP. Il en résulte des calculs plus précis de la fenêtre TCP et de la mémoire tampon.

En outre, les horodatages TCP constituent une méthode alternative pour déterminer l'âge et l'ordre d'un segment, et protègent contre les numéros de séquence enveloppés. Les en-têtes des paquets TCP enregistrent le numéro de séquence dans un champ de 32 bits. Sur une connexion à 10 Gbps, la valeur de ce champ peut être remplacée au bout de 1,7 seconde. Sans les horodatages TCP, le récepteur ne pourrait pas déterminer si un segment avec un numéro de séquence enveloppé est un nouveau segment ou un ancien duplicata. Avec les horodatages TCP, cependant, le récepteur peut choisir de recevoir ou de rejeter le segment. Il est donc essentiel d'activer les horodatages TCP sur les systèmes dotés d'interfaces réseau rapides.

Le paramètre **net.ipv4.tcp_timestamps** kernel peut prendre l'une des valeurs suivantes :

- **0**: Les horodatages TCP sont désactivés.
- **1**: Les horodatages TCP sont activés (par défaut).
- **2**: Les horodatages TCP sont activés mais sans décalage aléatoire.



IMPORTANT

Sans décalage aléatoire pour chaque connexion, il est possible de déterminer approximativement le temps de fonctionnement et l'empreinte digitale de l'hôte et d'utiliser ces informations dans des attaques.

Par défaut, les horodatages TCP sont activés dans Red Hat Enterprise Linux et utilisent des décalages aléatoires pour chaque connexion au lieu de stocker uniquement l'heure actuelle :

```
# sysctl net.ipv4.tcp_timestamps
net.ipv4.tcp_timestamps = 1
```

Si le paramètre **net.ipv4.tcp_timestamps** a une valeur différente de la valeur par défaut (**1**), inversez le réglage de la même manière que vous l'avez fait.

Ressources supplémentaires

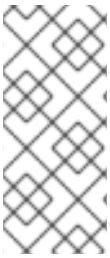
- RFC 1323 : Extensions TCP pour de hautes performances

31.14. CONTRÔLE DE FLUX POUR LES RÉSEAUX ETHERNET

Sur une liaison Ethernet, la transmission continue de données entre une interface réseau et un port de commutation peut conduire à une pleine capacité de la mémoire tampon. La pleine capacité de la mémoire tampon entraîne une congestion du réseau. Dans ce cas, lorsque l'expéditeur transmet des données à un taux supérieur à la capacité de traitement du récepteur, une perte de paquets peut se produire en raison de la plus faible capacité de traitement des données d'une interface réseau à l'autre extrémité de la liaison, qui est un port de commutation.

Le mécanisme de contrôle de flux gère la transmission de données sur la liaison Ethernet où chaque émetteur et chaque récepteur ont des capacités d'envoi et de réception différentes. Pour éviter la perte de paquets, le mécanisme de contrôle de flux Ethernet suspend temporairement la transmission de paquets pour gérer un taux de transmission plus élevé à partir d'un port de commutation. Notez que les routeurs ne transmettent pas les trames de pause au-delà d'un port de commutation.

Lorsque les tampons de réception (RX) sont pleins, un récepteur envoie des trames de pause à l'émetteur. L'émetteur interrompt alors la transmission des données pendant une courte période de moins d'une seconde, tout en continuant à mettre en mémoire tampon les données entrantes pendant cette période de pause. Cette durée permet au récepteur de vider ses tampons d'interface et d'éviter un débordement de la mémoire tampon.



NOTE

Chaque extrémité de la liaison Ethernet peut envoyer des trames de pause à une autre extrémité. Si les tampons de réception d'une interface réseau sont pleins, l'interface réseau envoie des trames de pause au port de commutation. De même, lorsque les tampons de réception d'un port de commutation sont pleins, le port de commutation envoie des trames de pause à l'interface réseau.

Par défaut, la plupart des pilotes réseau de Red Hat Enterprise Linux ont activé la prise en charge de la trame de pause. Pour afficher les paramètres actuels d'une interface réseau, entrez :

```
# ethtool --show-pause enp1s0
Pause parameters for enp1s0:
...
RX:    on
TX:    on
...
```

Vérifiez auprès du fournisseur de votre commutateur si celui-ci prend en charge les trames de pause.

Ressources supplémentaires

- **ethtool(8)** page de manuel
- Qu'est-ce que le contrôle de flux de liaison réseau et comment fonctionne-t-il dans Red Hat Enterprise Linux ?

CHAPITRE 32. FACTEURS AFFECTANT LES PERFORMANCES DES E/S ET DU SYSTÈME DE FICHIERS

Les paramètres appropriés pour le stockage et les performances du système de fichiers dépendent fortement de l'objectif du stockage.

Les performances des E/S et du système de fichiers peuvent être affectées par l'un des facteurs suivants :

- Modèles d'écriture ou de lecture de données
- Séquentielle ou aléatoire
- IO tamponnée ou directe
- Alignement des données sur la géométrie sous-jacente
- Taille du bloc
- Taille du système de fichiers
- Taille et emplacement du journal
- Enregistrement des temps d'accès
- Garantir la fiabilité des données
- Recherche préalable de données
- Pré-allocation de l'espace disque
- Fragmentation des fichiers
- Contraction des ressources

32.1. OUTILS DE SURVEILLANCE ET DE DIAGNOSTIC DES PROBLÈMES D'E/S ET DE SYSTÈME DE FICHIERS

Les outils suivants sont disponibles dans Red Hat Enterprise Linux 9 pour surveiller les performances du système et diagnostiquer les problèmes de performance liés aux E/S, aux systèmes de fichiers et à leur configuration :

- **vmstat** fournit des rapports sur les processus, la mémoire, la pagination, les blocs d'E/S, les interruptions et l'activité de l'unité centrale sur l'ensemble du système. Il peut aider les administrateurs à déterminer si le sous-système d'E/S est responsable de problèmes de performances. Si l'analyse avec **vmstat** montre que le sous-système d'E/S est responsable de la baisse des performances, les administrateurs peuvent utiliser l'outil **iostat** pour déterminer le périphérique d'E/S responsable.
- **iostat** rend compte de la charge des périphériques d'E/S dans votre système. Il est fourni par le paquetage **sysstat**.
- **blktrace** fournit des informations détaillées sur le temps passé dans le sous-système d'E/S. L'utilitaire connexe **blkparse** lit les données brutes de **blktrace** et produit un résumé lisible par l'homme des opérations d'entrée et de sortie enregistrées par **blktrace**.

- **btt** analyse la sortie de **blktrace** et affiche le temps que les données passent dans chaque zone de la pile d'E/S, ce qui permet de repérer plus facilement les goulets d'étranglement dans le sous-système d'E/S. Cet utilitaire est fourni avec le paquetage **blktrace**. Voici quelques-uns des événements importants suivis par le mécanisme **blktrace** et analysés par **btt**:
  - Mise en file d'attente de l'événement E/S (**Q**)
  - Envoi de l'événement E/S au pilote (**D**)
  - Achèvement de l'événement E/S (**C**)
- **iowatcher** peut utiliser la sortie **blktrace** pour représenter graphiquement les E/S au fil du temps. Il se concentre sur l'adresse de bloc logique (LBA) des E/S de disque, le débit en mégaoctets par seconde, le nombre de recherches par seconde et les opérations d'E/S par seconde. Cela peut aider à identifier le moment où vous atteignez la limite d'opérations par seconde d'un périphérique.
- BPF Compiler Collection (BCC) est une bibliothèque qui facilite la création de programmes Berkeley Packet Filter (**eBPF**) étendus. Les programmes **eBPF** sont déclenchés par des événements tels que des entrées/sorties de disque, des connexions TCP et des créations de processus. Les outils BCC sont installés dans le répertoire **/usr/share/bcc/tools/**. Le site **bcc-tools** suivant permet d'analyser les performances :
  - **biolatency** résume la latence dans les blocs d'E/S de périphérique (E/S de disque) sous forme d'histogramme. Cela permet d'étudier la distribution, y compris les deux modes pour les accès au cache du périphérique et pour les absences de cache, ainsi que les valeurs aberrantes de la latence.
  - **biosnoop** est un outil de traçage des blocs d'E/S de base qui permet d'afficher chaque événement d'E/S avec l'identifiant du processus émetteur et la latence d'E/S. Cet outil permet d'étudier les problèmes de performance des E/S sur disque.
  - **biotop** est utilisé pour les opérations d'entrée/sortie de blocs dans le noyau.
  - **filelife** retrace les appels de service de **stat()**.
  - **fileslower** trace des lectures et écritures synchrones et lentes de fichiers.
  - **filetop** affiche les lectures et écritures de fichiers par processus.
  - **ext4slower**, **nfsslower**, et **xfsslower** sont des outils qui montrent les opérations du système de fichiers plus lentes qu'un certain seuil, dont la valeur par défaut est **10ms**. Pour plus d'informations, voir la section Analyse des performances du système avec BPF Compiler Collection.
- **bpftace** est un langage de traçage pour **eBPF** utilisé pour analyser les problèmes de performance. Il fournit également des utilitaires de traçage tels que BCC pour l'observation du système, ce qui est utile pour étudier les problèmes de performance des E/S.
- Les scripts **SystemTap** suivants peuvent être utiles pour diagnostiquer les problèmes de performances du système de stockage ou de fichiers :
  - **disktop.stp**: Vérifie l'état de la lecture ou de l'écriture du disque toutes les 5 secondes et affiche les dix premières entrées au cours de cette période.
  - **iotime.stp**: Imprime le temps passé sur les opérations de lecture et d'écriture, ainsi que le nombre d'octets lus et écrits.

- **traceio.stp**: Imprime les dix premiers exécutables sur la base du trafic E/S cumulé observé, toutes les secondes.
- **traceio2.stp**: Imprime le nom de l'exécutable et l'identifiant du processus au fur et à mesure des lectures et des écritures sur le périphérique spécifié.
- **inodewatch.stp**: Imprime le nom de l'exécutable et l'identifiant du processus chaque fois qu'une lecture ou une écriture se produit dans l'inode spécifié sur le périphérique majeur ou mineur spécifié.
- **inodewatch2.stp**: Imprime le nom de l'exécutable, l'identifiant du processus et les attributs chaque fois que les attributs sont modifiés dans l'inode spécifié sur le périphérique majeur ou mineur spécifié.

Ressources supplémentaires

- **vmstat(8)**pages de manuel : **iostat(1)**, **blktrace(8)**, **blkparse(1)**, **btt(1)**, **bpftrace**, et **iowatcher(1)**
- Analyse des performances du système avec BPF Compiler Collection

32.2. OPTIONS DE RÉGLAGE DISPONIBLES POUR LE FORMATAGE D'UN SYSTÈME DE FICHIERS

Certaines décisions relatives à la configuration du système de fichiers ne peuvent être modifiées après le formatage de l'appareil.

Les options suivantes sont disponibles avant de formater un périphérique de stockage :

Size

Créez un système de fichiers de taille appropriée à votre charge de travail. Les systèmes de fichiers plus petits nécessitent moins de temps et de mémoire pour les contrôles du système de fichiers. Toutefois, si un système de fichiers est trop petit, ses performances sont affectées par une fragmentation importante.

Block size

Le bloc est l'unité de travail du système de fichiers. La taille du bloc détermine la quantité de données pouvant être stockées dans un seul bloc, et donc la plus petite quantité de données écrites ou lues en une seule fois.
La taille de bloc par défaut convient à la plupart des cas d'utilisation. Toutefois, votre système de fichiers fonctionne mieux et stocke les données plus efficacement si la taille du bloc ou la taille de plusieurs blocs est identique ou légèrement supérieure à la quantité de données généralement lues ou écrites en une seule fois. Un petit fichier utilise toujours un bloc entier. Les fichiers peuvent être répartis sur plusieurs blocs, mais cela peut entraîner des frais généraux d'exécution supplémentaires.

En outre, certains systèmes de fichiers sont limités à un certain nombre de blocs, ce qui limite la taille maximale du système de fichiers. La taille des blocs est spécifiée dans les options du système de fichiers lors du formatage d'un périphérique à l'aide de la commande **mkfs**. Le paramètre qui spécifie la taille des blocs varie selon le système de fichiers.

Geometry

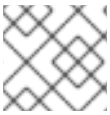
La géométrie du système de fichiers concerne la distribution des données dans un système de fichiers. Si votre système utilise un stockage par bandes, comme le RAID, vous pouvez améliorer les performances en alignant les données et les métadonnées sur la géométrie de stockage sous-jacente lorsque vous formatez le périphérique.

De nombreux périphériques exportent une géométrie recommandée, qui est ensuite définie automatiquement lorsque les périphériques sont formatés avec un système de fichiers particulier. Si votre périphérique n'exporte pas ces recommandations ou si vous souhaitez modifier les paramètres recommandés, vous devez spécifier la géométrie manuellement lorsque vous formatez le périphérique à l'aide de la commande **mkfs**.

Les paramètres qui spécifient la géométrie du système de fichiers varient en fonction du système de fichiers.

External journals

Les systèmes de fichiers de journalisation documentent les modifications qui seront apportées au cours d'une opération d'écriture dans un fichier journal avant que l'opération ne soit exécutée. Cela réduit la probabilité qu'un périphérique de stockage soit corrompu en cas de panne du système ou de coupure de courant, et accélère le processus de récupération.



NOTE

Red Hat ne recommande pas l'utilisation de l'option des journaux externes.

Les charges de travail à forte intensité de métadonnées impliquent des mises à jour très fréquentes du journal. Un journal plus volumineux utilise plus de mémoire, mais réduit la fréquence des opérations d'écriture. En outre, vous pouvez améliorer le temps de recherche d'un périphérique avec une charge de travail intensive en métadonnées en plaçant son journal sur un stockage dédié qui est aussi rapide, voire plus rapide, que le stockage principal.

AVERTISSEMENT

Veillez à ce que les journaux externes soient fiables. La perte d'un journal externe entraîne une corruption du système de fichiers. Les journaux externes doivent être créés au moment du formatage, les périphériques de journaux étant spécifiés au moment du montage.

Ressources supplémentaires

- **mkfs(8)** et **mount(8)** pages de manuel
- Aperçu des systèmes de fichiers disponibles

32.3. OPTIONS DE RÉGLAGE DISPONIBLES POUR LE MONTAGE D'UN SYSTÈME DE FICHIERS

Les options suivantes sont disponibles pour la plupart des systèmes de fichiers et peuvent être spécifiées lors du montage du périphérique :

Access Time

Chaque fois qu'un fichier est lu, ses métadonnées sont mises à jour avec l'heure à laquelle l'accès a eu lieu (**atime**). Cela implique des E/S d'écriture supplémentaires. Le paramètre **relatime** est le paramètre par défaut **atime** pour la plupart des systèmes de fichiers.

Toutefois, si la mise à jour de ces métadonnées prend du temps et si des données précises sur les temps d'accès ne sont pas nécessaires, vous pouvez monter le système de fichiers à l'aide de l'option de montage **noatime**. Cette option désactive la mise à jour des métadonnées lorsqu'un fichier est lu. Elle active également le comportement **nodiratime**, qui désactive la mise à jour des métadonnées lors de la lecture d'un répertoire.

NOTE

La désactivation des mises à jour de **atime** à l'aide de l'option **noatime mount** peut perturber les applications qui en dépendent, par exemple les programmes de sauvegarde.

Read-ahead

Read-ahead accélère l'accès aux fichiers en récupérant à l'avance les données susceptibles d'être utilisées prochainement et en les chargeant dans le cache de la page, où elles peuvent être récupérées plus rapidement que si elles se trouvaient sur le disque. Plus la valeur de lecture anticipée est élevée, plus le système récupère les données à l'avance.
Red Hat Enterprise Linux tente de définir une valeur de lecture anticipée appropriée en fonction de ce qu'il détecte sur votre système de fichiers. Cependant, une détection précise n'est pas toujours possible. Par exemple, si une matrice de stockage se présente au système comme un LUN unique, le système détecte le LUN unique et ne définit pas la valeur de lecture anticipée appropriée pour une matrice.

Les charges de travail qui impliquent un flux important d'E/S séquentielles bénéficient souvent de valeurs de lecture anticipée élevées. Les profils de réglage liés au stockage fournis avec Red Hat Enterprise Linux augmentent la valeur de l'avance de lecture, tout comme l'utilisation du striping LVM, mais ces ajustements ne sont pas toujours suffisants pour toutes les charges de travail.

Ressources supplémentaires

- **mount(8)**, **xfs(5)**, et **ext4(5)** pages de manuel

32.4. TYPES D'ÉLIMINATION DES BLOCS INUTILISÉS

L'élimination régulière des blocs qui ne sont pas utilisés par le système de fichiers est une pratique recommandée à la fois pour les disques durs et pour le stockage à provisionnement fin.

Les deux méthodes d'élimination des blocs inutilisés sont décrites ci-dessous :

Batch discard

Ce type d'exclusion fait partie de la commande **fstrim**. Il élimine tous les blocs inutilisés d'un système de fichiers qui correspondent aux critères spécifiés par l'administrateur. Red Hat Enterprise Linux 9 prend en charge l'élimination par lots sur les périphériques formatés XFS et ext4 qui prennent en charge les opérations d'élimination physique.

Online discard

Ce type d'opération d'élimination est configuré au moment du montage avec l'option d'élimination et s'exécute en temps réel sans intervention de l'utilisateur. Cependant, il n'élimine que les blocs qui sont en train de passer d'utilisés à libres. Red Hat Enterprise Linux 9 prend en charge l'élimination en ligne sur les périphériques formatés XFS et ext4.
Red Hat recommande l'élimination par lots, sauf lorsque l'élimination en ligne est nécessaire pour maintenir les performances, ou lorsque l'élimination par lots n'est pas réalisable pour la charge de travail du système.

La pré-allocation marque l'espace disque comme étant alloué à un fichier sans qu'aucune donnée ne soit écrite dans cet espace. Cela peut être utile pour limiter la fragmentation des données et les mauvaises performances de lecture. Red Hat Enterprise Linux 9 prend en charge la pré-allocation d'espace sur les systèmes de fichiers XFS, ext4 et GFS2. Les applications peuvent également bénéficier de la pré-allocation d'espace en utilisant l'appel **fallocate(2) glibc**.

Ressources supplémentaires

- **mount(8)** et **fallocate(2)** pages de manuel

32.5. CONSIDÉRATIONS SUR LA MISE AU POINT DES DISQUES À SEMI-CONDUCTEURS

Les disques d'état solide (SSD) utilisent des puces flash NAND plutôt que des plateaux magnétiques rotatifs pour stocker des données persistantes. Les disques SSD offrent un temps d'accès constant aux données sur l'ensemble de leur plage d'adresses de blocs logiques et n'entraînent pas de coûts de recherche mesurables comme leurs homologues rotatifs. Ils sont plus chers par gigaoctet d'espace de stockage et ont une densité de stockage plus faible, mais ils ont également une latence plus faible et un débit plus élevé que les disques durs.

Les performances se dégradent généralement lorsque les blocs utilisés sur un disque SSD approchent de la capacité du disque. Le degré de dégradation varie d'un fournisseur à l'autre, mais tous les dispositifs subissent une dégradation dans ces circonstances. L'activation du comportement de rejet peut contribuer à atténuer cette dégradation. Pour plus d'informations, voir Types d'élimination des blocs inutilisés.

Les options par défaut du planificateur d'E/S et de la mémoire virtuelle sont adaptées aux disques SSD. Tenez compte des facteurs suivants lors de la configuration des paramètres susceptibles d'affecter les performances des disques SSD :

I/O Scheduler

Tout planificateur d'E/S devrait fonctionner correctement avec la plupart des disques SSD. Cependant, comme pour tout autre type de stockage, Red Hat recommande d'effectuer des analyses comparatives afin de déterminer la configuration optimale pour une charge de travail donnée. Lors de l'utilisation de disques SSD, Red Hat conseille de changer le planificateur d'E/S uniquement pour l'analyse comparative de charges de travail particulières. Pour obtenir des instructions sur la manière de basculer entre les ordonnanceurs d'E/S, consultez le fichier **/usr/share/doc/kernel-version/Documentation/block/switching-sched.txt**.
Pour les HBA à file d'attente unique, le planificateur d'E/S par défaut est **deadline**. Pour les HBA à files d'attente multiples, le planificateur d'E/S par défaut est **none**. Pour plus d'informations sur la configuration du planificateur d'E/S, voir Configuration du planificateur de disque.

Virtual Memory

Tout comme le planificateur d'E/S, le sous-système de mémoire virtuelle (VM) ne nécessite aucun réglage particulier. Étant donné la nature rapide des E/S sur les disques SSD, essayez de réduire les paramètres **vm_dirty_background_ratio** et **vm_dirty_ratio**, car l'augmentation de l'activité d'écriture n'a généralement pas d'impact négatif sur la latence des autres opérations sur le disque. Cependant, ce réglage peut générer plus d'E/S globales et n'est donc généralement pas recommandé sans tests spécifiques à la charge de travail.

Swap

Un disque SSD peut également être utilisé comme périphérique d'échange, et il est susceptible de produire de bonnes performances de sortie et d'entrée de page.

32.6. PARAMÈTRES DE RÉGLAGE DU BLOC GÉNÉRIQUE

Les paramètres de réglage génériques énumérés ici sont disponibles dans le répertoire **/sys/block/sdX/queue/**.

Les paramètres de réglage énumérés ci-après sont distincts du réglage de l'ordonnanceur d'E/S et sont applicables à tous les ordonnanceurs d'E/S :

add_random

Certains événements d'E/S contribuent à la réserve d'entropie pour le site **/dev/random**. Ce paramètre peut être fixé à **0** si la surcharge de ces contributions devient mesurable.

iostats

Par défaut, **iostats** est activé et la valeur par défaut est **1**. La définition de la valeur **iostats** sur **0** désactive la collecte de statistiques d'E/S pour le périphérique, ce qui supprime une petite quantité de surcharge avec le chemin d'E/S. La définition de **iostats** à **0** peut légèrement améliorer les performances des périphériques très performants, tels que certains périphériques de stockage à semi-conducteurs NVMe. Il est recommandé de laisser **iostats** activé, sauf indication contraire du fournisseur pour un modèle de stockage donné.

Si vous désactivez le paramètre **iostats**, les statistiques d'E/S pour le périphérique ne sont plus présentes dans le fichier **/proc/diskstats**. Le contenu du fichier **/sys/diskstats** est la source des informations d'E/S pour les outils de surveillance des E/S, tels que **sar** ou **iostats**. Par conséquent, si vous désactivez le paramètre **iostats** pour un périphérique, ce dernier n'est plus présent dans les résultats des outils de surveillance des E/S.

max_sectors_kb

Spécifie la taille maximale d'une requête d'E/S en kilo-octets. La valeur par défaut est **512** KB. La valeur minimale de ce paramètre est déterminée par la taille du bloc logique de l'unité de stockage. La valeur maximale de ce paramètre est déterminée par la valeur de **max_hw_sectors_kb**.

Red Hat recommande que **max_sectors_kb** soit toujours un multiple de la taille optimale d'E/S et de la taille du bloc d'effacement interne. Utilisez une valeur de **logical_block_size** pour l'un ou l'autre des paramètres s'ils sont nuls ou non spécifiés par le périphérique de stockage.

nomerges

La plupart des charges de travail bénéficient de la fusion des requêtes. Cependant, la désactivation des fusions peut être utile à des fins de débogage. Par défaut, le paramètre **nomerges** est défini sur **0**, ce qui active la fusion. Pour désactiver la fusion simple, définissez **nomerges** sur **1**. Pour désactiver tous les types de fusion, définissez **nomerges** sur **2**.

nr_requests

Il s'agit du nombre maximum autorisé d'E/S en file d'attente. Si l'ordonnanceur d'E/S actuel est **none**, ce nombre ne peut être que réduit ; sinon, il peut être augmenté ou réduit.

optimal_io_size

Certains périphériques de stockage indiquent une taille d'E/S optimale par le biais de ce paramètre. Si cette valeur est signalée, Red Hat recommande que les applications émettent des E/S alignées sur la taille d'E/S optimale et en multiples de celle-ci dans la mesure du possible.

read_ahead_kb

Définit le nombre maximal de kilo-octets que le système d'exploitation peut lire en avance lors d'une opération de lecture séquentielle. Ainsi, les informations nécessaires sont déjà présentes dans le cache de pages du noyau pour la prochaine lecture séquentielle, ce qui améliore les performances des E/S de lecture.

Les cartographes de périphériques bénéficient souvent d'une valeur élevée de **read_ahead_kb**. **128** Une valeur de 1 Ko pour chaque périphérique à mapper est un bon point de départ, mais

l'augmentation de la valeur de **read_ahead_kb** jusqu'à la file d'attente **max_sectors_kb** du disque peut améliorer les performances dans les environnements d'application où la lecture séquentielle de fichiers volumineux a lieu.

rotational

Certains disques à l'état solide n'annoncent pas correctement leur état à l'état solide et sont montés comme des disques rotatifs traditionnels. Définissez manuellement la valeur **rotational** sur **0** pour désactiver la logique inutile de réduction de la recherche dans l'ordonnanceur.

rq_affinity

La valeur par défaut de **rq_affinity** est **1**. Il termine les opérations d'E/S sur un cœur de processeur qui se trouve dans le même groupe de processeurs que le cœur de processeur qui a émis la requête. Pour effectuer les opérations d'E/S uniquement sur le processeur qui a émis la demande d'E/S, réglez **rq_affinity** sur **2**. Pour désactiver les deux capacités mentionnées, réglez-le sur **0**.

scheduler

Pour définir l'ordonnanceur ou l'ordre de préférence de l'ordonnanceur pour un périphérique de stockage particulier, modifiez le fichier **/sys/block/*devname*/queue/scheduler** où *devname* est le nom de l'unité que vous souhaitez configurer.

CHAPITRE 33. UTILISER SYSTEMD POUR GÉRER LES RESSOURCES UTILISÉES PAR LES APPLICATIONS

RHEL 9 déplace les paramètres de gestion des ressources du niveau du processus au niveau de l'application en liant le système de hiérarchies **cgroup** à l'arborescence d'unités **systemd**. Par conséquent, vous pouvez gérer les ressources du système à l'aide de la commande **systemctl** ou en modifiant les fichiers d'unité **systemd**.

Pour ce faire, **systemd** prend diverses options de configuration dans les fichiers unitaires ou directement via la commande **systemctl**. Ensuite, **systemd** applique ces options à des groupes de processus spécifiques en utilisant les appels système du noyau Linux et des fonctions telles que **cgroups** et **namespaces**.

NOTE

Vous pouvez consulter l'ensemble des options de configuration pour **systemd** dans les pages suivantes du manuel :

- **systemd.resource-control(5)**
- **systemd.exec(5)**

33.1. ALLOCATION DES RESSOURCES SYSTÈME À L'AIDE DE SYSTEMD

Pour modifier la distribution des ressources du système, vous pouvez appliquer un ou plusieurs des modèles de distribution suivants :

Poids

Vous pouvez distribuer la ressource en additionnant les poids de tous les sous-groupes et en donnant à chaque sous-groupe la fraction correspondant à son ratio par rapport à la somme. Par exemple, si vous avez 10 cgroups, chacun avec un poids de valeur 100, la somme est de 1000. Chaque cgroup reçoit un dixième de la ressource.

Le poids est généralement utilisé pour distribuer des ressources sans état. Par exemple, l'option *CPUWeight=* est une implémentation de ce modèle de distribution des ressources.

Limites

Un cgroup peut consommer jusqu'à la quantité configurée de la ressource. La somme des limites des sous-groupes peut dépasser la limite du cgroupe parent. Il est donc possible de surcharger les ressources dans ce modèle.
Par exemple, l'option *MemoryMax=* est une mise en œuvre de ce modèle de distribution des ressources.

Protections

Vous pouvez définir une quantité protégée d'une ressource pour un cgroup. Si l'utilisation de la ressource est inférieure à la limite de protection, le noyau essaiera de ne pas pénaliser ce cgroup en faveur d'autres cgroups qui sont en concurrence pour la même ressource. Un surengagement est également possible.
Par exemple, l'option *MemoryLow=* est une mise en œuvre de ce modèle de distribution des ressources.

Allocations

Allocations exclusives d'une quantité absolue d'une ressource finie. Un surengagement n'est pas possible. Un exemple de ce type de ressource sous Linux est le budget temps réel.

option de fichier d'unité

Paramètre de configuration du contrôle des ressources.
Par exemple, vous pouvez configurer la ressource CPU avec des options telles que *CPUAccounting=* ou *CPUQuota=*. De même, vous pouvez configurer la mémoire ou les ressources E/S avec des options telles que *AllowedMemoryNodes=* et *IOAccounting=*.

Procédure

Pour modifier la valeur requise de l'option du fichier d'unités de votre service, vous pouvez ajuster la valeur dans le fichier d'unités ou utiliser la commande **systemctl**:

1. Vérifiez les valeurs attribuées pour le service de votre choix.

```
# systemctl show --propriété <unit file option> <service name>
```

2. Définir la valeur requise de l'option de politique d'allocation du temps CPU :

```
# systemctl set-property <service name> <unit file option> =<value>
```

Verification steps

- Vérifiez les valeurs nouvellement attribuées pour le service de votre choix.

```
# systemctl show --propriété <unit file option> <service name>
```

Ressources supplémentaires

- **systemd.resource-control(5)**, **systemd.exec(5)** pages de manuel

33.2. RÔLE DE SYSTEMD DANS LA GESTION DES RESSOURCES

La fonction principale de **systemd** est la gestion et la supervision des services. Le gestionnaire du système et des services **systemd** veille à ce que les services gérés démarrent au bon moment et dans le bon ordre pendant le processus de démarrage. Les services doivent fonctionner sans heurts pour utiliser de manière optimale la plate-forme matérielle sous-jacente. C'est pourquoi **systemd** fournit également des fonctionnalités permettant de définir des politiques de gestion des ressources et de régler diverses options susceptibles d'améliorer les performances du service.

IMPORTANT

En général, Red Hat vous recommande d'utiliser **systemd** pour contrôler l'utilisation des ressources du système. Vous ne devez configurer manuellement le système de fichiers virtuel **cgroups** que dans des cas particuliers. Par exemple, lorsque vous devez utiliser des contrôleurs **cgroup-v1** qui n'ont pas d'équivalents dans la hiérarchie **cgroup-v2**.

33.3. VUE D'ENSEMBLE DE LA HIÉRARCHIE DE SYSTEMD POUR LES CGROUPS

En arrière-plan, le gestionnaire de systèmes et de services **systemd** utilise les unités **slice**, **scope** et **service** pour organiser et structurer les processus dans les groupes de contrôle. Vous pouvez modifier cette hiérarchie en créant des fichiers d'unités personnalisés ou en utilisant la commande **systemctl**. En outre, **systemd** monte automatiquement les hiérarchies pour les contrôleurs de ressources importants du noyau dans le répertoire /**sys**/**fs**/**cgroup**/.

Trois types d'unités **systemd** sont utilisés pour le contrôle des ressources :

- **Service** - Un processus ou un groupe de processus, qui **systemd** démarré selon un fichier de configuration d'unité. Les services encapsulent les processus spécifiés afin qu'ils puissent être démarrés et arrêtés en tant qu'ensemble. Les services sont nommés de la manière suivante :

  ```
  <name>.service
  ```

- **Scope** - Un groupe de processus créés de l'extérieur. Les portées encapsulent les processus qui sont démarrés et arrêtés par les processus arbitraires via la fonction **fork()**, puis enregistrés par **systemd** au moment de l'exécution. Par exemple, les sessions utilisateur, les conteneurs et les machines virtuelles sont traités comme des portées. Les champs d'application sont nommés comme suit :

  ```
  <name>.scope
  ```

- **Slice** - Un groupe d'unités organisées hiérarchiquement. Les tranches organisent une hiérarchie dans laquelle sont placés les champs d'application et les services. Les processus réels sont contenus dans les scopes ou dans les services. Chaque nom d'une unité de tranche correspond au chemin d'accès à un emplacement dans la hiérarchie. Le tiret ("-") sert de séparateur entre les composants du chemin d'accès à une tranche et la tranche racine **-.slice**. Dans l'exemple suivant :

  ```
  <parent-name>.slice
  ```

  parent-name.slice est une sous-tranche de **parent.slice**, qui est une sous-tranche de la tranche racine **-.slice**. **parent-name.slice** peut avoir sa propre sous-tranche nommée **parent-name-name2.slice**, et ainsi de suite.

Les unités **service**, **scope** et **slice** sont directement associées à des objets dans la hiérarchie du groupe de contrôle. Lorsque ces unités sont activées, elles correspondent directement aux chemins des groupes de contrôle construits à partir des noms des unités.

Voici un exemple abrégé de la hiérarchie d'un groupe de contrôle :

```
Control group /:
-.slice
├─user.slice
│ ├─user-42.slice
│ │ ├─session-c1.scope
│ │ │ ├─ 967 gdm-session-worker [pam/gdm-launch-environment]
│ │ │ ├─1035 /usr/libexec/gdm-x-session gnome-session --autostart
/usr/share/gdm/greeter/autostart
│ │ │ ├─1054 /usr/libexec/Xorg vt1 -displayfd 3 -auth /run/user/42/gdm/Xauthority -background none
-noreset -keeptty -verbose 3
│ │ │ ├─1212 /usr/libexec/gnome-session-binary --autostart /usr/share/gdm/greeter/autostart
│ │ │ ├─1369 /usr/bin/gnome-shell
│ │ │ ├─1732 ibus-daemon --xim --panel disable
│ │ │ ├─1752 /usr/libexec/ibus-dconf
```

```
│ │ │ ├─1762 /usr/libexec/ibus-x11 --kill-daemon
│ │ │ ├─1912 /usr/libexec/gsd-xsettings
│ │ │ ├─1917 /usr/libexec/gsd-a11y-settings
│ │ │ ├─1920 /usr/libexec/gsd-clipboard
…
├─init.scope
│ └─1 /usr/lib/systemd/systemd --switched-root --system --deserialize 18
└─system.slice
  ├─rngd.service
  │ └─800 /sbin/rngd -f
  ├─systemd-udevd.service
  │ └─659 /usr/lib/systemd/systemd-udevd
  ├─chronyd.service
  │ └─823 /usr/sbin/chronyd
  ├─auditd.service
  │ ├─761 /sbin/auditd
  │ └─763 /usr/sbin/sedispatch
  ├─accounts-daemon.service
  │ └─876 /usr/libexec/accounts-daemon
  ├─example.service
  │ ├─ 929 /bin/bash /home/jdoe/example.sh
  │ └─4902 sleep 1
  …
```

L'exemple ci-dessus montre que les services et les champs d'application contiennent des processus et sont placés dans des tranches qui ne contiennent pas de processus propres.

Ressources supplémentaires

- *Configuring basic system settings* dans Red Hat Enterprise Linux
- Que sont les contrôleurs de ressources du noyau ?
- **systemd.resource-control(5)**pages du manuel, **systemd.exec(5)**, **cgroups(7)**, **fork()**, **fork(2)**
- Comprendre les cgroups

33.4. LISTE DES UNITÉS SYSTEMD

Utilisez le système **systemd** et le gestionnaire de services pour dresser la liste de ses unités.

Procédure

- Dressez la liste de toutes les unités actives du système à l'aide de la commande **# systemctl**. Le terminal renverra une sortie similaire à l'exemple suivant :

```
# systemctl
UNIT                                LOAD   ACTIVE SUB       DESCRIPTION
…
init.scope                          loaded active running   System and Service Manager
session-2.scope                     loaded active running   Session 2 of user jdoe
abrt-ccpp.service                   loaded active exited    Install ABRT coredump hook
abrt-oops.service                   loaded active running   ABRT kernel log watcher
abrt-vmcore.service                 loaded active exited    Harvest vmcores for ABRT
abrt-xorg.service                   loaded active running   ABRT Xorg log watcher
```

```
…
-.slice                                    loaded active active    Root Slice
machine.slice                              loaded active active    Virtual Machine and Container
Slice system-getty.slice                                           loaded active active
system-getty.slice
system-lvm2\x2dpvscan.slice                loaded active active    system-
lvm2\x2dpvscan.slice
system-sshd\x2dkeygen.slice                loaded active active    system-
sshd\x2dkeygen.slice
system-systemd\x2dhibernate\x2dresume.slice        loaded active active    system-
systemd\x2dhibernate\x2dresume>
system-user\x2druntime\x2ddir.slice            loaded active active    system-
user\x2druntime\x2ddir.slice
system.slice                               loaded active active    System Slice
user-1000.slice                            loaded active active    User Slice of UID 1000
user-42.slice                              loaded active active    User Slice of UID 42
user.slice                                 loaded active active    User and Session Slice
…
```

- **UNIT** - un nom d'unité qui reflète également la position de l'unité dans la hiérarchie d'un groupe de contrôle. Les unités pertinentes pour le contrôle des ressources sont *slice*, *scope* et *service*.
- **LOAD** - indique si le fichier de configuration de l'unité a été correctement chargé. Si le fichier de l'unité n'a pas été chargé, le champ contient l'état *error* au lieu de *loaded*. Les autres états de chargement de l'unité sont les suivants : *stub*, *merged*, et *masked*.
- **ACTIVE** - l'état d'activation de l'unité de haut niveau, qui est une généralisation de **SUB**.
- **SUB** - l'état d'activation de l'unité de bas niveau. Les valeurs possibles dépendent du type d'unité.
- **DESCRIPTION** - la description du contenu et de la fonctionnalité de l'unité.

- Liste des unités inactives.

```
# systemctl --all
```

- Limiter la quantité d'informations dans le résultat.

```
# systemctl --type service,masked
```

L'option **--type** requiert une liste de types d'unités séparés par des virgules, tels que *service* et *slice*, ou d'états de charge des unités, tels que *loaded* et *masked*.

Ressources supplémentaires

- *Configuring basic system settings* dans RHEL
- Les pages du manuel **systemd.resource-control(5)**, **systemd.exec(5)**

33.5. VISUALISATION DE LA HIÉRARCHIE DES GROUPES DE CONTRÔLE DE SYSTEMD

Afficher la hiérarchie des groupes de contrôle (**cgroups**) et les processus en cours d'exécution dans un site spécifique **cgroups**.

Procédure

- Affichez l'ensemble de la hiérarchie **cgroups** sur votre système à l'aide de la commande **systemd-cgls**.

```
# systemd-cgls
Control group /:
-.slice
├─user.slice
│ ├─user-42.slice
│ │ ├─session-c1.scope
│ │ │ ├─ 965 gdm-session-worker [pam/gdm-launch-environment]
│ │ │ └─1040 /usr/libexec/gdm-x-session gnome-session --autostart
/usr/share/gdm/greeter/autostart
…
├─init.scope
│ └─1 /usr/lib/systemd/systemd --switched-root --system --deserialize 18
└─system.slice
  …
  ├─example.service
  │ ├─6882 /bin/bash /home/jdoe/example.sh
  │ └─6902 sleep 1
  ├─systemd-journald.service
  │ └─629 /usr/lib/systemd/systemd-journald
  …
```

L'exemple de sortie renvoie l'ensemble de la hiérarchie **cgroups**, dont le niveau le plus élevé est formé par *slices*.

- Affichez la hiérarchie **cgroups** filtrée par un contrôleur de ressources avec la commande **systemd-cgls <*resource_controller*>** commande.

```
# systemd-cgls memory
Controller memory; Control group /:
├─1 /usr/lib/systemd/systemd --switched-root --system --deserialize 18
├─user.slice
│ ├─user-42.slice
│ │ ├─session-c1.scope
│ │ │ ├─ 965 gdm-session-worker [pam/gdm-launch-environment]
…
└─system.slice
  │
  …
  ├─chronyd.service
  │ └─844 /usr/sbin/chronyd
  ├─example.service
  │ ├─8914 /bin/bash /home/jdoe/example.sh
  │ └─8916 sleep 1
  …
```

L'exemple de sortie de la commande ci-dessus liste les services qui interagissent avec le contrôleur sélectionné.

- Affichez des informations détaillées sur une unité donnée et sur sa partie de la hiérarchie **cgroups** à l'aide de la commande **systemctl status <*system_unit*>** commande.

```
# systemctl status example.service
● example.service - My example service
   Loaded: loaded (/usr/lib/systemd/system/example.service; enabled; vendor preset:
disabled)
   Active: active (running) since Tue 2019-04-16 12:12:39 CEST; 3s ago
 Main PID: 17737 (bash)
    Tasks: 2 (limit: 11522)
   Memory: 496.0K (limit: 1.5M)
   CGroup: /system.slice/example.service
           ├─17737 /bin/bash /home/jdoe/example.sh
           └─17743 sleep 1
Apr 16 12:12:39 redhat systemd[1]: Started My example service.
Apr 16 12:12:39 redhat bash[17737]: The current time is Tue Apr 16 12:12:39 CEST 2019
Apr 16 12:12:40 redhat bash[17737]: The current time is Tue Apr 16 12:12:40 CEST 2019
```

Ressources supplémentaires

- Que sont les contrôleurs de ressources du noyau ?
- Les pages du manuel **systemd.resource-control(5)**, **cgroups(7)**

33.6. VISUALISATION DES GROUPES DE PROCESSUS

La procédure suivante décrit comment savoir à quel site *control group* (**cgroup**) appartient un processus. Vous pouvez ensuite consulter le site **cgroup** pour connaître les contrôleurs et les configurations spécifiques qu'il utilise.

Procédure

1. Pour savoir à quel site **cgroup** un processus appartient, exécutez la commande suivante **# cat proc/<*PID*>/cgroup** commande :

```
# cat /proc/2467/cgroup
0::/system.slice/example.service
```

   L'exemple de sortie se rapporte à un processus d'intérêt. Dans ce cas, il s'agit d'un processus identifié par **PID 2467**, qui appartient à l'unité **example.service**. Vous pouvez déterminer si le processus a été placé dans un groupe de contrôle correct, tel que défini par les spécifications du fichier de l'unité **systemd**.

2. Pour afficher les contrôleurs utilisés par le site **cgroup** et les fichiers de configuration correspondants, consultez le répertoire **cgroup**:

```
# cat /sys/fs/cgroup/system.slice/example.service/cgroup.controllers
memory pids

# ls /sys/fs/cgroup/system.slice/example.service/
cgroup.controllers
cgroup.events
…
cpu.pressure
```

```
cpu.stat
io.pressure
memory.current
memory.events
…
pids.current
pids.events
pids.max
```

NOTE

La hiérarchie de la version 1 de **cgroups** utilise un modèle par contrôleur. Par conséquent, la sortie du fichier **/proc/*PID*/cgroup** indique à quel **cgroups** de chaque contrôleur le PID appartient. Vous pouvez trouver les **cgroups** correspondants dans les répertoires des contrôleurs à l'adresse suivante **/sys/fs/cgroup/*<controller_name>*/**.

Ressources supplémentaires

- **cgroups(7)** page du manuel
- Que sont les contrôleurs de ressources du noyau ?
- Documentation dans le fichier **/usr/share/doc/kernel-doc-<kernel_version>/Documentation/admin-guide/cgroup-v2.rst** (après l'installation du paquet **kernel-doc**)

33.7. CONTRÔLE DE LA CONSOMMATION DES RESSOURCES

Affichez une liste des groupes de contrôle en cours d'exécution (**cgroups**) et leur consommation de ressources en temps réel.

Procédure

1. La commande **systemd-cgtop** permet d'afficher un compte dynamique des sites en cours d'exécution (**cgroups**).

```
# systemd-cgtop
Control Group                          Tasks   %CPU   Memory  Input/s Output/s
/                                        607   29.8     1.5G        -        -
/system.slice                            125      -   428.7M        -        -
/system.slice/ModemManager.service         3      -     8.6M        -        -
/system.slice/NetworkManager.service       3      -    12.8M        -        -
/system.slice/accounts-daemon.service      3      -     1.8M        -        -
/system.slice/boot.mount                   -      -    48.0K        -        -
/system.slice/chronyd.service              1      -     2.0M        -        -
/system.slice/cockpit.socket               -      -     1.3M        -        -
/system.slice/colord.service               3      -     3.5M        -        -
/system.slice/crond.service                1      -     1.8M        -        -
/system.slice/cups.service                 1      -     3.1M        -        -
/system.slice/dev-hugepages.mount          -      -   244.0K        -        -
/system.slice/dev-mapper-rhel\x2dswap.swap -      -   912.0K        -        -
/system.slice/dev-mqueue.mount             -      -    48.0K        -        -
```

```
/system.slice/example.service            2    -    2.0M      -      -
/system.slice/firewalld.service          2    -   28.8M      -      -
...
```

L'exemple suivant affiche les sites **cgroups** en cours d'exécution, classés en fonction de leur utilisation des ressources (CPU, mémoire, charge d'E/S sur disque). La liste est actualisée toutes les secondes par défaut. Elle offre donc un aperçu dynamique de l'utilisation réelle des ressources de chaque groupe de contrôle.

Ressources supplémentaires

- La page du manuel **systemd-cgtop(1)**

33.8. UTILISATION DES FICHIERS UNITAIRES DE SYSTEMD POUR FIXER DES LIMITES AUX APPLICATIONS

Chaque unité existante ou en cours d'exécution est supervisée par **systemd**, qui crée également des groupes de contrôle pour ces unités. Les unités ont des fichiers de configuration dans le répertoire **/usr/lib/systemd/system/**. Vous pouvez modifier manuellement les fichiers d'unité pour fixer des limites, établir des priorités ou contrôler l'accès aux ressources matérielles pour des groupes de processus.

Conditions préalables

- Vous disposez des privilèges **root**.

Procédure

1. Modifier le fichier **/usr/lib/systemd/system/example.service** pour limiter l'utilisation de la mémoire d'un service :

```
…
[Service]
MemoryMax=1500K
…
```

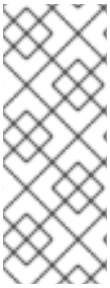
La configuration ci-dessus impose une limite de mémoire maximale que les processus d'un groupe de contrôle ne peuvent pas dépasser. Le service **example.service** fait partie d'un tel groupe de contrôle auquel des limites ont été imposées. Vous pouvez utiliser les suffixes K, M, G ou T pour identifier le kilo-octet, le méga-octet, le giga-octet ou le téra-octet comme unité de mesure.

2. Recharger tous les fichiers de configuration de l'unité :

```
# systemctl daemon-reload
```

3. Redémarrer le service :

```
# systemctl restart example.service
```



NOTE

Vous pouvez consulter l'ensemble des options de configuration pour **systemd** dans les pages suivantes du manuel :

- **systemd.resource-control(5)**
- **systemd.exec(5)**

Vérification

1. Vérifiez que les modifications ont bien été prises en compte :

```
# cat /sys/fs/cgroup/system.slice/example.service/memory.max
1536000
```

L'exemple montre que la consommation de mémoire a été limitée à environ 1 500 Ko.

Ressources supplémentaires

- Comprendre les cgroups
- *Configuring basic system settings* dans Red Hat Enterprise Linux
- **systemd.resource-control(5) systemd.exec(5)**, pages de manuel **cgroups(7)**

33.9. UTILISATION DE LA COMMANDE SYSTEMCTL POUR FIXER DES LIMITES AUX APPLICATIONS

Les paramètres d'affinité de l'unité centrale permettent de limiter l'accès d'un processus particulier à certaines unités centrales. En effet, le planificateur de CPU ne planifie jamais l'exécution d'un processus sur une unité centrale qui ne figure pas dans le masque d'affinité du processus.

Le masque d'affinité CPU par défaut s'applique à tous les services gérés par **systemd**.

Pour configurer le masque d'affinité CPU pour un service particulier **systemd**, **systemd** propose **CPUAffinity=** à la fois comme option de fichier d'unité et comme option de configuration de gestionnaire dans le fichier **/etc/systemd/system.conf**.

L'option **CPUAffinity= unit file option** définit une liste d'unités centrales ou de plages d'unités centrales qui sont fusionnées et utilisées comme masque d'affinité.

Après avoir configuré le masque d'affinité CPU pour un service **systemd** particulier, vous devez redémarrer le service pour appliquer les modifications.

Procédure

Pour définir le masque d'affinité CPU pour un service **systemd** particulier en utilisant l'option **CPUAffinity unit file** option :

1. Vérifiez les valeurs de l'option de fichier de l'unité **CPUAffinity** dans le service de votre choix :

```
systemctl show --property <CPU affinity configuration option> <service name>
```

2. En tant que root, définissez la valeur requise de l'option **CPUAffinity** unit file pour les plages de CPU utilisées comme masque d'affinité :

```
# systemctl set-property <service name> CPUAffinity=<value>
```

3. Redémarrez le service pour appliquer les modifications.

```
# systemctl restart <service name>
```

NOTE

Vous pouvez consulter l'ensemble des options de configuration pour **systemd** dans les pages suivantes du manuel :

- **systemd.resource-control(5)**
- **systemd.exec(5)**

33.10. DÉFINITION DE L'AFFINITÉ PAR DÉFAUT DE L'UNITÉ CENTRALE PAR LE BIAIS DE LA CONFIGURATION DU GESTIONNAIRE

Le fichier **CPUAffinity option** dans le fichier **/etc/systemd/system.conf** définit un masque d'affinité pour le numéro d'identification de processus (PID) 1 et tous les processus dérivés du PID1. Vous pouvez ensuite remplacer le fichier **CPUAffinity** pour chaque service.

Pour définir le masque d'affinité CPU par défaut pour tous les services systemd en utilisant l'option **manager configuration**:

1. Définissez les numéros de CPU pour l'option **CPUAffinity=** dans le fichier **/etc/systemd/system.conf**.
2. Enregistrez le fichier modifié et rechargez le service **systemd**:

```
# systemctl daemon-reload
```

3. Redémarrez le serveur pour appliquer les modifications.

NOTE

Vous pouvez consulter l'ensemble des options de configuration pour **systemd** dans les pages suivantes du manuel :

- **systemd.resource-control(5)**
- **systemd.exec(5)**

33.11. CONFIGURATION DES POLITIQUES NUMA À L'AIDE DE SYSTEMD

L'accès non uniforme à la mémoire (NUMA) est une conception de sous-système de mémoire d'ordinateur, dans laquelle le temps d'accès à la mémoire dépend de l'emplacement physique de la mémoire par rapport au processeur.

La mémoire proche de l'unité centrale a un temps de latence plus faible (mémoire locale) que la mémoire locale d'une autre unité centrale (mémoire étrangère) ou partagée entre plusieurs unités centrales.

En ce qui concerne le noyau Linux, la politique NUMA régit où (par exemple, sur quels nœuds NUMA) le noyau alloue des pages de mémoire physique pour le processus.

systemd fournit les options de fichier d'unité **NUMAPolicy** et **NUMAMask** pour contrôler les politiques d'allocation de mémoire pour les services.

Procédure

Pour définir la politique de mémoire NUMA à l'aide de l'option **NUMAPolicy unit file** option :

1. Vérifiez les valeurs de l'option de fichier de l'unité **NUMAPolicy** dans le service de votre choix :

```
$ systemctl show --property <NUMA policy configuration option> <service name>
```

2. En tant qu'utilisateur principal, définissez le type de stratégie requis pour l'option de fichier d'unité **NUMAPolicy**:

```
# systemctl set-property <service name> NUMAPolicy=<value>
```

3. Redémarrez le service pour appliquer les modifications.

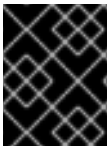
```
# systemctl restart <service name>
```

Pour définir un paramètre global **NUMAPolicy** via l'option **manager configuration**:

1. Recherchez l'option **NUMAPolicy** dans le fichier **/etc/systemd/system.conf**.
2. Modifiez le type de politique et enregistrez le fichier.
3. Recharger la configuration de **systemd**:

```
# systemd daemon-reload
```

4. Redémarrer le serveur.



IMPORTANT

Lorsque vous configurez une politique NUMA stricte, par exemple **bind**, veillez à définir également l'option de fichier d'unité **CPUAffinity=**.

Ressources supplémentaires

- Utilisation de la commande systemctl pour fixer des limites aux applications
- Les pages du manuel **systemd.resource-control(5)**, **systemd.exec(5)**, **set_mempolicy(2)**.

33.12. OPTIONS DE CONFIGURATION DE LA POLITIQUE NUMA POUR SYSTEMD

Systemd propose les options suivantes pour configurer la politique NUMA :

NUMAPolicy

Contrôle la politique de mémoire NUMA des processus exécutés. Les types de politique suivants sont possibles :

- par défaut
- préférée
- lier
- entrelacement
- local

NUMAMask

Contrôle la liste des nœuds NUMA associée à la politique NUMA sélectionnée.
Notez que l'option **NUMAMask** ne doit pas être spécifiée pour les politiques suivantes :

- par défaut
- local

Pour la stratégie préférée, la liste ne spécifie qu'un seul nœud NUMA.

Ressources supplémentaires

- **systemd.resource-control(5)**, **systemd.exec(5)**, et **set_mempolicy(2)** pages de manuel

33.13. CRÉATION DE CGROUPS TRANSITOIRES À L'AIDE DE LA COMMANDE SYSTEMD-RUN

Le site transitoire **cgroups** fixe des limites aux ressources consommées par une unité (service ou champ d'application) pendant sa durée d'exécution.

Procédure

- Pour créer un groupe de contrôle transitoire, utilisez la commande **systemd-run** dans le format suivant :

```
# systemd-run --unit=<name> --slice=<name>.slice <command>
```

Cette commande crée et démarre un service transitoire ou une unité d'étendue et exécute une commande personnalisée dans cette unité.

  - L'option **--unit=<name>** donne un nom à l'unité. Si **--unit** n'est pas spécifié, le nom est généré automatiquement.
  - L'option **--slice=<*name*>.slice** fait de votre service ou de votre unité de portée un membre d'une tranche spécifiée. Remplacez **<*name*>.slice** par le nom d'une tranche existante (comme indiqué dans la sortie de **systemctl -t slice**), ou créez une nouvelle tranche en indiquant un nom unique. Par défaut, les services et les champs d'application sont créés en tant que membres de la tranche **system.slice**.

- Remplacez ***<command>*** par la commande que vous souhaitez exécuter dans le service ou l'unité de portée.
  Le message suivant s'affiche pour confirmer que vous avez créé et démarré le service ou l'étendue avec succès :

  ```
  # Exécution en tant qu'unité <name>.service
  ```

- Il est possible de laisser l'unité fonctionner après la fin de ses processus afin de collecter des informations sur l'exécution :

  ```
  # systemd-run --unit=<name> --slice=<name>.slice --remain-after-exit <command>
  ```

  La commande crée et démarre une unité de service transitoire et exécute une commande personnalisée dans cette unité. L'option **--remain-after-exit** permet de s'assurer que le service continue de fonctionner après la fin de ses processus.

Ressources supplémentaires

- Comprendre les groupes de contrôle
- *Configuring basic system settings* dans RHEL
- la page du manuel **systemd-run(1)**

33.14. SUPPRESSION DES GROUPES DE CONTRÔLE TRANSITOIRES

Vous pouvez utiliser le gestionnaire de systèmes et de services **systemd** pour supprimer les groupes de contrôle transitoires (**cgroups**) si vous n'avez plus besoin de limiter, de hiérarchiser ou de contrôler l'accès aux ressources matérielles pour des groupes de processus.

Les sites transitoires **cgroups** sont automatiquement libérés lorsque tous les processus contenus dans un service ou une unité d'étendue sont terminés.

Procédure

- Pour arrêter l'unité de service avec tous ses processus, exécutez :

  ```
  # systemctl stop name .service
  ```

- Pour mettre fin à un ou plusieurs processus de l'unité, exécutez :

  ```
  # systemctl kill name .service --kill-who=PID,... --signal=<signal>
  ```

  La commande ci-dessus utilise l'option **--kill-who** pour sélectionner le(s) processus du groupe de contrôle que vous souhaitez arrêter. Pour tuer plusieurs processus en même temps, passez une liste de PIDs séparés par des virgules. L'option **--signal** détermine le type de signal POSIX à envoyer aux processus spécifiés. Le signal par défaut est *SIGTERM*.

Ressources supplémentaires

- Comprendre les groupes de contrôle
- Que sont les contrôleurs de ressources du noyau ?

- **systemd.resource-control(5)**, **cgroups(7)** pages de manuel
- Rôle de systemd dans les groupes de contrôle
- *Configuring basic system settings* dans RHEL

CHAPITRE 34. COMPRENDRE LES CGROUPS

Vous pouvez utiliser la fonctionnalité du noyau *control groups* (**cgroups**) pour fixer des limites, établir des priorités ou isoler les ressources matérielles des processus. Cela vous permet de contrôler granulairement l'utilisation des ressources des applications afin de les utiliser plus efficacement.

34.1. COMPRENDRE LES GROUPES DE CONTRÔLE

Control groups est une fonctionnalité du noyau Linux qui vous permet d'organiser les processus en groupes hiérarchiquement ordonnés - **cgroups**. La hiérarchie (arbre des groupes de contrôle) est définie en fournissant une structure au système de fichiers virtuel **cgroups**, monté par défaut sur le répertoire /**sys**/**fs**/**cgroup**/. Le gestionnaire de systèmes et de services **systemd** utilise **cgroups** pour organiser toutes les unités et tous les services qu'il régit. Vous pouvez également gérer manuellement les hiérarchies de **cgroups** en créant et en supprimant des sous-répertoires dans le répertoire /**sys**/**fs**/**cgroup**/.

Les contrôleurs de ressources (un composant du noyau) modifient alors le comportement des processus dans **cgroups** en limitant, en priorisant ou en allouant les ressources du système (telles que le temps de l'unité centrale, la mémoire, la largeur de bande du réseau ou diverses combinaisons) de ces processus.

La valeur ajoutée de **cgroups** est l'agrégation de processus qui permet de répartir les ressources matérielles entre les applications et les utilisateurs. Il est ainsi possible d'accroître l'efficacité globale, la stabilité et la sécurité de l'environnement des utilisateurs.

Groupes de contrôle version 1

Control groups version 1 (**cgroups-v1**) fournissent une hiérarchie de contrôleurs par ressource. Cela signifie que chaque ressource, telle que l'unité centrale, la mémoire, les E/S, etc., possède sa propre hiérarchie de groupes de contrôle. Il est possible de combiner différentes hiérarchies de groupes de contrôle de manière à ce qu'un contrôleur puisse coordonner avec un autre la gestion de leurs ressources respectives. Toutefois, les deux contrôleurs peuvent appartenir à des hiérarchies de processus différentes, ce qui ne permet pas une bonne coordination.
Les contrôleurs **cgroups-v1** ont été développés sur une longue période et, par conséquent, le comportement et la dénomination de leurs fichiers de contrôle ne sont pas uniformes.

Groupes de contrôle version 2

Les problèmes de coordination des contrôleurs, qui découlaient de la flexibilité de la hiérarchie, ont conduit au développement de *control groups version 2*.
Control groups version 2 (**cgroups-v2**) fournit une hiérarchie de groupe de contrôle unique par rapport à laquelle tous les contrôleurs de ressources sont montés.

Le comportement et la dénomination des fichiers de contrôle sont cohérents d'un contrôleur à l'autre.

IMPORTANT

Par défaut, RHEL 9 monte et utilise **cgroups-v2**.

Cette sous-section est basée sur une présentation de Devconf.cz 2019.[1]

Ressources supplémentaires

- Que sont les contrôleurs de ressources du noyau ?
- **cgroups(7)** page du manuel
- cgroups-v1
- cgroups-v2

34.2. QUE SONT LES CONTRÔLEURS DE RESSOURCES DU NOYAU ?

La fonctionnalité des groupes de contrôle est activée par les contrôleurs de ressources du noyau. RHEL 9 prend en charge différents contrôleurs pour *control groups version 1* (**cgroups-v1**) et *control groups version 2* (**cgroups-v2**).

Un contrôleur de ressources, également appelé sous-système de groupe de contrôle, est un sous-système du noyau qui représente une ressource unique, telle que le temps de l'unité centrale, la mémoire, la bande passante du réseau ou les entrées/sorties du disque. Le noyau Linux fournit une gamme de contrôleurs de ressources qui sont montés automatiquement par le système **systemd** et le gestionnaire de services. La liste des contrôleurs de ressources actuellement montés se trouve dans le fichier /**proc**/**cgroups**.

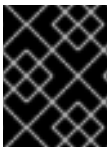
Les contrôleurs suivants sont disponibles pour **cgroups-v1**:

- **blkio** - peut fixer des limites à l'accès aux entrées/sorties vers et depuis les périphériques de bloc.
- **cpu** - peut ajuster les paramètres de l'ordonnanceur Completely Fair Scheduler (CFS) pour les tâches du groupe de contrôle. Il est monté avec le contrôleur **cpuacct** sur le même support.
- **cpuacct** - crée des rapports automatiques sur les ressources CPU utilisées par les tâches d'un groupe de contrôle. Il est monté avec le contrôleur **cpu** sur le même support.
- **cpuset** - peut être utilisé pour limiter l'exécution des tâches du groupe de contrôle à un sous-ensemble spécifié de CPU et pour ordonner aux tâches d'utiliser la mémoire uniquement sur les nœuds de mémoire spécifiés.
- **devices** - peut contrôler l'accès aux appareils pour les tâches d'un groupe de contrôle.
- **freezer** - peut être utilisé pour suspendre ou reprendre des tâches dans un groupe de contrôle.
- **memory** - peut être utilisé pour fixer des limites à l'utilisation de la mémoire par les tâches d'un groupe de contrôle et génère des rapports automatiques sur les ressources mémoire utilisées par ces tâches.
- **net_cls** - marque les paquets réseau avec un identifiant de classe (**classid**) qui permet au contrôleur de trafic Linux (la commande **tc**) d'identifier les paquets qui proviennent d'une tâche de groupe de contrôle particulière. Un sous-système de **net_cls**, **net_filter** (iptables), peut également utiliser cette étiquette pour effectuer des actions sur ces paquets. Le **net_filter** marque les sockets réseau avec un identifiant de pare-feu (**fwid**) qui permet au pare-feu Linux (via la commande **iptables**) d'identifier les paquets provenant d'une tâche particulière du groupe de contrôle.
- **net_prio** - définit la priorité du trafic réseau.
- **pids** - peut fixer des limites pour un certain nombre de processus et leurs enfants dans un groupe de contrôle.

- **perf_event** - peut regrouper les tâches à surveiller par l'utilitaire de surveillance des performances et de création de rapports **perf**.
- **rdma** - peut fixer des limites aux ressources spécifiques Remote Direct Memory Access/InfiniBand dans un groupe de contrôle.
- **hugetlb** - peut être utilisé pour limiter l'utilisation de pages de mémoire virtuelle de grande taille par les tâches d'un groupe de contrôle.

Les contrôleurs suivants sont disponibles pour **cgroups-v2**:

- **io** - Un suivi de **blkio** of **cgroups-v1**.
- **memory** - Un suivi de **memory** of **cgroups-v1**.
- **pids** - Identique à **pids** dans **cgroups-v1**.
- **rdma** - Identique à **rdma** dans **cgroups-v1**.
- **cpu** - Un suivi de **cpu** et **cpuacct** de **cgroups-v1**.
- **cpuset** - Ne prend en charge que la fonctionnalité de base (**cpus{,.effective}**, **mems{,.effective}**) avec une nouvelle fonction de partition.
- **perf_event** - La prise en charge est inhérente, il n'y a pas de fichier de contrôle explicite. Vous pouvez spécifier une adresse **v2 cgroup** en tant que paramètre de la commande **perf** qui établira le profil de toutes les tâches contenues dans cette adresse **cgroup**.



IMPORTANT

Un contrôleur de ressources peut être utilisé soit dans une hiérarchie **cgroups-v1**, soit dans une hiérarchie **cgroups-v2**, mais pas simultanément dans les deux.

Ressources supplémentaires

- **cgroups(7)** page du manuel
- Documentation dans le répertoire **/usr/share/doc/kernel-doc-<kernel_version>/Documentation/cgroups-v1/** (après avoir installé le paquetage **kernel-doc**).

34.3. QU'EST-CE QU'UN ESPACE DE NOMS ?

Les espaces de noms sont l'une des méthodes les plus importantes pour organiser et identifier les objets logiciels.

Un espace de noms enveloppe une ressource système globale (par exemple un point de montage, un périphérique réseau ou un nom d'hôte) dans une abstraction qui donne l'impression aux processus de l'espace de noms qu'ils ont leur propre instance isolée de la ressource globale. Les conteneurs sont l'une des technologies les plus courantes qui utilisent les espaces de noms.

Les modifications apportées à une ressource globale particulière ne sont visibles que par les processus de cet espace de noms et n'affectent pas le reste du système ou d'autres espaces de noms.

Pour savoir de quels espaces de noms un processus est membre, vous pouvez vérifier les liens symboliques dans le répertoire **/proc/*<PID>*/ns/** dans le répertoire

Le tableau suivant présente les espaces de noms pris en charge et les ressources qu'ils isolent :

Espace de noms	Isolats
Mount	Points de montage
UTS	Nom d'hôte et nom de domaine NIS
IPC	System V IPC, files d'attente de messages POSIX
PID	ID de processus
Network	Dispositifs de réseau, piles, ports, etc
User	ID d'utilisateur et de groupe
Control groups	Répertoire racine du groupe de contrôle

Ressources supplémentaires

- **namespaces(7)** et **cgroup_namespaces(7)** pages de manuel
- Comprendre les groupes de contrôle

[1] Linux Control Group v2 - An Introduction, Devconf.cz 2019 présentation par Waiman Long

CHAPITRE 35. AMÉLIORER LES PERFORMANCES DU SYSTÈME AVEC ZSWAP

Vous pouvez améliorer les performances du système en activant la fonction du noyau **zswap**.

35.1. QU'EST-CE QUE ZSWAP ?

zswap est une fonctionnalité du noyau qui fournit un cache RAM compressé pour les pages d'échange, ce qui peut améliorer les performances du système.

Le mécanisme fonctionne comme suit : **zswap** prend les pages qui sont en train d'être échangées et tente de les compresser dans un pool de mémoire RAM alloué dynamiquement. Lorsque le pool est plein ou que la RAM est épuisée, **zswap** expulse les pages de la mémoire cache compressée sur une base LRU (la moins récemment utilisée) vers le périphérique d'échange de sauvegarde. Une fois la page décompressée dans le cache d'échange, **zswap** libère la version compressée dans le pool.

Les avantages de lazswap

- réduction significative des E/S
- amélioration significative de la performance de la charge de travail

Dans Red Hat Enterprise Linux 9, **zswap** est activé par défaut.

Ressources supplémentaires

- Qu'est-ce que Zswap ?

35.2. ACTIVATION DE ZSWAP AU MOMENT DE L'EXÉCUTION

Vous pouvez activer la fonction **zswap** lors de l'exécution du système à l'aide de l'interface **sysfs**.

Conditions préalables

- Vous disposez des droits d'accès à la racine.

Procédure

- Activer **zswap**:

```
# echo 1 > /sys/module/zswap/parameters/enabled
```

Étape de vérification

- Vérifiez que **zswap** est activé :

```
# grep -r . /sys/kernel/debug/zswap

duplicate_entry:0
pool_limit_hit:13422200
pool_total_size:6184960 (pool size in total in pages)
reject_alloc_fail:5
```

```
reject_compress_poor:0
reject_kmemcache_fail:0
reject_reclaim_fail:13422200
stored_pages:4251 (pool size after compression)
written_back_pages:0
```

Ressources supplémentaires

- Comment activer la fonction Zswap ?

35.3. ACTIVATION PERMANENTE DE ZSWAP

Vous pouvez activer la fonction **zswap** de manière permanente en fournissant le paramètre de ligne de commande **zswap.enabled=1** kernel.

Conditions préalables

- Vous disposez des droits d'accès à la racine.
- L'utilitaire **grubby** ou **zipl** est installé sur votre système.

Procédure

1. Activer **zswap** de façon permanente :

```
# grubby --update-kernel=/boot/vmlinuz-$(uname -r) --args="zswap.enabled=1"
```

2. Redémarrez le système pour que les modifications soient prises en compte.

Verification steps

- Vérifiez que **zswap** est activé :

```
# cat /proc/cmdline

BOOT_IMAGE=(hd0,msdos1)/vmlinuz-5.14.0-70.5.1.el9_0.x86_64
root=/dev/mapper/rhel-root ro crashkernel=1G-4G:192M,4G-64G:256M,64G-:512M
resume=/dev/mapper/rhel-swap rd.lvm.lv=rhel/root
rd.lvm.lv=rhel/swap rhgb quiet
zswap.enabled=1
```

Ressources supplémentaires

- Comment activer la fonction Zswap ?
- Configuration des paramètres de la ligne de commande du noyau

CHAPITRE 36. UTILISATION DE CGROUPFS POUR GÉRER MANUELLEMENT LES CGROUPS

Vous pouvez gérer les hiérarchies **cgroup** sur votre système en créant des répertoires sur le système de fichiers virtuel **cgroupfs**. Le système de fichiers est monté par défaut sur le répertoire **/sys/fs/cgroup/** et vous pouvez spécifier les configurations souhaitées dans des fichiers de contrôle dédiés.

IMPORTANT

En général, Red Hat vous recommande d'utiliser **systemd** pour contrôler l'utilisation des ressources du système. Vous ne devez configurer manuellement le système de fichiers virtuel **cgroups** que dans des cas particuliers. Par exemple, lorsque vous devez utiliser des contrôleurs **cgroup-v1** qui n'ont pas d'équivalents dans la hiérarchie **cgroup-v2**.

36.1. CRÉATION DE CGROUPS ET ACTIVATION DE CONTRÔLEURS DANS LE SYSTÈME DE FICHIERS CGROUPS-V2

Vous pouvez gérer le répertoire *control groups* (**cgroups**) en créant ou en supprimant des répertoires et en écrivant sur les fichiers du système de fichiers virtuel **cgroups**. Le système de fichiers est monté par défaut sur le répertoire **/sys/fs/cgroup/**. Pour utiliser les paramètres des contrôleurs **cgroups**, vous devez également activer les contrôleurs souhaités pour l'enfant **cgroups**. La racine **cgroup** a, par défaut, activé les contrôleurs **memory** et **pids** pour son enfant **cgroups**. Par conséquent, Red Hat recommande de créer au moins deux niveaux d'enfants **cgroups** à l'intérieur de la racine **/sys/fs/cgroup/ cgroup** . De cette manière, vous pouvez éventuellement supprimer les contrôleurs **memory** et **pids** de l'enfant **cgroups** et maintenir une meilleure clarté organisationnelle des fichiers **cgroup**.

Conditions préalables

- Vous disposez des droits d'accès à la racine.

Procédure

1. Créez le répertoire **/sys/fs/cgroup/Example/**:

```
# mkdir /sys/fs/cgroup/Example/
```

   Le répertoire **/sys/fs/cgroup/Example/** définit un groupe enfant. Lorsque vous créez le répertoire **/sys/fs/cgroup/Example/**, certains fichiers d'interface **cgroups-v2** sont automatiquement créés dans le répertoire. Le répertoire **/sys/fs/cgroup/Example/** contient également des fichiers spécifiques aux contrôleurs **memory** et **pids**.

2. Il est possible d'inspecter le groupe de contrôle enfant nouvellement créé :

```
# ll /sys/fs/cgroup/Example/
-r—r—r--. 1 root root 0 Jun  1 10:33 cgroup.controllers
-r—r—r--. 1 root root 0 Jun  1 10:33 cgroup.events
-rw-r—r--. 1 root root 0 Jun  1 10:33 cgroup.freeze
-rw-r—r--. 1 root root 0 Jun  1 10:33 cgroup.procs
…
-rw-r—r--. 1 root root 0 Jun  1 10:33 cgroup.subtree_control
-r—r—r--. 1 root root 0 Jun  1 10:33 memory.events.local
-rw-r—r--. 1 root root 0 Jun  1 10:33 memory.high
```

```
-rw-r—r--. 1 root root 0 Jun  1 10:33 memory.low
…
-r—r—r--. 1 root root 0 Jun  1 10:33 pids.current
-r—r—r--. 1 root root 0 Jun  1 10:33 pids.events
-rw-r—r--. 1 root root 0 Jun  1 10:33 pids.max
```

L'exemple de sortie montre les fichiers généraux de l'interface de contrôle **cgroup** tels que **cgroup.procs** ou **cgroup.controllers**. Ces fichiers sont communs à tous les groupes de contrôle, quels que soient les contrôleurs activés.

Les fichiers tels que **memory.high** et **pids.max** se rapportent aux contrôleurs **memory** et **pids**, qui se trouvent dans le groupe de contrôle racine (/**sys**/**fs**/**cgroup**/), et sont activés par défaut par **systemd**.

Par défaut, le groupe enfant nouvellement créé hérite de tous les paramètres du parent **cgroup**. Dans ce cas, aucune limite n'est imposée par la racine **cgroup**.

3. Vérifiez que les contrôleurs souhaités sont disponibles dans le fichier **/sys/fs/cgroup/cgroup.controllers**:

```
# cat /sys/fs/cgroup/cgroup.controllers
cpuset cpu io memory hugetlb pids rdma
```

4. Activez les contrôleurs souhaités. Dans cet exemple, il s'agit des contrôleurs **cpu** et **cpuset**:

```
# echo "+cpu" >> /sys/fs/cgroup/cgroup.subtree_control
# echo "+cpuset" >> /sys/fs/cgroup/cgroup.subtree_control
```

Ces commandes activent les contrôleurs **cpu** et **cpuset** pour les groupes enfants immédiats du groupe de contrôle racine /**sys**/**fs**/**cgroup**/. Y compris le groupe de contrôle **Example** nouvellement créé. Un site *child group* est l'endroit où vous pouvez spécifier des processus et appliquer des contrôles à chacun des processus en fonction de vos critères.

Les utilisateurs peuvent lire le contenu du fichier **cgroup.subtree_control** à n'importe quel niveau pour avoir une idée des contrôleurs qui seront disponibles pour l'activation dans le groupe enfant immédiat.

NOTE

Par défaut, le fichier /**sys**/**fs**/**cgroup**/**cgroup.subtree_control** du groupe de contrôle racine contient les contrôleurs **memory** et **pids**.

5. Activez les contrôleurs souhaités pour l'enfant **cgroups** du groupe de contrôle **Example**:

```
# echo " cpu cpuset" >> /sys/fs/cgroup/Example/cgroup.subtree_control
```

Cette commande garantit que le groupe de contrôle enfant immédiat *only* aura des contrôleurs pertinents pour réguler la distribution du temps CPU - et non des contrôleurs **memory** ou **pids**.

6. Créez le répertoire /**sys**/**fs**/**cgroup**/**Example**/**tasks**/:

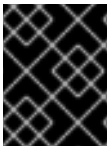
```
# mkdir /sys/fs/cgroup/Example/tasks/
```

Le répertoire /**sys**/**fs**/**cgroup**/**Example**/**tasks**/ définit un groupe enfant avec des fichiers qui se

rapportent uniquement aux contrôleurs **cpu** et **cpuset**. Vous pouvez maintenant affecter des processus à ce groupe de contrôle et utiliser les options des contrôleurs **cpu** et **cpuset** pour vos processus.

7. Optionnellement, inspecter le groupe de contrôle de l'enfant :

```
# ll /sys/fs/cgroup/Example/tasks
-r—r—r--. 1 root root 0 Jun  1 11:45 cgroup.controllers
-r—r—r--. 1 root root 0 Jun  1 11:45 cgroup.events
-rw-r—r--. 1 root root 0 Jun  1 11:45 cgroup.freeze
-rw-r—r--. 1 root root 0 Jun  1 11:45 cgroup.max.depth
-rw-r—r--. 1 root root 0 Jun  1 11:45 cgroup.max.descendants
-rw-r—r--. 1 root root 0 Jun  1 11:45 cgroup.procs
-r—r—r--. 1 root root 0 Jun  1 11:45 cgroup.stat
-rw-r—r--. 1 root root 0 Jun  1 11:45 cgroup.subtree_control
-rw-r—r--. 1 root root 0 Jun  1 11:45 cgroup.threads
-rw-r—r--. 1 root root 0 Jun  1 11:45 cgroup.type
-rw-r—r--. 1 root root 0 Jun  1 11:45 cpu.max
-rw-r—r--. 1 root root 0 Jun  1 11:45 cpu.pressure
-rw-r—r--. 1 root root 0 Jun  1 11:45 cpuset.cpus
-r—r—r--. 1 root root 0 Jun  1 11:45 cpuset.cpus.effective
-rw-r—r--. 1 root root 0 Jun  1 11:45 cpuset.cpus.partition
-rw-r—r--. 1 root root 0 Jun  1 11:45 cpuset.mems
-r—r—r--. 1 root root 0 Jun  1 11:45 cpuset.mems.effective
-r—r—r--. 1 root root 0 Jun  1 11:45 cpu.stat
-rw-r—r--. 1 root root 0 Jun  1 11:45 cpu.weight
-rw-r—r--. 1 root root 0 Jun  1 11:45 cpu.weight.nice
-rw-r—r--. 1 root root 0 Jun  1 11:45 io.pressure
-rw-r—r--. 1 root root 0 Jun  1 11:45 memory.pressure
```



IMPORTANT

Le contrôleur **cpu** n'est activé que si le groupe de contrôle enfant concerné comporte au moins deux processus qui se disputent le temps d'une seule unité centrale.

Verification steps

- Facultatif : confirmez que vous avez créé un nouveau site **cgroup** avec uniquement les contrôleurs souhaités actifs :

```
# cat /sys/fs/cgroup/Example/tasks/cgroup.controllers
cpuset cpu
```

Ressources supplémentaires

- Comprendre les groupes de contrôle
- Que sont les contrôleurs de ressources du noyau ?
- Montage de cgroups-v1
- **cgroups(7)**, **sysfs(5)** pages de manuel

36.2. CONTRÔLE DE LA RÉPARTITION DU TEMPS D'UTILISATION DE L'UNITÉ CENTRALE POUR LES APPLICATIONS EN AJUSTANT LE POIDS DE L'UNITÉ CENTRALE

Vous devez attribuer des valeurs aux fichiers pertinents du contrôleur **cpu** pour réguler la distribution du temps de CPU aux applications sous l'arborescence spécifique du cgroup.

Conditions préalables

- Vous disposez des droits d'accès à la racine.
- Vous disposez d'applications pour lesquelles vous souhaitez contrôler la répartition du temps de l'unité centrale.
- Vous avez créé une hiérarchie à deux niveaux de *child control groups* à l'intérieur de /**sys**/**fs**/**cgroup**/ *root control group* comme dans l'exemple suivant :

```
...
├── Example
│   ├── g1
│   ├── g2
│   └── g3
...
```

- Vous avez activé le contrôleur **cpu** dans le groupe de contrôle parent et dans les groupes de contrôle enfants de la même manière que celle décrite dans la section Création de cgroups et activation de contrôleurs dans le système de fichiers cgroups-v2.

Procédure

1. Configurez les poids CPU souhaités afin de respecter les restrictions de ressources au sein des groupes de contrôle :

```
# echo "150" > /sys/fs/cgroup/Example/g1/cpu.weight
# echo "100" > /sys/fs/cgroup/Example/g2/cpu.weight
# echo "50" > /sys/fs/cgroup/Example/g3/cpu.weight
```

2. Ajoutez les PID des applications aux groupes enfants **g1**, **g2**, et **g3**:

```
# echo "33373" > /sys/fs/cgroup/Example/g1/cgroup.procs
# echo "33374" > /sys/fs/cgroup/Example/g2/cgroup.procs
# echo "33377" > /sys/fs/cgroup/Example/g3/cgroup.procs
```

Les commandes de l'exemple garantissent que les applications souhaitées deviennent membres des cgroups enfants **Example**/**g***/ et que leur temps d'utilisation de l'unité centrale est réparti conformément à la configuration de ces cgroups.

Les poids des cgroups enfants (**g1**, **g2**, **g3**) qui ont des processus en cours sont additionnés au niveau du cgroup parent (**Example**). Les ressources de l'unité centrale sont ensuite réparties proportionnellement en fonction des poids respectifs.

Par conséquent, lorsque tous les processus s'exécutent en même temps, le noyau alloue à chacun d'entre eux un temps d'utilisation proportionnel basé sur le fichier **cpu.weight** de leur cgroupe respectif :

Enfant cgroup	**cpu.weight** fichier	Attribution du temps de l'unité centrale
g1	150	~50% (150/300)
g2	100	~33% (100/300)
g3	50	~16% (50/300)

La valeur du fichier du contrôleur **cpu.weight** n'est pas un pourcentage.

Si un processus cessait de fonctionner, laissant le cgroup **g2** sans aucun processus en cours, le calcul ne tiendrait pas compte du cgroup **g2** et ne prendrait en compte que les poids des cgroups **g1** et **g3**:

Enfant cgroup	**cpu.weight** fichier	Attribution du temps de l'unité centrale
g1	150	~75% (150/200)
g3	50	~25% (50/200)

IMPORTANT

Si un cgroup enfant a plusieurs processus en cours d'exécution, le temps CPU alloué au cgroup respectif sera distribué de manière égale aux processus membres de ce cgroup.

Vérification

1. Vérifiez que les applications s'exécutent dans les groupes de contrôle spécifiés :

```
# cat /proc/33373/cgroup /proc/33374/cgroup /proc/33377/cgroup
0::/Example/g1
0::/Example/g2
0::/Example/g3
```

La sortie de la commande montre les processus des applications spécifiées qui s'exécutent dans les cgroups enfants **Example/g***/.

2. Examinez la consommation actuelle de l'unité centrale des applications limitées :

```
# top
top - 05:17:18 up 1 day, 18:25,  1 user,  load average: 3.03, 3.03, 3.00
Tasks:  95 total,   4 running,  91 sleeping,   0 stopped,   0 zombie
%Cpu(s): 18.1 us, 81.6 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.3 hi,  0.0 si,  0.0 st
MiB Mem :   3737.0 total,   3233.7 free,    132.8 used,    370.5 buff/cache
MiB Swap:   4060.0 total,   4060.0 free,      0.0 used.   3373.1 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
33373 root      20   0   18720   1748   1460 R  49.5   0.0 415:05.87 sha1sum
```

```
  33374 root      20   0   18720   1756   1464 R  32.9   0.0 412:58.33 sha1sum
  33377 root      20   0   18720   1860   1568 R  16.3   0.0 411:03.12 sha1sum
    760 root      20   0  416620  28540  15296 S   0.3   0.7   0:10.23 tuned
      1 root      20   0  186328  14108   9484 S   0.0   0.4   0:02.00 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.01 kthread
...
```

NOTE

Nous avons forcé tous les processus de l'exemple à s'exécuter sur une seule unité centrale pour une illustration plus claire. Le poids de l'unité centrale applique les mêmes principes lorsqu'il est utilisé sur plusieurs unités centrales.

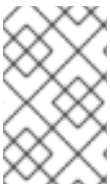
Notez que la ressource CPU pour les applications **PID 33373**, **PID 33374** et **PID 33377** a été allouée en fonction des poids, 150, 100, 50, que vous avez attribués aux groupes enfants respectifs. Ces poids correspondent à environ 50 %, 33 % et 16 % du temps d'utilisation de l'unité centrale pour chaque application.

Ressources supplémentaires

- Comprendre les groupes de contrôle
- Que sont les contrôleurs de ressources du noyau ?
- Création de cgroups et activation de contrôleurs dans le système de fichiers cgroups-v2
- Modèles de distribution des ressources
- **cgroups(7)**, **sysfs(5)** pages de manuel

36.3. MONTAGE DE CGROUPS-V1

Au cours du processus de démarrage, RHEL 9 monte par défaut le système de fichiers virtuel **cgroup-v2**. Pour utiliser la fonctionnalité **cgroup-v1** en limitant les ressources pour vos applications, configurez manuellement le système.



NOTE

cgroup-v1 et **cgroup-v2** sont tous deux pleinement activés dans le noyau. Il n'y a pas de version de groupe de contrôle par défaut du point de vue du noyau, et c'est **systemd** qui décide du montage au démarrage.

Conditions préalables

- Vous disposez des droits d'accès à la racine.

Procédure

1. Configurer le système pour qu'il monte **cgroups-v1** par défaut lors du démarrage du système par le système **systemd** et le gestionnaire de services :

```
# grubby --update-kernel=/boot/vmlinuz-$(uname -r) --
args="systemd.unified_cgroup_hierarchy=0
systemd.legacy_systemd_cgroup_controller"
```

Cette opération ajoute les paramètres de ligne de commande du noyau nécessaires à l'entrée de démarrage actuelle.

Pour ajouter les mêmes paramètres à toutes les entrées de démarrage du noyau :

```
# grubby --update-kernel=ALL --args="systemd.unified_cgroup_hierarchy=0
systemd.legacy_systemd_cgroup_controller"
```

2. Redémarrez le système pour que les modifications soient prises en compte.

Vérification

1. Optionnellement, vérifiez que le système de fichiers **cgroups-v1** a été monté :

```
# mount -l | grep cgroup
tmpfs on /sys/fs/cgroup type tmpfs
(ro,nosuid,nodev,noexec,seclabel,size=4096k,nr_inodes=1024,mode=755,inode64)
cgroup on /sys/fs/cgroup/systemd type cgroup
(rw,nosuid,nodev,noexec,relatime,seclabel,xattr,release_agent=/usr/lib/systemd/systemd-
cgroups-agent,name=systemd)
cgroup on /sys/fs/cgroup/perf_event type cgroup
(rw,nosuid,nodev,noexec,relatime,seclabel,perf_event)
cgroup on /sys/fs/cgroup/cpu,cpuacct type cgroup
(rw,nosuid,nodev,noexec,relatime,seclabel,cpu,cpuacct)
cgroup on /sys/fs/cgroup/pids type cgroup (rw,nosuid,nodev,noexec,relatime,seclabel,pids)
cgroup on /sys/fs/cgroup/cpuset type cgroup
(rw,nosuid,nodev,noexec,relatime,seclabel,cpuset)
cgroup on /sys/fs/cgroup/net_cls,net_prio type cgroup
(rw,nosuid,nodev,noexec,relatime,seclabel,net_cls,net_prio)
cgroup on /sys/fs/cgroup/hugetlb type cgroup
(rw,nosuid,nodev,noexec,relatime,seclabel,hugetlb)
cgroup on /sys/fs/cgroup/memory type cgroup
(rw,nosuid,nodev,noexec,relatime,seclabel,memory)
cgroup on /sys/fs/cgroup/blkio type cgroup (rw,nosuid,nodev,noexec,relatime,seclabel,blkio)
cgroup on /sys/fs/cgroup/devices type cgroup
(rw,nosuid,nodev,noexec,relatime,seclabel,devices)
cgroup on /sys/fs/cgroup/misc type cgroup (rw,nosuid,nodev,noexec,relatime,seclabel,misc)
cgroup on /sys/fs/cgroup/freezer type cgroup
(rw,nosuid,nodev,noexec,relatime,seclabel,freezer)
cgroup on /sys/fs/cgroup/rdma type cgroup (rw,nosuid,nodev,noexec,relatime,seclabel,rdma)
```

Les systèmes de fichiers **cgroups-v1** correspondant aux différents contrôleurs **cgroup-v1** ont été montés avec succès dans le répertoire **/sys/fs/cgroup/**.

2. Il est possible d'inspecter le contenu du répertoire **/sys/fs/cgroup/**:

```
# ll /sys/fs/cgroup/
dr-xr-xr-x. 10 root root  0 Mar 16 09:34 blkio
lrwxrwxrwx.  1 root root 11 Mar 16 09:34 cpu → cpu,cpuacct
lrwxrwxrwx.  1 root root 11 Mar 16 09:34 cpuacct → cpu,cpuacct
dr-xr-xr-x. 10 root root  0 Mar 16 09:34 cpu,cpuacct
dr-xr-xr-x.  2 root root  0 Mar 16 09:34 cpuset
dr-xr-xr-x. 10 root root  0 Mar 16 09:34 devices
dr-xr-xr-x.  2 root root  0 Mar 16 09:34 freezer
```

```
dr-xr-xr-x.  2 root root  0 Mar 16 09:34 hugetlb
dr-xr-xr-x. 10 root root  0 Mar 16 09:34 memory
dr-xr-xr-x.  2 root root  0 Mar 16 09:34 misc
lrwxrwxrwx.  1 root root 16 Mar 16 09:34 net_cls → net_cls,net_prio
dr-xr-xr-x.  2 root root  0 Mar 16 09:34 net_cls,net_prio
lrwxrwxrwx.  1 root root 16 Mar 16 09:34 net_prio → net_cls,net_prio
dr-xr-xr-x.  2 root root  0 Mar 16 09:34 perf_event
dr-xr-xr-x. 10 root root  0 Mar 16 09:34 pids
dr-xr-xr-x.  2 root root  0 Mar 16 09:34 rdma
dr-xr-xr-x. 11 root root  0 Mar 16 09:34 systemd
```

Le répertoire **/sys/fs/cgroup/**, également appelé *root control group*, par défaut, contient des répertoires spécifiques aux contrôleurs, tels que **cpuset**. En outre, il existe des répertoires liés à **systemd**.

Ressources supplémentaires

- Comprendre les groupes de contrôle
- Que sont les contrôleurs de ressources du noyau ?
- **cgroups(7)**, **sysfs(5)** pages de manuel
- cgroup-v2 activé par défaut dans RHEL 9

36.4. FIXER DES LIMITES DE CPU AUX APPLICATIONS EN UTILISANT CGROUPS-V1

Il arrive qu'une application consomme beaucoup de temps processeur, ce qui peut avoir un impact négatif sur la santé globale de votre environnement. Utilisez le système de fichiers virtuel **/sys/fs/** pour configurer des limites de CPU pour une application utilisant *control groups version 1* (**cgroups-v1**).

Conditions préalables

- Vous disposez des droits d'accès à la racine.
- Vous disposez d'une application dont vous souhaitez limiter la consommation de l'unité centrale.
- Vous avez configuré le système pour qu'il monte **cgroups-v1** par défaut lors du démarrage du système par le système **systemd** et le gestionnaire de services :

```
# grubby --update-kernel=/boot/vmlinuz-$(uname -r) --
args="systemd.unified_cgroup_hierarchy=0
systemd.legacy_systemd_cgroup_controller"
```

Cette opération ajoute les paramètres de ligne de commande du noyau nécessaires à l'entrée de démarrage actuelle.

Procédure

1. Identifiez l'ID du processus (PID) de l'application dont vous souhaitez limiter la consommation de CPU :

```
# top
```

```
top - 11:34:09 up 11 min,  1 user,  load average: 0.51, 0.27, 0.22
Tasks: 267 total,   3 running, 264 sleeping,   0 stopped,   0 zombie
%Cpu(s): 49.0 us,  3.3 sy,  0.0 ni, 47.5 id,  0.0 wa,  0.2 hi,  0.0 si,  0.0 st
MiB Mem :   1826.8 total,    303.4 free,   1046.8 used,    476.5 buff/cache
MiB Swap:   1536.0 total,   1396.0 free,    140.0 used.    616.4 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
 6955 root      20   0  228440   1752   1472 R  99.3   0.1   0:32.71 sha1sum
 5760 jdoe      20   0 3603868 205188  64196 S   3.7  11.0   0:17.19 gnome-shell
 6448 jdoe      20   0  743648  30640  19488 S   0.7   1.6   0:02.73 gnome-terminal-
    1 root      20   0  245300   6568   4116 S   0.3   0.4   0:01.87 systemd
  505 root      20   0       0      0      0 I   0.3   0.0   0:00.75 kworker/u4:4-events_unbound
...
```

L'exemple de sortie du programme **top** révèle que **PID 6955** (application illustrative **sha1sum**) consomme beaucoup de ressources de l'unité centrale.

2. Créez un sous-répertoire dans le répertoire du contrôleur de ressources **cpu**:

```
# mkdir /sys/fs/cgroup/cpu/Example/
```

Le répertoire ci-dessus représente un groupe de contrôle, dans lequel vous pouvez placer des processus spécifiques et leur appliquer certaines limites de CPU. En même temps, certains fichiers d'interface **cgroups-v1** et des fichiers spécifiques au contrôleur **cpu** seront créés dans le répertoire.

3. Il est possible d'inspecter le groupe de contrôle nouvellement créé :

```
# ll /sys/fs/cgroup/cpu/Example/
-rw-r—r--. 1 root root 0 Mar 11 11:42 cgroup.clone_children
-rw-r—r--. 1 root root 0 Mar 11 11:42 cgroup.procs
-r—r—r--. 1 root root 0 Mar 11 11:42 cpuacct.stat
-rw-r—r--. 1 root root 0 Mar 11 11:42 cpuacct.usage
-r—r—r--. 1 root root 0 Mar 11 11:42 cpuacct.usage_all
-r—r—r--. 1 root root 0 Mar 11 11:42 cpuacct.usage_percpu
-r—r—r--. 1 root root 0 Mar 11 11:42 cpuacct.usage_percpu_sys
-r—r—r--. 1 root root 0 Mar 11 11:42 cpuacct.usage_percpu_user
-r—r—r--. 1 root root 0 Mar 11 11:42 cpuacct.usage_sys
-r—r—r--. 1 root root 0 Mar 11 11:42 cpuacct.usage_user
-rw-r—r--. 1 root root 0 Mar 11 11:42 cpu.cfs_period_us
-rw-r—r--. 1 root root 0 Mar 11 11:42 cpu.cfs_quota_us
-rw-r—r--. 1 root root 0 Mar 11 11:42 cpu.rt_period_us
-rw-r—r--. 1 root root 0 Mar 11 11:42 cpu.rt_runtime_us
-rw-r—r--. 1 root root 0 Mar 11 11:42 cpu.shares
-r—r—r--. 1 root root 0 Mar 11 11:42 cpu.stat
-rw-r—r--. 1 root root 0 Mar 11 11:42 notify_on_release
-rw-r—r--. 1 root root 0 Mar 11 11:42 tasks
```

L'exemple de sortie montre des fichiers, tels que **cpuacct.usage**, **cpu.cfs._period_us**, qui représentent des configurations et/ou des limites spécifiques, qui peuvent être définies pour les processus dans le groupe de contrôle **Example**. Notez que les noms de fichiers respectifs sont précédés du nom du contrôleur du groupe de contrôle auquel ils appartiennent.

Par défaut, le groupe de contrôle nouvellement créé hérite de l'accès à l'ensemble des ressources de l'unité centrale du système, sans limite.

4. Configurer les limites de CPU pour le groupe de contrôle :

```
# echo "1000000" > /sys/fs/cgroup/cpu/Example/cpu.cfs_period_us
# echo "200000" > /sys/fs/cgroup/cpu/Example/cpu.cfs_quota_us
```

Le fichier **cpu.cfs_period_us** représente une période de temps en microsecondes (µs, représentée ici par "us") pour la fréquence à laquelle l'accès d'un groupe de contrôle aux ressources de l'unité centrale doit être réattribué. La limite supérieure est de 1 seconde et la limite inférieure de 1000 microsecondes.

Le fichier **cpu.cfs_quota_us** représente la durée totale en microsecondes pendant laquelle tous les processus d'un groupe de contrôle peuvent s'exécuter au cours d'une période (telle que définie par **cpu.cfs_period_us**). Dès que les processus d'un groupe de contrôle, au cours d'une période unique, utilisent la totalité du temps spécifié par le quota, ils sont bridés pour le reste de la période et ne sont plus autorisés à s'exécuter jusqu'à la période suivante. La limite inférieure est de 1000 microsecondes.

Les exemples de commandes ci-dessus définissent les limites de temps de l'unité centrale de sorte que tous les processus du groupe de contrôle **Example** ne puissent s'exécuter que pendant 0,2 seconde (définie par **cpu.cfs_quota_us**) sur 1 seconde (définie par **cpu.cfs_period_us**).

5. Il est possible de vérifier les limites :

```
# cat /sys/fs/cgroup/cpu/Example/cpu.cfs_period_us
/sys/fs/cgroup/cpu/Example/cpu.cfs_quota_us
1000000
200000
```

6. Ajouter le PID de l'application au groupe de contrôle **Example**:

```
# echo "6955" > /sys/fs/cgroup/cpu/Example/cgroup.procs

or

# echo "6955" > /sys/fs/cgroup/cpu/Example/tasks
```

La commande précédente garantit qu'une application souhaitée devient membre du groupe de contrôle **Example** et ne dépasse donc pas les limites de CPU configurées pour le groupe de contrôle **Example**. Le PID doit représenter un processus existant dans le système. L'adresse **PID 6955** a été attribuée au processus **sha1sum /dev/zero &**, utilisé pour illustrer le cas d'utilisation du contrôleur **cpu**.

7. Vérifiez que l'application s'exécute dans le groupe de contrôle spécifié :

```
# cat /proc/6955/cgroup
12:cpuset:/
11:hugetlb:/
10:net_cls,net_prio:/
9:memory:/user.slice/user-1000.slice/user@1000.service
8:devices:/user.slice
7:blkio:/
6:freezer:/
5:rdma:/
4:pids:/user.slice/user-1000.slice/user@1000.service
```

```
3:perf_event:/
2:cpu,cpuacct:/Example
1:name=systemd:/user.slice/user-1000.slice/user@1000.service/gnome-terminal-
server.service
```

L'exemple ci-dessus montre que le processus de l'application souhaitée s'exécute dans le groupe de contrôle **Example**, qui applique des limites de CPU au processus de l'application.

8. Identifiez la consommation actuelle de l'unité centrale de votre application limitée :

```
# top
top - 12:28:42 up  1:06,  1 user,  load average: 1.02, 1.02, 1.00
Tasks: 266 total,   6 running, 260 sleeping,   0 stopped,   0 zombie
%Cpu(s): 11.0 us,  1.2 sy,  0.0 ni, 87.5 id,  0.0 wa,  0.2 hi,  0.0 si,  0.2 st
MiB Mem :   1826.8 total,    287.1 free,   1054.4 used,    485.3 buff/cache
MiB Swap:   1536.0 total,   1396.7 free,    139.2 used.    608.3 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
 6955 root      20   0  228440   1752   1472 R  20.6   0.1  47:11.43 sha1sum
 5760 jdoe      20   0 3604956 208832  65316 R   2.3  11.2   0:43.50 gnome-shell
 6448 jdoe      20   0  743836  31736  19488 S   0.7   1.7   0:08.25 gnome-terminal-
  505 root      20   0       0      0      0 I   0.3   0.0   0:03.39 kworker/u4:4-events_unbound
 4217 root      20   0   74192   1612   1320 S   0.3   0.1   0:01.19 spice-vdagentd
...
```

Remarquez que la consommation de l'unité centrale du site **PID 6955** est passée de 99 % à 20 %.

IMPORTANT

La contrepartie de **cgroups-v2** pour **cpu.cfs_period_us** et **cpu.cfs_quota_us** est le fichier **cpu.max**. Le fichier **cpu.max** est disponible via le contrôleur **cpu**.

Ressources supplémentaires

- Comprendre les groupes de contrôle
- Ce que sont les contrôleurs de ressources du noyau
- **cgroups(7)**, **sysfs(5)** pages de manuel

CHAPITRE 37. ANALYSE DES PERFORMANCES DU SYSTÈME AVEC BPF COMPILER COLLECTION

En tant qu'administrateur système, vous pouvez utiliser la bibliothèque BPF Compiler Collection (BCC) pour créer des outils d'analyse des performances de votre système d'exploitation Linux et recueillir des informations qui pourraient être difficiles à obtenir par d'autres interfaces.

37.1. INSTALLATION DU PAQUETAGE BCC-TOOLS

Installez le paquetage **bcc-tools**, qui installe également la bibliothèque BPF Compiler Collection (BCC) en tant que dépendance.

Procédure

1. Installer **bcc-tools**.

   ```
   # dnf install bcc-tools
   ```

   Les outils BCC sont installés dans le répertoire **/usr/share/bcc/tools/**.

2. Optionnellement, inspecter les outils :

   ```
   # ll /usr/share/bcc/tools/
   ...
   -rwxr-xr-x. 1 root root  4198 Dec 14 17:53 dcsnoop
   -rwxr-xr-x. 1 root root  3931 Dec 14 17:53 dcstat
   -rwxr-xr-x. 1 root root 20040 Dec 14 17:53 deadlock_detector
   -rw-r--r--. 1 root root  7105 Dec 14 17:53 deadlock_detector.c
   drwxr-xr-x. 3 root root  8192 Mar 11 10:28 doc
   -rwxr-xr-x. 1 root root  7588 Dec 14 17:53 execsnoop
   -rwxr-xr-x. 1 root root  6373 Dec 14 17:53 ext4dist
   -rwxr-xr-x. 1 root root 10401 Dec 14 17:53 ext4slower
   ...
   ```

   Le répertoire **doc** de la liste ci-dessus contient la documentation de chaque outil.

37.2. UTILISATION DE CERTAINS OUTILS BCC POUR L'ANALYSE DES PERFORMANCES

Utilisez certains programmes prédéfinis de la bibliothèque BPF Compiler Collection (BCC) pour analyser efficacement et en toute sécurité les performances du système au cas par cas. L'ensemble des programmes pré-créés de la bibliothèque BCC peut servir d'exemple pour la création de programmes supplémentaires.

Conditions préalables

- Installation du paquet bcc-tools
- Autorisations de la racine

Utilisation de execsnoop pour examiner les processus du système

1. Exécutez le programme **execsnoop** dans un terminal :

```
# /usr/share/bcc/tools/execsnoop
```

2. Dans un autre terminal, par exemple :

```
$ ls /usr/share/bcc/tools/doc/
```

Ce qui précède crée un processus éphémère de la commande **ls**.

3. Le terminal exécutant **execsnoop** affiche une sortie similaire à la suivante :

```
PCOMM PID    PPID   RET ARGS
ls    8382   8287     0 /usr/bin/ls --color=auto /usr/share/bcc/tools/doc/
...
```

Le programme **execsnoop** imprime une ligne de sortie pour chaque nouveau processus, ce qui consomme des ressources système. Il détecte même les processus de programmes qui s'exécutent très brièvement, tels que **ls**, et que la plupart des outils de surveillance n'enregistreraient pas.

Le site **execsnoop** affiche les champs suivants :

- **PCOMM** - Le nom du processus parent. (**ls**)
- **PID** - L'ID du processus. (**8382**)
- **PPID** - L'ID du processus parent. (**8287**)
- **RET** - La valeur de retour de l'appel système **exec()** (**0**), qui charge le code du programme dans de nouveaux processus.
- **ARGS** - Emplacement du programme lancé avec les arguments.

Pour plus de détails, d'exemples et d'options concernant **execsnoop**, consultez le fichier **/usr/share/bcc/tools/doc/execsnoop_example.txt**.

Pour plus d'informations sur **exec()**, voir les pages du manuel **exec(3)**.

Utiliser opensnoop pour savoir quels fichiers une commande ouvre

1. Exécutez le programme **opensnoop** dans un terminal :

```
# /usr/share/bcc/tools/opensnoop -n uname
```

Ce qui précède imprime la sortie des fichiers qui ne sont ouverts que par le processus de la commande **uname**.

2. Dans un autre terminal, entrez :

```
$ uname
```

La commande ci-dessus ouvre certains fichiers, qui sont capturés à l'étape suivante.

3. Le terminal exécutant **opensnoop** affiche une sortie similaire à la suivante :

```
PID    COMM  FD ERR PATH
```

```
8596   uname  3  0   /etc/ld.so.cache
8596   uname  3  0   /lib64/libc.so.6
8596   uname  3  0   /usr/lib/locale/locale-archive
...
```

Le programme **opensnoop** surveille l'appel système **open()** sur l'ensemble du système et imprime une ligne de sortie pour chaque fichier que **uname** a essayé d'ouvrir en cours de route.

Le site **opensnoop** affiche les champs suivants :

- **PID** - L'ID du processus. (**8596**)
- **COMM** - Le nom du processus. (**uname**)
- **FD** - Le descripteur de fichier - une valeur que **open()** renvoie pour faire référence au fichier ouvert. (**3**)
- **ERR** - Erreurs éventuelles.
- **PATH** - Emplacement des fichiers que **open()** a tenté d'ouvrir.
  Si une commande tente de lire un fichier inexistant, la colonne **FD** renvoie **-1** et la colonne **ERR** imprime une valeur correspondant à l'erreur en question. Par conséquent, **opensnoop** peut vous aider à identifier une application qui ne se comporte pas correctement.

Pour plus de détails, d'exemples et d'options concernant **opensnoop**, consultez le fichier **/usr/share/bcc/tools/doc/opensnoop_example.txt**.

Pour plus d'informations sur **open()**, voir les pages du manuel **open(2)**.

Utilisation de biotop pour examiner les opérations d'E/S sur le disque

1. Exécutez le programme **biotop** dans un terminal :

```
# /usr/share/bcc/tools/biotop 30
```

Cette commande vous permet de surveiller les principaux processus qui effectuent des opérations d'entrée/sortie sur le disque. L'argument garantit que la commande produira un résumé de 30 secondes.

NOTE

Si aucun argument n'est fourni, l'écran de sortie est rafraîchi par défaut toutes les 1 secondes.

2. Dans un autre terminal, entrez, par exemple, :

```
# dd if=/dev/vda of=/dev/zero
```

La commande ci-dessus lit le contenu du disque dur local et écrit la sortie dans le fichier **/dev/zero**. Cette étape génère un certain trafic d'E/S pour illustrer **biotop**.

3. Le terminal exécutant **biotop** affiche une sortie similaire à la suivante :

```
PID    COMM             D MAJ MIN DISK       I/O  Kbytes     AVGms
9568   dd               R 252 0   vda      16294 14440636.0  3.69
```

```
48    kswapd0          W 252 0   vda      1763 120696.0    1.65
7571  gnome-shell      R 252 0   vda       834 83612.0     0.33
1891  gnome-shell      R 252 0   vda      1379 19792.0     0.15
7515  Xorg             R 252 0   vda       280  9940.0     0.28
7579  llvmpipe-1       R 252 0   vda       228  6928.0     0.19
9515  gnome-control-c  R 252 0   vda        62  6444.0     0.43
8112  gnome-terminal-  R 252 0   vda        67  2572.0     1.54
7807  gnome-software   R 252 0   vda        31  2336.0     0.73
9578  awk              R 252 0   vda        17  2228.0     0.66
7578  llvmpipe-0       R 252 0   vda       156  2204.0     0.07
9581  pgrep            R 252 0   vda        58  1748.0     0.42
7531  InputThread      R 252 0   vda        30  1200.0     0.48
7504  gdbus            R 252 0   vda         3  1164.0     0.30
1983  llvmpipe-1       R 252 0   vda        39   724.0     0.08
1982  llvmpipe-0       R 252 0   vda        36   652.0     0.06
...
```

Le site **biotop** affiche les champs suivants :

- **PID** - L'ID du processus. (**9568**)
- **COMM** - Le nom du processus. (**dd**)
- **DISK** - Le disque effectuant les opérations de lecture. (**vda**)
- **I/O** - Nombre d'opérations de lecture effectuées. (16294)
- **Kbytes** - Le nombre de Kbytes atteint par les opérations de lecture. (14,440,636)
- **AVGms** - Le temps d'E/S moyen des opérations de lecture. (3.69)

Pour plus de détails, d'exemples et d'options concernant **biotop**, consultez le fichier **/usr/share/bcc/tools/doc/biotop_example.txt**.

Pour plus d'informations sur **dd**, voir les pages du manuel **dd(1)**.

Utilisation de xfsslower pour révéler les lenteurs inattendues du système de fichiers

1. Exécutez le programme **xfsslower** dans un terminal :

```
# /usr/share/bcc/tools/xfsslower 1
```

La commande ci-dessus mesure le temps que le système de fichiers XFS passe à effectuer des opérations de lecture, d'écriture, d'ouverture ou de synchronisation (**fsync**). L'argument **1** garantit que le programme n'affiche que les opérations qui sont plus lentes que 1 ms.

NOTE

En l'absence d'arguments, **xfsslower** affiche par défaut les opérations plus lentes que 10 ms.

2. Dans un autre terminal, entrez, par exemple, ce qui suit :

```
$ vim text
```

La commande ci-dessus crée un fichier texte dans l'éditeur **vim** afin d'initier certaines interactions avec le système de fichiers XFS.

3. Le terminal qui exécute **xfsslower** affiche quelque chose de similaire après avoir sauvegardé le fichier de l'étape précédente :

```
TIME     COMM           PID    T BYTES   OFF_KB   LAT(ms) FILENAME
13:07:14 b'bash'        4754   R 256     0          7.11 b'vim'
13:07:14 b'vim'         4754   R 832     0          4.03 b'libgpm.so.2.1.0'
13:07:14 b'vim'         4754   R 32      20         1.04 b'libgpm.so.2.1.0'
13:07:14 b'vim'         4754   R 1982    0          2.30 b'vimrc'
13:07:14 b'vim'         4754   R 1393    0          2.52 b'getscriptPlugin.vim'
13:07:45 b'vim'         4754   S 0       0          6.71 b'text'
13:07:45 b'pool'        2588   R 16      0          5.58 b'text'
...
```

Chaque ligne ci-dessus représente une opération dans le système de fichiers qui a pris plus de temps qu'un certain seuil. **xfsslower** est capable d'exposer les problèmes éventuels du système de fichiers, qui peuvent prendre la forme d'opérations inopinément lentes.

Le site **xfsslower** affiche les champs suivants :

- **COMM** - Le nom du processus. (**b'bash'**)
- **T** - Le type d'opération. (**R**)
  - **R**tête
  - **W**rite
  - **S**ync
- **OFF_KB** - Le décalage du fichier en Ko. (0)
- **FILENAME** - Le fichier en cours de lecture, d'écriture ou de synchronisation.

Pour plus de détails, d'exemples et d'options concernant **xfsslower**, consultez le fichier **/usr/share/bcc/tools/doc/xfsslower_example.txt**.

Pour plus d'informations sur **fsync**, voir les pages du manuel **fsync(2)**.

CHAPITRE 38. CONFIGURER UN SYSTÈME D'EXPLOITATION POUR OPTIMISER L'ACCÈS À LA MÉMOIRE

Vous pouvez configurer le système d'exploitation pour optimiser l'accès à la mémoire entre les charges de travail à l'aide des outils inclus dans RHEL

38.1. OUTILS DE SURVEILLANCE ET DE DIAGNOSTIC DES PROBLÈMES DE MÉMOIRE DU SYSTÈME

Les outils suivants sont disponibles dans Red Hat Enterprise Linux 9 pour surveiller les performances du système et diagnostiquer les problèmes de performance liés à la mémoire du système :

- **vmstat** fourni par le paquetage **procps-ng**, affiche des rapports sur les processus, la mémoire, la pagination, les E/S par bloc, les pièges, les disques et l'activité de l'unité centrale d'un système. Il fournit un rapport instantané de la moyenne de ces événements depuis la dernière mise sous tension de la machine ou depuis le rapport précédent.

- **valgrind** permet d'instrumenter les binaires de l'espace utilisateur. Installez cet outil à l'aide de la commande **dnf install valgrind**. Il comprend un certain nombre d'outils, que vous pouvez utiliser pour profiler et analyser les performances des programmes, tels que :

  - **memcheck** est l'outil par défaut de **valgrind**. Il détecte et signale un certain nombre d'erreurs de mémoire qui peuvent être difficiles à détecter et à diagnostiquer, telles que :

    - Accès à la mémoire qui ne devrait pas avoir lieu
    - Utilisation d'une valeur non définie ou non initialisée
    - La mémoire du tas a été libérée de manière incorrecte
    - Chevauchement de pointeurs
    - Fuites de mémoire

    NOTE

    Memcheck ne peut que signaler ces erreurs, il ne peut pas les empêcher de se produire. Cependant, **memcheck** enregistre un message d'erreur immédiatement avant que l'erreur ne se produise.

  - **cachegrind** simule l'interaction d'une application avec la hiérarchie de cache et le prédicteur de branche d'un système. Elle recueille des statistiques pendant la durée d'exécution de l'application et affiche un résumé sur la console.

  - **massif** mesure l'espace du tas utilisé par une application donnée. Elle mesure à la fois l'espace utile et tout espace supplémentaire alloué à des fins de comptabilité et d'alignement.

Ressources supplémentaires

- **vmstat(8)** et **valgrind(1)** pages de manuel
- **/usr/share/doc/valgrind-version/valgrind_manual.pdf** fichier

38.2. VUE D'ENSEMBLE DE LA MÉMOIRE D'UN SYSTÈME

Le noyau Linux est conçu pour maximiser l'utilisation des ressources mémoire d'un système (RAM). En raison de ces caractéristiques de conception, et en fonction des besoins en mémoire de la charge de travail, une partie de la mémoire du système est utilisée par le noyau pour le compte de la charge de travail, tandis qu'une petite partie de la mémoire est libre. Cette mémoire libre est réservée à des allocations spéciales du système et à d'autres services système de priorité faible ou élevée.

Le reste de la mémoire du système est consacré à la charge de travail elle-même et se divise en deux catégories :

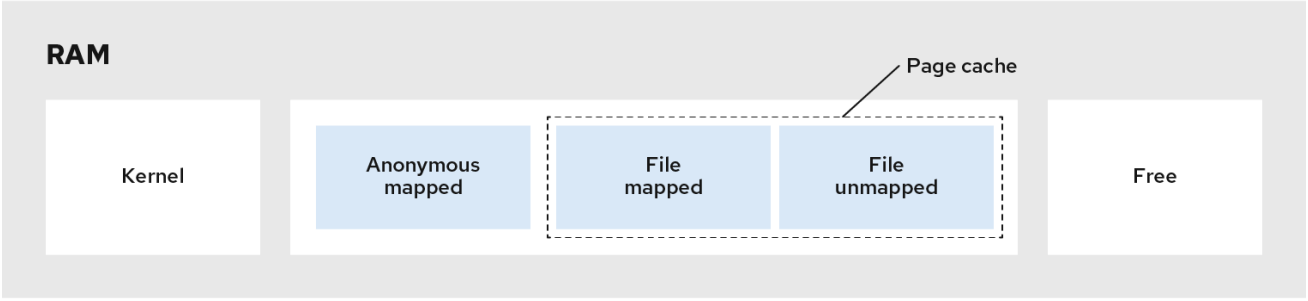
File memory

Les pages ajoutées dans cette catégorie représentent des parties de fichiers stockés en permanence. Ces pages, issues du cache de pages, peuvent être mappées ou démappées dans les espaces d'adressage d'une application. Vous pouvez utiliser des applications pour mapper des fichiers dans leur espace d'adressage à l'aide des appels système **mmap**, ou pour opérer sur des fichiers via les appels système de lecture ou d'écriture d'E/S en mémoire tampon.
Les appels système d'E/S en mémoire tampon, ainsi que les applications qui mappent directement les pages, peuvent réutiliser les pages non mappées. Par conséquent, ces pages sont stockées dans le cache par le noyau, en particulier lorsque le système n'exécute pas de tâches gourmandes en mémoire, afin d'éviter de réémettre des opérations d'E/S coûteuses sur le même ensemble de pages.

Anonymous memory

Les pages de cette catégorie sont utilisées par un processus alloué dynamiquement ou ne sont pas liées à des fichiers stockés en permanence. Cet ensemble de pages soutient les structures de contrôle en mémoire de chaque tâche, telles que la pile d'application et les zones de tas.

Figure 38.1. Modèles d'utilisation de la mémoire



38.3. PARAMÈTRES DE LA MÉMOIRE VIRTUELLE

Les paramètres de la mémoire virtuelle sont répertoriés dans le répertoire **/proc/sys/vm**.

Les paramètres de mémoire virtuelle disponibles sont les suivants :

vm.dirty_ratio

Est une valeur en pourcentage. Lorsque ce pourcentage de la mémoire totale du système est modifié, le système commence à écrire les modifications sur le disque avec l'opération **pdflush**. La valeur par défaut est **20** pour cent.

vm.dirty_background_ratio

Une valeur en pourcentage. Lorsque ce pourcentage de la mémoire totale du système est modifié, le système commence à écrire les modifications sur le disque en arrière-plan. La valeur par défaut est **10**.

vm.overcommit_memory

Définit les conditions qui déterminent si une demande de mémoire importante est acceptée ou refusée. La valeur par défaut est **0**.
Par défaut, le noyau vérifie si une demande d'allocation de mémoire virtuelle est compatible avec la quantité de mémoire disponible (swap total) et ne rejette que les demandes importantes. Dans le cas contraire, les allocations de mémoire virtuelle sont accordées, ce qui signifie qu'elles autorisent le surengagement de la mémoire.

Réglage de la valeur du paramètre **overcommit_memory**:

- Lorsque ce paramètre est fixé à **1**, le noyau n'effectue aucune gestion de surcharge de la mémoire. Cela augmente le risque de surcharge de la mémoire, mais améliore les performances pour les tâches nécessitant beaucoup de mémoire.

- Lorsque ce paramètre est défini sur **2**, le noyau refuse les demandes de mémoire égales ou supérieures à la somme de l'espace d'échange total disponible et du pourcentage de RAM physique spécifié dans **overcommit_ratio**. Cela réduit le risque de surutilisation de la mémoire, mais n'est recommandé que pour les systèmes dont l'espace d'échange est plus grand que la mémoire physique.

vm.overcommit_ratio

Spécifie le pourcentage de RAM physique pris en compte lorsque **overcommit_memory** est défini sur **2**. La valeur par défaut est **50**.

vm.max_map_count

Définit le nombre maximal de zones de la carte mémoire qu'un processus peut utiliser. La valeur par défaut est **65530**. Augmentez cette valeur si votre application a besoin de plus de zones de carte mémoire.

vm.min_free_kbytes

Définit la taille du pool de pages libres réservées. Il est également responsable de la définition des seuils **min_page**, **low_page**, et **high_page** qui régissent le comportement des algorithmes de récupération de pages du noyau Linux. Il spécifie également le nombre minimum de kilo-octets à garder libres dans le système. Il calcule une valeur spécifique pour chaque zone de mémoire basse, qui se voit attribuer un nombre de pages libres réservées proportionnel à sa taille.
Réglage de la valeur du paramètre **vm.min_free_kbytes**:

- L'augmentation de la valeur du paramètre réduit effectivement la mémoire utilisable de l'ensemble de travail de l'application. Par conséquent, vous ne devriez l'utiliser que pour les charges de travail pilotées par le noyau, où les tampons des pilotes doivent être alloués dans des contextes atomiques.

- La diminution de la valeur du paramètre peut rendre le noyau incapable de répondre aux demandes du système, si la mémoire devient très sollicitée dans le système.

AVERTISSEMENT

Les valeurs extrêmes peuvent nuire aux performances du système. En fixant la valeur de **vm.min_free_kbytes** à un niveau extrêmement bas, on empêche le système de récupérer la mémoire de manière efficace, ce qui peut entraîner des pannes du système et l'impossibilité de gérer les interruptions ou d'autres services du noyau. Cependant, une valeur trop élevée de **vm.min_free_kbytes** augmente considérablement l'activité de récupération du système, entraînant une latence d'allocation due à un faux état de récupération directe. Cela peut entraîner l'entrée immédiate du système dans un état de mémoire insuffisante.

Le paramètre **vm.min_free_kbytes** définit également un filigrane de récupération de page, appelé **min_pages**. Ce filigrane est utilisé comme facteur pour déterminer les deux autres filigranes de mémoire, **low_pages** et **high_pages**, qui régissent les algorithmes de récupération des pages.

/proc/*PID*/oom_adj

Si un système manque de mémoire et que le paramètre **panic_on_oom** est fixé à **0**, la fonction **oom_killer** tue les processus, en commençant par le processus qui a la valeur **oom_score** la plus élevée, jusqu'à ce que le système se rétablisse.

Le paramètre **oom_adj** détermine le **oom_score** d'un processus. Ce paramètre est défini pour chaque identifiant de processus. Une valeur de **-17** désactive **oom_killer** pour ce processus. Les autres valeurs valables sont comprises entre **-16** et **15**.

NOTE

Les processus créés par un processus ajusté héritent de l'adresse **oom_score** de ce processus.

vm.swappiness

La valeur de permutation, comprise entre **0** et **200**, détermine dans quelle mesure le système favorise la récupération de la mémoire dans le pool de mémoire anonyme ou dans le pool de mémoire du cache de pages.

Réglage de la valeur du paramètre **swappiness**:

- Des valeurs plus élevées favorisent les charges de travail basées sur les fichiers tout en éliminant la mémoire anonyme de la RAM des processus les moins activement accédés. Ceci est utile pour les serveurs de fichiers ou les applications de streaming qui dépendent des données, des fichiers dans le stockage, pour résider dans la mémoire afin de réduire la latence d'E/S pour les demandes de service.

- Des valeurs faibles favorisent les charges de travail basées sur le mappage anonyme tout en récupérant le cache de page (mémoire mappée sur fichier). Ce paramètre est utile pour les applications qui ne dépendent pas fortement des informations du système de fichiers et qui utilisent fortement la mémoire privée et allouée dynamiquement, telles que les applications mathématiques et de calcul, et quelques superviseurs de virtualisation matérielle comme QEMU.

  La valeur par défaut du paramètre **vm.swappiness** est **60**.

AVERTISSEMENT

En définissant **vm.swappiness** sur **0**, on évite agressivement d'échanger la mémoire anonyme sur un disque, ce qui augmente le risque que les processus soient tués par la fonction **oom_killer** en cas de charge de travail intensive en termes de mémoire ou d'entrées/sorties.

Ressources supplémentaires

- **sysctl(8)** page de manuel
- Réglage des paramètres du noyau liés à la mémoire

38.4. PARAMÈTRES DU SYSTÈME DE FICHIERS

Les paramètres du système de fichiers sont répertoriés dans le répertoire **/proc/sys/fs**. Les paramètres du système de fichiers disponibles sont les suivants :

aio-max-nr

Définit le nombre maximal d'événements autorisés dans tous les contextes d'entrée/sortie asynchrones actifs. La valeur par défaut est **65536**, et la modification de cette valeur n'entraîne pas de pré-allocation ou de redimensionnement des structures de données du noyau.

file-max

Détermine le nombre maximum de gestionnaires de fichiers pour l'ensemble du système. La valeur par défaut sur Red Hat Enterprise Linux 9 est soit **8192**, soit un dixième des pages de mémoire libres disponibles au moment du démarrage du noyau, la valeur la plus élevée étant retenue. L'augmentation de cette valeur peut résoudre les erreurs causées par un manque de gestionnaires de fichiers disponibles.

Ressources supplémentaires

- **sysctl(8)** page de manuel

38.5. PARAMÈTRES DU NOYAU

Les valeurs par défaut des paramètres du noyau se trouvent dans le répertoire **/proc/sys/kernel/**. Il s'agit de valeurs par défaut fournies par le noyau ou de valeurs spécifiées par un utilisateur via **sysctl**.

Voici les paramètres du noyau disponibles utilisés pour définir les limites des appels système **msg*** et **shm*** System V IPC (**sysvipc**) :

msgmax

Définit la taille maximale autorisée en octets d'un seul message dans une file d'attente. Cette valeur ne doit pas dépasser la taille de la file d'attente (**msgmnb**). Utilisez la commande **sysctl msgmax** pour déterminer la valeur actuelle de **msgmax** sur votre système.

msgmnb

Définit la taille maximale en octets d'une file d'attente de messages unique. Utilisez la commande **sysctl msgmnb** pour déterminer la valeur actuelle de **msgmnb** sur votre système.

msgmni

Définit le nombre maximum d'identifiants de file d'attente de messages, et donc le nombre maximum de files d'attente. Utilisez la commande **sysctl msgmni** pour déterminer la valeur actuelle de **msgmni** sur votre système.

shmall

Définit la quantité totale de mémoire partagée **pages** qui peut être utilisée sur le système à un moment donné. Par exemple, une page représente **4096** octets sur les architectures AMD64 et Intel 64. Utilisez la commande **sysctl shmall** pour déterminer la valeur actuelle de **shmall** sur votre système.

shmmax

Définit la taille maximale en octets d'un segment de mémoire partagée autorisé par le noyau. Les segments de mémoire partagée allant jusqu'à 1 Go sont désormais pris en charge par le noyau. Utilisez la commande **sysctl shmmax** pour déterminer la valeur actuelle de **shmmax** sur votre système.

shmmni

Définit le nombre maximal de segments de mémoire partagée à l'échelle du système. La valeur par défaut est **4096** sur tous les systèmes.

Ressources supplémentaires

- **sysvipc(7)** et **sysctl(8)** pages de manuel

38.6. RÉGLAGE DES PARAMÈTRES DU NOYAU LIÉS À LA MÉMOIRE

Le réglage temporaire d'un paramètre est utile pour déterminer l'effet du paramètre sur un système. Vous pouvez ensuite définir le paramètre de manière permanente lorsque vous êtes sûr que la valeur du paramètre a l'effet désiré.

Cette procédure décrit comment définir un paramètre du noyau lié à la mémoire de manière temporaire et persistante.

Procédure

- Pour définir temporairement les paramètres du noyau liés à la mémoire, éditez les fichiers correspondants dans le système de fichiers /**proc** ou l'outil **sysctl**.
  Par exemple, pour régler temporairement le paramètre **vm.overcommit_memory** sur *1*:

```
# echo 1 > /proc/sys/vm/overcommit_memory
# sysctl -w vm.overcommit_memory=1
```

- Pour définir de manière persistante le paramètre du noyau lié à la mémoire, modifiez le fichier /**etc**/**sysctl.conf** et rechargez les paramètres.
  Par exemple, pour définir de manière persistante le paramètre **vm.overcommit_memory** sur *1*:
  - Ajoutez le contenu suivant dans le fichier /**etc**/**sysctl.conf**:

```
vm.overcommit_memory=1
```

  - Rechargez les paramètres de **sysctl** à partir du fichier /**etc**/**sysctl.conf**:

```
# sysctl -p
```

Ressources supplémentaires

- **sysctl(8)** page de manuel
- **proc(5)** page de manuel

CHAPITRE 39. CONFIGURATION DE PAGES VOLUMINEUSES

La mémoire physique est gérée en morceaux de taille fixe appelés pages. Sur l'architecture x86_64, prise en charge par Red Hat Enterprise Linux 9, la taille par défaut d'une page de mémoire est **4 KB**. Cette taille de page par défaut s'est avérée adaptée aux systèmes d'exploitation à usage général, tels que Red Hat Enterprise Linux, qui prend en charge de nombreux types de charges de travail.

Toutefois, des applications spécifiques peuvent bénéficier de l'utilisation de pages plus grandes dans certains cas. Par exemple, une application qui travaille avec un ensemble de données volumineux et relativement fixe de plusieurs centaines de mégaoctets, voire de plusieurs dizaines de gigaoctets, peut rencontrer des problèmes de performance lorsqu'elle utilise les pages **4 KB**. De tels ensembles de données peuvent nécessiter un très grand nombre de pages **4 KB**, ce qui peut entraîner une surcharge du système d'exploitation et de l'unité centrale.

Cette section fournit des informations sur les pages énormes disponibles dans RHEL 9 et sur la manière dont vous pouvez les configurer.

39.1. CARACTÉRISTIQUES DE L'IMMENSE PAGE DISPONIBLE

Avec Red Hat Enterprise Linux 9, vous pouvez utiliser des pages volumineuses pour les applications qui travaillent avec des ensembles de données volumineux, et améliorer les performances de ces applications.

Vous trouverez ci-dessous les méthodes d'affichage des pages les plus importantes, qui sont prises en charge par RHEL 9 :

HugeTLB pages

Les pages HugeTLB sont également appelées pages énormes statiques. Il existe deux façons de réserver des pages HugeTLB :

- Au moment du démarrage : Cela augmente les chances de succès car la mémoire n'a pas encore été fragmentée de manière significative. Toutefois, sur les machines NUMA, le nombre de pages est automatiquement réparti entre les nœuds NUMA.

Pour plus d'informations sur les paramètres qui influencent le comportement des pages HugeTLB au démarrage, voir Paramètres de réservation des pages HugeTLB au démarrage et comment utiliser ces paramètres pour configurer les pages HugeTLB au démarrage, voir Configuration de HugeTLB au démarrage.

- Au moment de l'exécution : Elle permet de réserver les grandes pages par nœud NUMA. Si la réservation au moment de l'exécution est effectuée le plus tôt possible dans le processus de démarrage, la probabilité de fragmentation de la mémoire est plus faible.

Pour plus d'informations sur les paramètres qui influencent le comportement des pages HugeTLB au moment de l'exécution, voir Paramètres de réservation des pages HugeTLB au moment de l'exécution et comment utiliser ces paramètres pour configurer les pages HugeTLB au moment de l'exécution, voir Configuration de HugeTLB au moment de l'exécution .

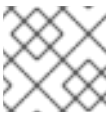
Transparent HugePages (THP)

Avec THP, le noyau attribue automatiquement des pages volumineuses aux processus, et il n'est donc pas nécessaire de réserver manuellement les pages volumineuses statiques. Les deux modes de fonctionnement de THP sont décrits ci-dessous :

- **system-wide**: Dans ce cas, le noyau tente d'attribuer des pages volumineuses à un processus lorsqu'il est possible d'allouer ces pages et que le processus utilise une grande zone de

mémoire virtuelle contiguë.

- **per-process**: Dans ce cas, le noyau n'attribue que d'énormes pages aux zones de mémoire des processus individuels que vous pouvez spécifier à l'aide de l'appel système **madvise**().



NOTE

La fonction THP ne prend en charge que les pages **2 MB**.

Pour plus d'informations sur les paramètres qui influencent le comportement de la page HugeTLB au démarrage, voir Activer les hugepages transparents et Désactiver les hugepages transparents .

39.2. PARAMÈTRES DE RÉSERVATION DES PAGES DE LA HUGETLB AU MOMENT DU DÉMARRAGE

Les paramètres suivants permettent d'influencer le comportement de la page HugeTLB au démarrage.

Pour plus d'informations sur l'utilisation de ces paramètres pour configurer les pages HugeTLB au démarrage, voir Configuration de HugeTLB au démarrage .

Tableau 39.1. Paramètres utilisés pour configurer les pages HugeTLB au moment du démarrage

Paramètres	Description	Valeur par défaut
hugepages	Définit le nombre d'énormes pages persistantes configurées dans le noyau au moment du démarrage. Dans un système NUMA, les pages volumineuses pour lesquelles ce paramètre est défini sont réparties de manière égale entre les nœuds. Vous pouvez affecter des pages volumineuses à des nœuds spécifiques au moment de l'exécution en modifiant la valeur des nœuds dans le fichier **/sys/devices/system/node/node_id/hugepages/hugepages-size/nr_hugepages**.	La valeur par défaut est **0**. Pour mettre à jour cette valeur au démarrage, modifiez la valeur de ce paramètre dans le fichier **/proc/sys/vm/nr_hugepages**.
hugepagesz	Définit la taille des grandes pages persistantes configurées dans le noyau au moment du démarrage.	Les valeurs valables sont **2 MB** et **1 GB**. La valeur par défaut est **2 MB**.
default_hugepagesz	Définit la taille par défaut des grandes pages persistantes configurées dans le noyau au moment du démarrage.	Les valeurs valables sont **2 MB** et **1 GB**. La valeur par défaut est **2 MB**.

39.3. CONFIGURATION DE HUGETLB AU DÉMARRAGE

La taille de page prise en charge par le sous-système HugeTLB dépend de l'architecture. L'architecture x86_64 prend en charge **2 MB** huge pages et **1 GB** gigantic pages.

Cette procédure décrit comment réserver une page **1 GB** au moment du démarrage.

Procédure

1. Pour créer un pool HugeTLB pour les pages **1 GB**, activez les options de noyau **default_hugepagesz=1G** et **hugepagesz=1G**:

```
# grubby --update-kernel=ALL --args="default_hugepagesz=1G hugepagesz=1G"
```

2. Créez un nouveau fichier appelé **hugetlb-gigantic-pages.service** dans le répertoire **/usr/lib/systemd/system/** et ajoutez le contenu suivant :

```
[Unit]
Description=HugeTLB Gigantic Pages Reservation
DefaultDependencies=no
Before=dev-hugepages.mount
ConditionPathExists=/sys/devices/system/node
ConditionKernelCommandLine=hugepagesz=1G

[Service]
Type=oneshot
RemainAfterExit=yes
ExecStart=/usr/lib/systemd/hugetlb-reserve-pages.sh

[Install]
WantedBy=sysinit.target
```

3. Créez un nouveau fichier appelé **hugetlb-reserve-pages.sh** dans le répertoire **/usr/lib/systemd/** et ajoutez le contenu suivant :
En ajoutant le contenu suivant, remplacez *number_of_pages* par le nombre de pages de 1GB que vous souhaitez réserver, et *node* par le nom du nœud sur lequel réserver ces pages.

```
#!/bin/sh

nodes_path=/sys/devices/system/node/
if [ ! -d $nodes_path ]; then
    echo "ERROR: $nodes_path does not exist"
    exit 1
fi

reserve_pages()
{
    echo $1 > $nodes_path/$2/hugepages/hugepages-1048576kB/nr_hugepages
}

reserve_pages number_of_pages node
```

Par exemple, pour réserver deux pages **1 GB** sur *node0* et une page de 1GB sur *node1*, remplacez *number_of_pages* par 2 pour *node0* et 1 pour *node1*:

```
-
```

```
reserve_pages 2 node0
reserve_pages 1 node1
```

4. Créer un script exécutable :

```
# chmod +x /usr/lib/systemd/hugetlb-reserve-pages.sh
```

5. Activer la réservation de démarrage anticipé :

```
# systemctl enable hugetlb-gigantic-pages
```

NOTE

- Vous pouvez essayer de réserver davantage de pages de 1 Go au moment de l'exécution en écrivant à tout moment sur **nr_hugepages**. Cependant, ces réservations peuvent échouer en raison de la fragmentation de la mémoire. La manière la plus fiable de réserver des pages **1 GB** est d'utiliser ce script **hugetlb-reserve-pages.sh**, qui s'exécute au début du démarrage.
- La réservation de pages volumineuses statiques peut effectivement réduire la quantité de mémoire disponible pour le système et l'empêcher d'utiliser correctement toute sa capacité de mémoire. Bien qu'un pool de pages volumineuses réservées correctement dimensionné puisse être bénéfique aux applications qui l'utilisent, un pool de pages volumineuses réservées surdimensionné ou inutilisé finira par nuire aux performances globales du système. Lorsque vous définissez un pool d'énormes pages réservées, assurez-vous que le système peut utiliser correctement toute sa capacité de mémoire.

Ressources supplémentaires

- **systemd.service(5)** page de manuel
- **/usr/share/doc/kernel-doc-kernel_version/Documentation/vm/hugetlbpage.txt** fichier

39.4. PARAMÈTRES DE RÉSERVATION DES PAGES HUGETLB AU MOMENT DE L'EXÉCUTION

Les paramètres suivants permettent d'influencer le comportement de la page HugeTLB au moment de l'exécution.

Pour plus d'informations sur l'utilisation de ces paramètres pour configurer les pages HugeTLB au moment de l'exécution, voir Configuration de HugeTLB au moment de l'exécution .

Tableau 39.2. Paramètres utilisés pour configurer les pages HugeTLB au moment de l'exécution

Paramètres	Description	Nom du fichier
nr_hugepages	Définit le nombre d'énormes pages d'une taille donnée attribuées à un nœud NUMA donné.	**/sys/devices/system/node/node_id/hugepages/hugepages-size/nr_hugepages**

Paramètres	Description	Nom du fichier
nr_overcommit_hugepages	Définit le nombre maximum d'énormes pages supplémentaires qui peuvent être créées et utilisées par le système par le biais d'un surengagement de la mémoire. L'écriture d'une valeur non nulle dans ce fichier indique que le système obtient ce nombre d'énormes pages à partir du pool de pages normal du noyau si le pool d'énormes pages persistant est épuisé. Au fur et à mesure que ces pages énormes excédentaires deviennent inutilisées, elles sont libérées et renvoyées dans le pool de pages normal du noyau.	**/proc/sys/vm/nr_overcommit_hugepages**

39.5. CONFIGURATION DE HUGETLB AU MOMENT DE L'EXÉCUTION

Cette procédure décrit comment ajouter *20 2048 kB* des pages énormes à *node2*.

Pour réserver des pages en fonction de vos besoins, remplacez :

- *20* avec le nombre d'énormes pages que vous souhaitez réserver,
- *2048kB* avec la taille des grandes pages,
- *node2* avec le nœud sur lequel vous souhaitez réserver les pages.

Procédure

1. Affiche les statistiques de la mémoire :

```
# numastat -cm | egrep 'Node|Huge'
                 Node 0 Node 1 Node 2 Node 3  Total add
AnonHugePages         0      2      0      8     10
HugePages_Total       0      0      0      0      0
HugePages_Free        0      0      0      0      0
HugePages_Surp        0      0      0      0      0
```

2. Ajoute au nœud le nombre d'énormes pages d'une taille spécifiée :

```
# echo 20 > /sys/devices/system/node/node2/hugepages/hugepages-2048kB/nr_hugepages
```

Verification steps

- Veiller à ce que le nombre d'énormes pages soit ajouté :

```
# numastat -cm | egrep 'Node|Huge'
                 Node 0 Node 1 Node 2 Node 3  Total
AnonHugePages         0      2      0      8     10
HugePages_Total       0      0     40      0     40
HugePages_Free        0      0     40      0     40
HugePages_Surp        0      0      0      0      0
```

Ressources supplémentaires

- **numastat(8)** page de manuel

39.6. PERMETTRE LA TRANSPARENCE DES IMAGES GÉANTES

THP est activé par défaut dans Red Hat Enterprise Linux 9. Cependant, vous pouvez activer ou désactiver THP.

Cette procédure décrit comment activer THP.

Procédure

1. Vérifier le statut actuel de THP :

```
# cat /sys/kernel/mm/transparent_hugepage/enabled
```

2. Activer le THP :

```
# echo always > /sys/kernel/mm/transparent_hugepage/enabled
```

3. Pour empêcher les applications d'allouer plus de ressources mémoire que nécessaire, désactivez les pages énormes transparentes à l'échelle du système et ne les activez que pour les applications qui en font explicitement la demande par l'intermédiaire de **madvise**:

```
# echo madvise > /sys/kernel/mm/transparent_hugepage/enabled
```

NOTE

Parfois, il est plus important d'assurer une faible latence pour les allocations à court terme que d'obtenir immédiatement les meilleures performances avec les allocations à long terme. Dans ce cas, vous pouvez désactiver le compactage direct tout en laissant le THP activé.

Le compactage direct est un compactage synchrone de la mémoire pendant l'allocation d'une grande page. La désactivation du compactage direct n'offre aucune garantie d'économie de mémoire, mais peut réduire le risque de latences plus élevées en cas d'erreurs de page fréquentes. Notez que si la charge de travail bénéficie de manière significative du THP, les performances diminuent. Désactiver le compactage direct :

```
# echo madvise > /sys/kernel/mm/transparent_hugepage/defrag
```

Ressources supplémentaires

- **madvise(2)** page de manuel

- Désactivation des pages de garde transparentes .

39.7. DÉSACTIVATION DES PAGES DE GARDE TRANSPARENTES

THP est activé par défaut dans Red Hat Enterprise Linux 9. Cependant, vous pouvez activer ou désactiver THP.

Cette procédure décrit comment désactiver THP.

Procédure

1. Vérifier le statut actuel de THP :

```
# cat /sys/kernel/mm/transparent_hugepage/enabled
```

2. Désactiver THP :

```
# echo never > /sys/kernel/mm/transparent_hugepage/enabled
```

39.8. IMPACT DE LA TAILLE DE LA PAGE SUR LA TAILLE DU TAMPON DE TRANSLATION (LOOKASIDE BUFFER)

La lecture des correspondances d'adresses à partir de la table des pages est longue et coûteuse en ressources, c'est pourquoi les processeurs sont dotés d'un cache pour les adresses récemment utilisées, appelé Translation Lookaside Buffer (TLB). Toutefois, le TLB par défaut ne peut mettre en cache qu'un certain nombre de correspondances d'adresses.

Si un mappage d'adresse demandé ne se trouve pas dans la TLB, ce qu'on appelle un oubli de la TLB, le système doit encore lire la table des pages pour déterminer le mappage de l'adresse physique à l'adresse virtuelle. En raison de la relation entre les besoins en mémoire des applications et la taille des pages utilisées pour mettre en cache les correspondances d'adresses, les applications nécessitant une grande quantité de mémoire sont plus susceptibles de subir une dégradation des performances due aux erreurs de la TLB que les applications nécessitant peu de mémoire. Il est donc important d'éviter les erreurs de la TLB dans la mesure du possible.

Les fonctions HugeTLB et Transparent Huge Page permettent aux applications d'utiliser des pages plus grandes que **4 KB**. Cela permet aux adresses stockées dans la TLB de référencer plus de mémoire, ce qui réduit le nombre d'erreurs de la TLB et améliore les performances des applications.

CHAPITRE 40. DÉMARRER AVEC SYSTEMTAP

En tant qu'administrateur système, vous pouvez utiliser SystemTap pour identifier les causes sous-jacentes d'un bogue ou d'un problème de performance sur un système Linux en cours d'exécution.

En tant que développeur d'applications, vous pouvez utiliser SystemTap pour surveiller en détail le comportement de votre application au sein du système Linux.

40.1. L'OBJECTIF DE SYSTEMTAP

SystemTap est un outil de traçage et de sondage que vous pouvez utiliser pour étudier et surveiller les activités de votre système d'exploitation (en particulier, le noyau) dans les moindres détails. SystemTap fournit des informations similaires à celles produites par des outils tels que **netstat**, **ps**, **top** et **iostat**. Toutefois, SystemTap offre davantage d'options de filtrage et d'analyse des informations collectées. Dans les scripts de SystemTap, vous spécifiez les informations que SystemTap recueille.

SystemTap vise à compléter la suite existante d'outils de surveillance Linux en fournissant aux utilisateurs l'infrastructure nécessaire pour suivre l'activité du noyau et en combinant cette capacité avec deux attributs :

Flexibility

le cadre SystemTap vous permet de développer des scripts simples pour étudier et surveiller une grande variété de fonctions du noyau, d'appels système et d'autres événements qui se produisent dans l'espace du noyau. Ainsi, SystemTap n'est pas tant un outil qu'un système qui vous permet de développer vos propres outils d'investigation et de surveillance spécifiques au noyau.

Ease-of-Use

SystemTap vous permet de surveiller l'activité du noyau sans avoir à recompiler le noyau ou à redémarrer le système.

40.2. INSTALLATION DE SYSTEMTAP

Pour commencer à utiliser SystemTap, installez les paquets requis. Pour utiliser SystemTap sur plus d'un noyau lorsqu'un système a plusieurs noyaux installés, installez les paquets requis correspondant à la version du noyau *each*.

Conditions préalables

- Vous avez activé les référentiels de débogage comme décrit dans la section Activation des référentiels de débogage et des référentiels de sources.

Procédure

1. Installez les paquets SystemTap requis :

```
# dnf install systemtap
```

2. Installez les paquets de noyau requis :

   a. Utilisation de **stap-prep**:

```
# stap-prep
```

   b. Si **stap-prep** ne fonctionne pas, installez manuellement les paquets de noyau requis :

```
# dnf install kernel-debuginfo-$(uname -r) kernel-debuginfo-common-$(uname -i)-
$(uname -r) kernel-devel-$(uname -r)
```

$(uname -i) est automatiquement remplacé par la plate-forme matérielle de votre système et **$(uname -r)** est automatiquement remplacé par la version de votre noyau.

Verification steps

- Si le noyau à sonder avec SystemTap est en cours d'utilisation, testez si votre installation a réussi :

```
# stap -v -e 'probe kernel.function("vfs_read") {printf("read performed\n"); exit()}'
```

Un déploiement réussi de SystemTap se traduit par un résultat similaire à celui qui suit :

```
Pass 1: parsed user script and 45 library script(s) in 340usr/0sys/358real ms.
Pass 2: analyzed script: 1 probe(s), 1 function(s), 0 embed(s), 0 global(s) in
290usr/260sys/568real ms.
Pass 3: translated to C into
"/tmp/stapiArgLX/stap_e5886fa50499994e6a87aacdc43cd392_399.c" in
490usr/430sys/938real ms.
Pass 4: compiled C into "stap_e5886fa50499994e6a87aacdc43cd392_399.ko" in
3310usr/430sys/3714real ms.
Pass 5: starting run. (1)
read performed (2)
Pass 5: run completed in 10usr/40sys/73real ms. (3)
```

Les trois dernières lignes de sortie (commençant par **Pass 5**) indiquent que :

1. SystemTap a créé avec succès l'instrumentation pour sonder le noyau et a exécuté l'instrumentation.
2. SystemTap a détecté l'événement spécifié (dans ce cas, une lecture VFS).
3. SystemTap a exécuté un gestionnaire valide (il a imprimé du texte et l'a fermé sans erreur).

40.3. PRIVILÈGES POUR EXÉCUTER SYSTEMTAP

L'exécution des scripts SystemTap nécessite des privilèges système élevés mais, dans certains cas, des utilisateurs non privilégiés peuvent avoir besoin d'exécuter l'instrumentation SystemTap sur leur machine.

Pour permettre aux utilisateurs d'exécuter SystemTap sans accès root, ajoutez des utilisateurs à **both** de ces groupes d'utilisateurs :

stapdev

Les membres de ce groupe peuvent utiliser **stap** pour exécuter des scripts SystemTap ou **staprun** pour exécuter des modules d'instrumentation SystemTap.
L'exécution de **stap** implique la compilation des scripts SystemTap en modules de noyau et leur chargement dans le noyau. Cela nécessite des privilèges élevés sur le système, qui sont accordés aux membres de **stapdev**. Malheureusement, ces privilèges accordent également un accès root effectif aux membres du groupe **stapdev**. Par conséquent, n'accordez l'appartenance au groupe **stapdev** qu'aux utilisateurs à qui l'on peut faire confiance pour l'accès à la racine.

stapusr

Les membres de ce groupe ne peuvent utiliser que **staprun** pour exécuter les modules d'instrumentation SystemTap. En outre, ils ne peuvent exécuter ces modules qu'à partir du répertoire **/lib/modules/*kernel_version*/systemtap/** . Ce répertoire doit appartenir uniquement à l'utilisateur root. Ce répertoire ne doit appartenir qu'à l'utilisateur root et ne doit être accessible en écriture qu'à l'utilisateur root.

40.4. EXÉCUTION DES SCRIPTS SYSTEMTAP

Les scripts SystemTap peuvent être exécutés à partir de l'entrée standard ou d'un fichier.

Les exemples de scripts distribués lors de l'installation de SystemTap se trouvent dans le répertoire **/usr/share/systemtap/examples**.

Conditions préalables

1. SystemTap et les paquets de noyau associés sont installés comme décrit dans la section Installation de Systemtap.

2. Pour exécuter des scripts SystemTap en tant qu'utilisateur normal, ajoutez l'utilisateur aux groupes SystemTap :

```
# usermod --append --groups
stapdev,stapusr user-name
```

Procédure

- Exécutez le script SystemTap :
  - A partir de l'entrée standard :

```
# echo "probe timer.s(1) {exit()}" | stap -
```

Cette commande demande à **stap** d'exécuter le script transmis par **echo** à l'entrée standard. Pour ajouter des options à **stap**, insérez-les avant le caractère **-**. Par exemple, pour rendre les résultats de cette commande plus verbeux, la commande est la suivante :

```
# echo "probe timer.s(1) {exit()}" | stap -v -
```

  - A partir d'un fichier :

```
# stap file_name
```

CHAPITRE 41. INSTRUMENTATION CROISÉE DE SYSTEMTAP

L'instrumentation croisée de SystemTap consiste à créer des modules d'instrumentation SystemTap à partir d'un script SystemTap sur un système pour les utiliser sur un autre système où SystemTap n'est pas entièrement déployé.

41.1. INSTRUMENTATION CROISÉE SYSTEMTAP

Lorsque vous exécutez un script SystemTap, un module noyau est construit à partir de ce script. SystemTap charge ensuite le module dans le noyau.

Normalement, les scripts SystemTap ne peuvent s'exécuter que sur les systèmes où SystemTap est déployé. Pour exécuter SystemTap sur dix systèmes, SystemTap doit être déployé sur tous ces systèmes. Dans certains cas, cela n'est ni possible ni souhaitable. Par exemple, la politique de l'entreprise peut vous interdire d'installer des paquets qui fournissent des compilateurs ou des informations de débogage sur des machines spécifiques, ce qui empêchera le déploiement de SystemTap.

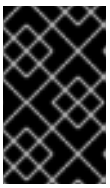
Pour contourner ce problème, utilisez *cross-instrumentation*. L'instrumentation croisée consiste à générer des modules d'instrumentation SystemTap à partir d'un script SystemTap sur un système afin de les utiliser sur un autre système. Ce processus offre les avantages suivants :

- Les paquets d'informations sur le noyau pour différentes machines peuvent être installés sur une seule machine hôte.

IMPORTANT

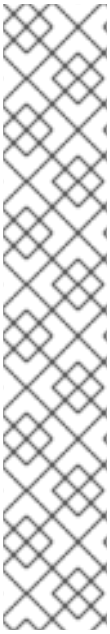
Des bogues dans l'emballage du noyau peuvent empêcher l'installation. Dans ce cas, les paquets **kernel-debuginfo** et **kernel-devel** pour *host system* et *target system* doivent correspondre. Si un bogue survient, signalez-le à https://bugzilla.redhat.com/.

- Chaque site *target machine* ne nécessite qu'une seule installation pour utiliser le module d'instrumentation SystemTap généré : **systemtap-runtime**.



IMPORTANT

Le site *host system* doit avoir la même architecture et utiliser la même distribution de Linux que le site *target system* pour que le site *instrumentation module* fonctionne.



TERMINOLOGIE

instrumentation module

Le module du noyau construit à partir d'un script SystemTap ; le module SystemTap est construit sur le site *host system*, et sera chargé sur le site *target kernel* du site *target system*.

host system

Le système sur lequel les modules d'instrumentation (issus des scripts SystemTap) sont compilés, pour être chargés sur *target systems*.

target system

Le système dans lequel le site *instrumentation module* est construit (à partir des scripts SystemTap).

target kernel

Le noyau de *target system* est le noyau qui charge et fait fonctionner *instrumentation module*.

41.2. INITIALISATION DE L'INSTRUMENTATION CROISÉE DE SYSTEMTAP

Initialiser l'instrumentation croisée de SystemTap pour construire des modules d'instrumentation SystemTap à partir d'un script SystemTap sur un système et les utiliser sur un autre système qui n'a pas SystemTap entièrement déployé.

Conditions préalables

- SystemTap est installé sur le site *host system* comme décrit dans la section Installation de Systemtap.
- Le paquet **systemtap-runtime** est installé sur chaque site *target system*:

```
# dnf install systemtap-runtime
```

- Les sites *host system* et *target system* ont tous deux la même architecture.
- Les sites *host system* et *target system* utilisent tous deux la même version majeure de Red Hat Enterprise Linux (telle que Red Hat Enterprise Linux 9).

IMPORTANT

Des bogues d'empaquetage du noyau peuvent empêcher l'installation de plusieurs paquets **kernel-debuginfo** et **kernel-devel** sur un même système. Dans ce cas, la version mineure des paquets *host system* et *target system* doit correspondre. Si un bogue survient, signalez-le à https://bugzilla.redhat.com/.

Procédure

1. Déterminez le noyau qui tourne sur chaque site *target system*:

```
$ uname -r
```

Répétez cette étape pour chaque *target system*.

2. Sur le site *host system*, installez les paquets *target kernel* et les paquets associés pour chaque *target system* en suivant la méthode décrite dans la section Installation de Systemtap.

3. Construire un module d'instrumentation sur le site *host system*, copier ce module sur le site *target system* et le faire fonctionner sur ce dernier :

   a. Utilisation de la mise en œuvre à distance :

   ```
   # stap --remote target_system script
   ```

   Cette commande met en œuvre à distance le script spécifié sur le site *target system*. Vous devez vous assurer qu'une connexion SSH peut être établie vers le site *target system* à partir du site *host system* pour que cette opération soit couronnée de succès.

   b. Manuellement :

      i. Construisez le module d'instrumentation sur le site *host system*:

      ```
      # stap -r kernel_version script -m module_name -p 4
      ```

      Ici, *kernel_version* fait référence à la version de *target kernel* déterminée à l'étape 1, *script* fait référence au script à convertir en *instrumentation module*, et *module_name* est le nom souhaité pour le *instrumentation module*. L'option **-p4** indique à SystemTap de ne pas charger et exécuter le module compilé.

      ii. Une fois que le site *instrumentation module* est compilé, copiez-le sur le système cible et chargez-le à l'aide de la commande suivante :

      ```
      # staprun module_name.ko
      ```

CHAPITRE 42. SURVEILLANCE DE L'ACTIVITÉ DU RÉSEAU AVEC SYSTEMTAP

Vous pouvez utiliser les exemples utiles de scripts SystemTap disponibles dans le répertoire **/usr/share/systemtap/testsuite/systemtap.examples/**, après avoir installé le paquetage **systemtap-testsuite**, pour surveiller et étudier l'activité réseau de votre système.

42.1. PROFILAGE DE L'ACTIVITÉ DU RÉSEAU AVEC SYSTEMTAP

Vous pouvez utiliser l'exemple de script SystemTap (**nettop.stp**) pour établir le profil de l'activité du réseau. Le script identifie les processus qui génèrent du trafic réseau sur le système et fournit les informations suivantes sur chaque processus :

PID

L'ID du processus répertorié.

UID

ID utilisateur. Un ID utilisateur de 0 correspond à l'utilisateur root.

DEV

Le périphérique Ethernet utilisé par le processus pour envoyer ou recevoir des données (par exemple, eth0, eth1).

XMIT_PK

Le nombre de paquets transmis par le processus.

RECV_PK

Nombre de paquets reçus par le processus.

XMIT_KB

Quantité de données envoyées par le processus, en kilo-octets.

RECV_KB

La quantité de données reçues par le service, en kilo-octets.

Conditions préalables

- Vous avez installé SystemTap comme décrit dans la section Installation de SystemTap.

Procédure

- Exécutez le script **nettop.stp**:

```
# stap  --example nettop.stp
```

Le script **nettop.stp** fournit un échantillonnage du profil du réseau toutes les 5 secondes.

La sortie du script **nettop.stp** ressemble à ce qui suit :

```
[...]
  PID   UID DEV     XMIT_PK RECV_PK XMIT_KB RECV_KB COMMAND
    0     0 eth0          0       5       0       0 swapper
11178     0 eth0          2       0       0       0 synergyc
  PID   UID DEV     XMIT_PK RECV_PK XMIT_KB RECV_KB COMMAND
 2886     4 eth0         79       0       5       0 cups-polld
```

```
11362     0 eth0         0      61       0       5 firefox
   0     0 eth0         3      32       0       3 swapper
 2886     4 lo           4       4       0       0 cups-polld
11178     0 eth0         3       0       0       0 synergyc
  PID   UID DEV     XMIT_PK RECV_PK XMIT_KB RECV_KB COMMAND
   0     0 eth0         0       6       0       0 swapper
 2886     4 lo           2       2       0       0 cups-polld
11178     0 eth0         3       0       0       0 synergyc
 3611     0 eth0         0       1       0       0 Xorg
  PID   UID DEV     XMIT_PK RECV_PK XMIT_KB RECV_KB COMMAND
   0     0 eth0         3      42       0       2 swapper
11178     0 eth0        43       1       3       0 synergyc
11362     0 eth0         0       7       0       0 firefox
 3897     0 eth0         0       1       0       0 multiload-apple
```

42.2. TRACER LES FONCTIONS APPELÉES DANS LE CODE D'UN SOCKET RÉSEAU AVEC SYSTEMTAP

Vous pouvez utiliser le script SystemTap de l'exemple **socket-trace.stp** pour suivre les fonctions appelées à partir du fichier net/socket.c du noyau. Cela vous permet d'identifier plus finement la manière dont chaque processus interagit avec le réseau au niveau du noyau.

Conditions préalables

- Vous avez installé SystemTap comme décrit dans la section Installation de SystemTap.

Procédure

- Exécutez le script **socket-trace.stp**:

```
# stap  --example socket-trace.stp
```

Un extrait de 3 secondes de la sortie du script **socket-trace.stp** ressemble à ce qui suit :

```
[...]
0 Xorg(3611): -> sock_poll
3 Xorg(3611): <- sock_poll
0 Xorg(3611): -> sock_poll
3 Xorg(3611): <- sock_poll
0 gnome-terminal(11106): -> sock_poll
5 gnome-terminal(11106): <- sock_poll
0 scim-bridge(3883): -> sock_poll
3 scim-bridge(3883): <- sock_poll
0 scim-bridge(3883): -> sys_socketcall
4 scim-bridge(3883):  -> sys_recv
8 scim-bridge(3883):   -> sys_recvfrom
12 scim-bridge(3883):-> sock_from_file
16 scim-bridge(3883):<- sock_from_file
20 scim-bridge(3883):-> sock_recvmsg
24 scim-bridge(3883):<- sock_recvmsg
28 scim-bridge(3883):   <- sys_recvfrom
31 scim-bridge(3883):  <- sys_recv
35 scim-bridge(3883): <- sys_socketcall
[...]
```

42.3. SURVEILLANCE DES CHUTES DE PAQUETS SUR LE RÉSEAU AVEC SYSTEMTAP

La pile réseau de Linux peut rejeter des paquets pour diverses raisons. Certains noyaux Linux incluent un point de traçage, **kernel.trace("kfree_skb")**`, qui permet de savoir où les paquets sont rejetés.

Le script **dropwatch.stp** SystemTap utilise **kernel.trace("kfree_skb")** pour tracer les rejets de paquets ; le script résume les emplacements qui rejettent des paquets à chaque intervalle de 5 secondes.

Conditions préalables

- Vous avez installé SystemTap comme décrit dans la section Installation de SystemTap.

Procédure

- Exécutez le script **dropwatch.stp**:

```
# stap  --example dropwatch.stp
```

L'exécution du script **dropwatch.stp** pendant 15 secondes produit un résultat similaire à celui qui suit :

```
Monitoring for dropped packets
51 packets dropped at location 0xffffffff8024cd0f
2 packets dropped at location 0xffffffff8044b472
51 packets dropped at location 0xffffffff8024cd0f
1 packets dropped at location 0xffffffff8044b472
97 packets dropped at location 0xffffffff8024cd0f
1 packets dropped at location 0xffffffff8044b472
Stopping dropped packet monitor
```

NOTE

Pour rendre l'emplacement des chutes de paquets plus significatif, consultez le fichier **/boot/System.map-$(uname -r)**. Ce fichier répertorie les adresses de départ de chaque fonction, ce qui vous permet de faire correspondre les adresses de la sortie du script **dropwatch.stp** à un nom de fonction spécifique. Dans l'extrait suivant du fichier **/boot/System.map-$(uname -r)**, l'adresse **0xffffffff8024cd0f** correspond à la fonction **unix_stream_recvmsg** et l'adresse **0xffffffff8044b472** correspond à la fonction **arp_rcv**:

```
[...]
ffffffff8024c5cd T unlock_new_inode
ffffffff8024c5da t unix_stream_sendmsg
ffffffff8024c920 t unix_stream_recvmsg
ffffffff8024cea1 t udp_v4_lookup_longway
[...]
ffffffff8044addc t arp_process
ffffffff8044b360 t arp_rcv
ffffffff8044b487 t parp_redo
ffffffff8044b48c t arp_solicit
[...]
```

CHAPITRE 43. PROFILER L'ACTIVITÉ DU NOYAU AVEC SYSTEMTAP

Vous pouvez établir le profil de l'activité du noyau en surveillant les appels de fonction à l'aide des scripts suivants.

43.1. COMPTER LES APPELS DE FONCTION AVEC SYSTEMTAP

Vous pouvez utiliser le script **functioncallcount.stp** SystemTap pour compter les appels de fonctions spécifiques du noyau. Vous pouvez également utiliser ce script pour cibler plusieurs fonctions du noyau.

Conditions préalables

- Vous avez installé SystemTap comme décrit dans la section Installation de Systemtap.

Procédure

- Exécutez le script **functioncallcount.stp**:

```
# stap --example functioncallcount.stp 'argument'
```

Ce script prend la fonction du noyau ciblée comme argument. Vous pouvez utiliser les caractères génériques de l'argument pour cibler plusieurs fonctions du noyau jusqu'à un certain point.

La sortie du script, par ordre alphabétique, contient les noms des fonctions appelées et le nombre de fois qu'elles ont été appelées pendant la durée de l'échantillon.

Prenons l'exemple suivant :

```
# stap -w -v --example functioncallcount.stp "*@mm*.c" -c /bin/true
```

où :

- -w : Supprime les avertissements.
- -v : Rend visible la sortie du noyau de départ.
- -c *command*: indique à SystemTap de compter les appels de fonctions pendant l'exécution d'une commande, dans cet exemple /**bin**/**true**.
  Le résultat devrait ressembler à ce qui suit :

```
[...]
__vma_link 97
__vma_link_file 66
__vma_link_list 97
__vma_link_rb 97
__xchg 103
add_page_to_active_list 102
add_page_to_inactive_list 19
add_to_page_cache 19
add_to_page_cache_lru 7
all_vm_events 6
```

```
alloc_pages_node 4630
alloc_slabmgmt 67
anon_vma_alloc 62
anon_vma_free 62
anon_vma_lock 66
anon_vma_prepare 98
anon_vma_unlink 97
anon_vma_unlock 66
arch_get_unmapped_area_topdown 94
arch_get_unmapped_exec_area 3
arch_unmap_area_topdown 97
atomic_add 2
atomic_add_negative 97
atomic_dec_and_test 5153
atomic_inc 470
atomic_inc_and_test 1
[...]
```

43.2. TRACER LES APPELS DE FONCTION AVEC SYSTEMTAP

Vous pouvez utiliser le script **para-callgraph.stp** SystemTap pour tracer les appels de fonction et les retours de fonction.

Conditions préalables

- Vous avez installé SystemTap comme décrit dans la section Installation de Systemtap.

Procédure

- Exécutez le script **para-callgraph.stp**.

```
# stap --example para-callgraph.stp 'argument1' 'argument2'
```

Le script **para-callgraph.stp** prend deux arguments en ligne de commande :

1. Le nom de la (des) fonction(s) dont vous souhaitez suivre l'entrée ou la sortie.
2. Une fonction de déclenchement optionnelle, qui active ou désactive le traçage pour chaque thread. Le suivi de chaque thread se poursuit tant que la fonction de déclenchement n'est pas terminée.

Prenons l'exemple suivant :

```
# stap -wv --example para-callgraph.stp 'kernel.function("*@fs/proc.c*")' 'kernel.function("vfs_read")' -c "cat /proc/sys/vm/* || true"
```

où :

- -w : Supprime les avertissements.
- -v : Rend visible la sortie du noyau de départ.
- -c *command*: indique à SystemTap de compter les appels de fonctions pendant l'exécution d'une commande, dans cet exemple /**bin**/**true**.

Le résultat devrait ressembler à ce qui suit :

```
[...]
   267 gnome-terminal(2921): <-do_sync_read return=0xfffffffffffffff5
   269 gnome-terminal(2921):<-vfs_read return=0xfffffffffffffff5
     0 gnome-terminal(2921):->fput file=0xffff880111eebbc0
     2 gnome-terminal(2921):<-fput
     0 gnome-terminal(2921):->fget_light fd=0x3 fput_needed=0xffff88010544df54
     3 gnome-terminal(2921):<-fget_light return=0xffff8801116ce980
     0 gnome-terminal(2921):->vfs_read file=0xffff8801116ce980 buf=0xc86504 count=0x1000
pos=0xffff88010544df48
     4 gnome-terminal(2921): ->rw_verify_area read_write=0x0 file=0xffff8801116ce980
ppos=0xffff88010544df48 count=0x1000
     7 gnome-terminal(2921): <-rw_verify_area return=0x1000
    12 gnome-terminal(2921): ->do_sync_read filp=0xffff8801116ce980 buf=0xc86504 len=0x1000
ppos=0xffff88010544df48
    15 gnome-terminal(2921): <-do_sync_read return=0xfffffffffffffff5
    18 gnome-terminal(2921):<-vfs_read return=0xfffffffffffffff5
     0 gnome-terminal(2921):->fput file=0xffff8801116ce980
```

43.3. DÉTERMINER LE TEMPS PASSÉ DANS LE NOYAU ET L'ESPACE UTILISATEUR AVEC SYSTEMTAP

Vous pouvez utiliser le script **thread-times.stp** SystemTap pour déterminer le temps passé par un thread donné dans le noyau ou dans l'espace utilisateur.

Conditions préalables

- Vous avez installé SystemTap comme décrit dans la section Installation de Systemtap.

Procédure

- Exécutez le script **thread-times.stp**:

  ```
  # stap --example thread-times.stp
  ```

  Ce script affiche les 20 processus les plus gourmands en temps CPU pendant une période de 5 secondes, ainsi que le nombre total de ticks CPU effectués pendant l'échantillon. La sortie de ce script indique également le pourcentage de temps CPU utilisé par chaque processus, et précise si ce temps a été passé dans l'espace noyau ou dans l'espace utilisateur.

  ```
  tid   %user %kernel (of 20002 ticks)
    0   0.00%  87.88%
  32169   5.24%   0.03%
  9815   3.33%   0.36%
  9859   0.95%   0.00%
  3611   0.56%   0.12%
  9861   0.62%   0.01%
  11106   0.37%   0.02%
  32167   0.08%   0.08%
  3897   0.01%   0.08%
  3800   0.03%   0.00%
  2886   0.02%   0.00%
  3243   0.00%   0.01%
  ```

```
3862   0.01%   0.00%
3782   0.00%   0.00%
21767   0.00%   0.00%
2522   0.00%   0.00%
3883   0.00%   0.00%
3775   0.00%   0.00%
3943   0.00%   0.00%
3873   0.00%   0.00%
```

43.4. SURVEILLANCE DES APPLICATIONS DE SONDAGE AVEC SYSTEMTAP

Vous pouvez utiliser le script SystemTap de **timeout.stp** pour identifier et surveiller les applications qui effectuent des interrogations. Cela vous permet de repérer les interrogations inutiles ou excessives, ce qui vous aide à identifier les domaines à améliorer en termes d'utilisation du processeur et d'économies d'énergie.

Conditions préalables

- Vous avez installé SystemTap comme décrit dans la section Installation de Systemtap.

Procédure

- Exécutez le script **timeout.stp**:

  ```
  # stap --example timeout.stp
  ```

  Ce script permet de savoir combien de fois chaque application utilise les appels système suivants au fil du temps :

- **poll**
- **select**
- **epoll**
- **itimer**
- **futex**
- **nanosleep**
- **signal**

Dans cet exemple, vous pouvez voir quel processus a utilisé quel appel système et combien de fois.

```
uid |   poll  select   epoll  itimer   futex nanosle  signal| process
28937 | 148793       0       0    4727   37288       0       0| firefox
22945 |      0   56949       0       1       0       0       0| scim-bridge
  0 |      0       0       0   36414       0       0       0| swapper
4275 |  23140       0       0       1       0       0       0| mixer_applet2
4191 |      0   14405       0       0       0       0       0| scim-launcher
22941 |   7908       1       0      62       0       0       0| gnome-terminal
4261 |      0       0       0       2       0    7622       0| escd
```

```
3695 |    0     0     0     0     0  7622     0| gdm-binary
3483 |    0  7206     0     0     0     0     0| dhcdbd
4189 | 6916     0     0     2     0     0     0| scim-panel-gtk
1863 | 5767     0     0     0     0     0     0| iscsid
```

43.5. SUIVI DES APPELS SYSTÈME LES PLUS FRÉQUEMMENT UTILISÉS AVEC SYSTEMTAP

Vous pouvez utiliser le script **topsys.stp** SystemTap pour dresser la liste des 20 principaux appels système utilisés par le système par intervalle de 5 secondes. Il indique également combien de fois chaque appel système a été utilisé au cours de cette période.

Conditions préalables

- Vous avez installé SystemTap comme décrit dans la section Installation de Systemtap.

Procédure

- Exécutez le script **topsys.stp**:

  ```
  # stap --example topsys.stp
  ```

  Prenons l'exemple suivant :

  ```
  # stap -v --example topsys.stp
  ```

  où -v rend visible la sortie du noyau de départ.

  Le résultat devrait ressembler à ce qui suit :

```
----------------------------------------------------------------
                SYSCALL      COUNT
           gettimeofday       1857
                   read       1821
                  ioctl       1568
                   poll       1033
                  close        638
                   open        503
                 select        455
                  write        391
                 writev        335
                  futex        303
                recvmsg        251
                 socket        137
          clock_gettime        124
         rt_sigprocmask        121
                 sendto        120
              setitimer        106
                   stat         90
                   time         81
              sigreturn         72
                  fstat         66
----------------------------------------------------------------
```

43.6. SUIVI DU VOLUME D'APPELS SYSTÈME PAR PROCESSUS AVEC SYSTEMTAP

Vous pouvez utiliser le script **syscalls_by_proc.stp** SystemTap pour voir quels sont les processus qui effectuent le plus grand nombre d'appels système. Il affiche les 20 processus qui effectuent le plus grand nombre d'appels système.

Conditions préalables

- Vous avez installé SystemTap comme décrit dans la section Installation de Systemtap.

Procédure

- Exécutez le script **syscalls_by_proc.stp**:

```
# stap --example syscalls_by_proc.stp
```

La sortie du script **syscalls_by_proc.stp** ressemble à ce qui suit :

```
Collecting data... Type Ctrl-C to exit and display results
#SysCalls  Process Name
1577       multiload-apple
692        synergyc
408        pcscd
376        mixer_applet2
299        gnome-terminal
293        Xorg
206        scim-panel-gtk
95         gnome-power-man
90         artsd
85         dhcdbd
84         scim-bridge
78         gnome-screensav
66         scim-launcher
[...]
```

CHAPITRE 44. SURVEILLANCE DE L'ACTIVITÉ DES DISQUES ET DES E/S AVEC SYSTEMTAP

Vous pouvez surveiller l'activité des disques et des E/S à l'aide des scripts suivants.

44.1. SYNTHÈSE DU TRAFIC DE LECTURE/ÉCRITURE DES DISQUES AVEC SYSTEMTAP

Vous pouvez utiliser le script **disktop.stp** SystemTap pour identifier les processus qui effectuent les lectures et les écritures les plus lourdes sur le disque du système.

Conditions préalables

- Vous avez installé SystemTap comme décrit dans la section Installation de Systemtap.

Procédure

- Exécutez le script **disktop.stp**:

```
# stap --example disktop.stp
```

Le script affiche les dix principaux processus responsables des lectures ou écritures les plus lourdes sur un disque.

La sortie comprend les données suivantes pour chaque processus répertorié :

UID
Identifiant de l'utilisateur. Un identifiant d'utilisateur de **0** correspond à l'utilisateur racine.

PID
L'ID du processus répertorié.

PPID
L'ID du processus parent du processus listé.

CMD
Le nom du processus répertorié.

DISPOSITIF
Le périphérique de stockage à partir duquel le processus répertorié lit ou écrit.

T
Le type d'action effectuée par le processus répertorié, où **W** fait référence à l'écriture et **R** à la lecture.

BYTES
Quantité de données lues ou écrites sur le disque.

La sortie du script **disktop.stp** ressemble à ce qui suit :

```
[...]
Mon Sep 29 03:38:28 2008 , Average:  19Kb/sec, Read: 7Kb, Write: 89Kb
UID     PID     PPID                CMD   DEVICE    T    BYTES
0    26319    26294               firefox     sda5    W       90229
0     2758     2757         pam_timestamp_c     sda5    R        8064
```

```
0    2885      1                cupsd    sda5   W       1678
Mon Sep 29 03:38:38 2008 , Average:   1Kb/sec, Read: 7Kb, Write: 1Kb
UID     PID    PPID                CMD   DEVICE    T    BYTES
0    2758    2757        pam_timestamp_c    sda5   R       8064
0    2885      1                cupsd    sda5   W       1678
```

44.2. SUIVI DU TEMPS D'E/S POUR CHAQUE LECTURE OU ÉCRITURE DE FICHIER AVEC SYSTEMTAP

Vous pouvez utiliser le script **iotime.stp** SystemTap pour surveiller le temps nécessaire à chaque processus pour lire ou écrire dans un fichier. Cela vous aide à déterminer quels fichiers sont lents à charger sur un système.

Conditions préalables

- Vous avez installé SystemTap comme décrit dans la section Installation de Systemtap.

Procédure

- Exécutez le script **iotime.stp**:

  ```
  # stap --example iotime.stp
  ```

  Le script suit chaque fois qu'un appel système ouvre, ferme, lit et écrit dans un fichier. Pour chaque fichier auquel un appel système accède, il compte le nombre de microsecondes nécessaires à la fin des lectures ou des écritures et enregistre la quantité de données, en octets, lues ou écrites dans le fichier.

  La sortie contient :

- Un horodatage, en microsecondes
- ID et nom du processus
- Un drapeau **access** ou **iotime**
- Le fichier consulté
  Si un processus a pu lire ou écrire des données, une paire de lignes access et **iotime** doit apparaître ensemble. La ligne d'accès fait référence à l'heure à laquelle un processus donné a commencé à accéder à un fichier. La fin de la ligne access indique la quantité de données lues ou écrites. La ligne **iotime** indique le temps, en microsecondes, que le processus a mis pour effectuer la lecture ou l'écriture.

La sortie du script **iotime.stp** ressemble à ce qui suit :

```
[...]
825946 3364 (NetworkManager) access /sys/class/net/eth0/carrier read: 8190 write: 0
825955 3364 (NetworkManager) iotime /sys/class/net/eth0/carrier time: 9
[...]
117061 2460 (pcscd) access /dev/bus/usb/003/001 read: 43 write: 0
117065 2460 (pcscd) iotime /dev/bus/usb/003/001 time: 7
[...]
```

```
3973737 2886 (sendmail) access /proc/loadavg read: 4096 write: 0
3973744 2886 (sendmail) iotime /proc/loadavg time: 11
[...]
```

44.3. SUIVI DES E/S CUMULATIVES AVEC SYSTEMTAP

Vous pouvez utiliser le script **traceio.stp** SystemTap pour suivre la quantité cumulée d'E/S sur le système.

Conditions préalables

- Vous avez installé SystemTap comme décrit dans la section Installation de Systemtap.

Procédure

- Exécutez le script **traceio.stp**:

  ```
  # stap --example traceio.stp
  ```

  Le script affiche les dix principaux exécutables générant du trafic d'E/S au fil du temps. Il indique également la quantité cumulée de lectures et d'écritures d'E/S effectuées par ces exécutables. Ces informations sont suivies et imprimées par intervalles d'une seconde et par ordre décroissant.

  La sortie du script **traceio.stp** ressemble à ce qui suit :

```
[...]
          Xorg r:   583401 KiB w:        0 KiB
      floaters r:       96 KiB w:     7130 KiB
multiload-apple r:      538 KiB w:      537 KiB
          sshd r:       71 KiB w:       72 KiB
pam_timestamp_c r:      138 KiB w:        0 KiB
       staprun r:       51 KiB w:       51 KiB
         snmpd r:       46 KiB w:        0 KiB
         pcscd r:       28 KiB w:        0 KiB
    irqbalance r:       27 KiB w:        4 KiB
         cupsd r:        4 KiB w:       18 KiB
          Xorg r:   588140 KiB w:        0 KiB
      floaters r:       97 KiB w:     7143 KiB
multiload-apple r:      543 KiB w:      542 KiB
          sshd r:       72 KiB w:       72 KiB
pam_timestamp_c r:      138 KiB w:        0 KiB
       staprun r:       51 KiB w:       51 KiB
         snmpd r:       46 KiB w:        0 KiB
         pcscd r:       28 KiB w:        0 KiB
    irqbalance r:       27 KiB w:        4 KiB
         cupsd r:        4 KiB w:       18 KiB
```

44.4. SURVEILLANCE DE L'ACTIVITÉ E/S SUR UN APPAREIL SPÉCIFIQUE AVEC SYSTEMTAP

Vous pouvez utiliser le script **traceio2.stp** SystemTap pour surveiller l'activité d'E/S sur un périphérique spécifique.

Conditions préalables

- Vous avez installé SystemTap comme décrit dans la section Installation de Systemtap.

Procédure

- Exécutez le script **traceio2.stp**.

```
# stap --example traceio2.stp 'argument'
```

Ce script prend en argument le numéro complet de l'appareil. Pour trouver ce numéro, vous pouvez utiliser :

```
# stat -c "0x " directory
```

Où se trouve *directory* sur l'appareil que vous souhaitez surveiller.

La sortie contient les éléments suivants :

- Le nom et l'ID de tout processus effectuant une lecture ou une écriture
- La fonction qu'il remplit (**vfs_read** ou **vfs_write**)
- Le numéro de périphérique du noyau

Considérons la sortie suivante de **# stap traceio2.stp 0x805**

```
[...]
synergyc(3722) vfs_read 0x800005
synergyc(3722) vfs_read 0x800005
cupsd(2889) vfs_write 0x800005
cupsd(2889) vfs_write 0x800005
cupsd(2889) vfs_write 0x800005
[...]
```

44.5. SURVEILLANCE DES LECTURES ET DES ÉCRITURES DANS UN FICHIER AVEC SYSTEMTAP

Vous pouvez utiliser le script SystemTap de **inodewatch.stp** pour surveiller les lectures et les écritures dans un fichier en temps réel.

Conditions préalables

- Vous avez installé SystemTap comme décrit dans la section Installation de Systemtap.

Procédure

- Exécutez le script **inodewatch.stp**.

```
# stap --example inodewatch.stp 'argument1' 'argument2' 'argument3'
```

Le script **inodewatch.stp** prend trois arguments en ligne de commande :

1. Numéro de l'appareil principal du fichier.
2. Le numéro de périphérique mineur du fichier.
3. Le numéro d'inode du fichier.

Vous pouvez obtenir ces chiffres en utilisant :

```
# stat -c ' %i'filename
```

Où *filename* est un chemin absolu.

Prenons l'exemple suivant :

```
# stat -c '%D %i' /etc/crontab
```

Le résultat devrait ressembler à ceci :

```
805 1078319
```

où :

- **805** est le numéro de l'appareil en base 16 (hexadécimal). Les deux derniers chiffres correspondent au numéro mineur de l'appareil et les autres au numéro majeur.
- **1078319** est le numéro d'inode.

Pour commencer à surveiller **/etc/crontab**, exécutez :

```
# stap inodewatch.stp 0x8 0x05 1078319
```

Dans les deux premiers arguments, vous devez utiliser les préfixes 0x pour les nombres en base 16.

La sortie contient les éléments suivants :

- Le nom et l'ID de tout processus effectuant une lecture ou une écriture
- La fonction qu'il remplit (**vfs_read** ou **vfs_write**)
- Le numéro de périphérique du noyau

La sortie de cet exemple devrait ressembler à ce qui suit :

```
cat(16437) vfs_read 0x800005/1078319
cat(16437) vfs_read 0x800005/1078319
```