Red Hat Directory Server 12

Red Hat Directory Server의 성능 튜닝

서버 및 데이터베이스 성능 개선

Last Updated: 2024-08-25

Red Hat Directory Server 12 Red Hat Directory Server의 성능 튜닝

서버 및 데이터베이스 성능 개선

법적 공지

초록

Directory Server 성능을 개선하기 위해 잠금, 리소스 제한 및 트랜잭션 로그의 수를 조정할 수 있습니다. 시스템에서 사용 가능한 디스크 공간이 충분하지 않은 경우 로컬 디스크를 모니터링하고 Directory Server를 종료할 수도 있습니다.

차례

RED HAT DIRECTORY SERVER에 대한 피드백 제공

Red Hat의 문서 및 제품에 대한 의견을 제공해 주셔서 감사합니다. Red Hat이 어떻게 이를 개선할 수 있는지 알려 주십시오. 이렇게 하려면 다음을 수행합니다.

- Jira (계정 필요)를 통해 Red Hat Directory Server 설명서에 피드백을 제출하려면 다음을 수행합니다.
  1. Red Hat 문제 추적기 로 이동하십시오.
  2. **요약** 필드에 설명 제목을 입력합니다.
  3. **설명** 필드에 개선을 위한 제안을 입력합니다. 문서의 관련 부분에 대한 링크를 포함합니다.
  4. 대화 상자 하단에서 **생성** 을 클릭합니다.
- Jira를 통해 Red Hat Directory Server 제품에 대한 피드백을 제출하기 위해 필요한 경우:
  1. Red Hat 문제 추적기 로 이동하십시오.
  2. **문제 생성** 페이지에서 **다음을** 클릭합니다.
  3. **Summary** 필드를 입력합니다.
  4. **Component** 필드에서 구성 요소를 선택합니다.
  5. 다음을 포함하여 **Description** 필드를 작성합니다.
     a. 선택한 구성 요소의 버전 번호입니다.
     b. 문제 또는 개선을 위한 제안을 재현하는 단계입니다.
  6. **생성**을 클릭합니다.

1장. 데이터베이스 활동 모니터링

관리자는 캐시와 같은 튜닝 설정이 올바르게 구성되었는지 확인하기 위해 데이터베이스 활동을 모니터링해야 합니다.

1.1. 명령줄을 사용하여 데이터베이스 활동 모니터링

명령줄을 사용하여 모니터링 활동을 표시하려면 **cn=monitor,cn=*database_name*,cn=ldbm 데이터베이스,cn=plugins,cn=config** 에 저장된 동적으로 업데이트된 읽기 전용 속성을 표시합니다.

절차

- 데이터베이스의 현재 활동을 표시하려면 다음을 입력합니다.

  ```
  # dsconf -D "cn=Directory Manager" ldap://server.example.com monitor backend userRoot
  ```

  이 명령은 **사용자Root** 데이터베이스의 활동을 표시합니다.

추가 리소스

- 데이터베이스 모니터링 속성

1.2. 웹 콘솔을 사용하여 데이터베이스 활동 모니터링

웹 콘솔에서 디렉터리 서버는 **cn=monitor,cn=*database_name*,cn=ldbm 데이터베이스,cn=plugins,cn=plugins,cn=config** 의 읽기 전용 모니터링 속성의 값을 'Monitoring' 탭에 표시합니다.

절차

1. **모니터링 → *데이터베이스 이름으로*** 이동합니다.
2. **Entry Cache** 및 **DN Cache** 탭의 캐시 값을 표시합니다.

추가 리소스

- 데이터베이스 모니터링 속성

1.3. 데이터베이스 모니터링 속성

표 1.1. 상속 설정

속성	설명
readonly	데이터베이스가 읽기 전용 모드(**1**)인지 또는 읽기 쓰기 모드(**0**)인지 여부를 나타냅니다.
entrycachehits	성공한 총 항목 캐시 조회 수입니다. 값은 데이터베이스에서 다시 로드하지 않고 서버가 항목 캐시에서 항목을 검색할 수 있는 총 횟수입니다.

속성	설명
entrycachetries	인스턴스를 시작한 이후 총 항목 캐시 조회 수입니다. 이 값은 인스턴스가 시작된 이후 총 수이며 Directory Server는 항목 캐시에서 항목을 검색하려고 했습니다.
entrycachehitratio	항목 캐시 수가 성공적으로 항목 캐시 조회를 시도합니다. 이 수는 인스턴스를 마지막으로 시작한 이후 총 조회 및 조회를 기반으로 합니다. 항목 캐시 적중 비율이 100%에 가까울수록 좋습니다. 작업이 항목 캐시에 없는 항목을 검색하려고 할 때마다 서버는 해당 항목을 얻기 위해 데이터베이스에 액세스해야 합니다. 따라서 이 비율이 0으로 떨어지면 디스크 액세스 수가 증가하고 디렉터리 검색 성능이 저하됩니다. 이 비율을 개선하려면 데이터베이스의 항목 캐시 크기를 늘립니다. 이 비율을 개선하려면 **cn=*database_name*,cn=ldbm 데이터베이스,cn=plugins,cn=config** 항목에서 **nsslapd-cachememsize** 속성 값을 늘려 항목 캐시의 크기를 늘립니다.
currententrycachesize	현재 항목 캐시에 있는 디렉터리 항목의 총 크기(바이트)입니다. 캐시에 존재할 수 있는 항목의 크기를 늘리려면 **cn=*database_name*,cn=ldbm 데이터베이스,cn=plugins,cn=config** 항목에서 **nsslapd-cachememsize** 속성 값을 늘립니다.
maxentrycachesize	Directory Server가 항목 캐시에서 유지 관리할 수 있는 디렉터리 항목의 최대 크기(바이트)입니다. 캐시에 존재할 수 있는 항목의 크기를 늘리려면 **cn=*database_name*,cn=ldbm 데이터베이스,cn=plugins,cn=config** 항목에서 **nsslapd-cachememsize** 속성 값을 늘립니다.
currententrycachecount	지정된 백엔드의 항목 캐시에 저장된 현재 항목 수입니다.
maxentrycachecount	데이터베이스의 항목 캐시에 저장된 최대 항목 수입니다. 이 값을 조정하려면 **cn=*database_name*,cn=ldbm 데이터베이스,cn=plugins,cn=config** 항목의 **nsslapd-cachesize** 속성 값을 늘립니다.
dncachehits	서버가 다시 정규화하지 않고 DN 캐시에서 정규화된 고유 이름(DN)을 가져와서 요청을 처리할 수 있는 횟수입니다.
dncachetries	인스턴스를 시작한 이후 총 DN 캐시 액세스 수입니다.
dncachehitratio	캐시 비율은 DN 캐시 적중을 성공적으로 수행하려고 합니다. 이 값이 100%에 가까울수록 좋습니다.

속성	설명
currentdncachesize	현재 DN 캐시에 있는 DN의 총 크기(바이트)입니다. DN 캐시에 존재할 수 있는 항목의 크기를 늘리려면 **cn=*database_name*,cn=ldbm 데이터베이스,cn=plugins,cn=config** 항목에서 **nsslapd-dncachememsize** 속성 값을 늘립니다.
maxdncachesize	Directory Server에서 DN 캐시에서 유지할 수 있는 DN의 최대 크기(바이트)입니다. 캐시에 존재할 수 있는 항목의 크기를 늘리려면 **cn=*database_name*,cn=ldbm 데이터베이스,cn=plugins,cn=config** 항목에서 **nsslapd-dncachememsize** 속성 값을 늘립니다.
currentdncachecount	DN 캐시에 현재 존재하는 DN 수입니다.
maxdncachecount	DN 캐시에 허용되는 최대 DN 수입니다.

2장. 뷰의 성능 개선

뷰 기반 계층 구조의 성능은 계층 자체의 구성과 디렉터리 트리(DIT)의 항목 수에 따라 달라집니다.

가상 DIT 보기를 사용하는 경우 일반적으로 성능에 약간의 변경 사항이 있을 수 있습니다(표준 DIT에서 동일한 검색의 일부 백분율로 인해). 검색에서 뷰를 호출하지 않으면 성능에 영향을 미치지 않습니다. 배포 전에 예상되는 검색 패턴 및 로드에 대해 가상 DIT 보기를 테스트합니다.

뷰를 조직의 범용 탐색 도구로 사용하려는 경우 뷰 필터에 사용되는 속성을 인덱싱하는 것이 좋습니다.

또한 뷰에서 하위 필터 평가에 사용할 VLV(가상 목록 보기) 인덱스를 구성할 수 있습니다.

뷰를 위해 특별히 디렉터리의 다른 부분을 튜닝할 필요가 없습니다.

2.1. 명령줄을 사용하여 뷰의 성능을 개선하는 인덱스 생성

뷰는 지정된 필터를 기반으로 검색 결과에서 파생됩니다. 필터의 일부는 **nsViewFilter** 에서 명시적으로 제공되는 속성입니다. 나머지 필터는 보기에 포함된 실제 항목의 **entryid** 및 **parentid** 작동 속성을 찾고 있는 항목 계층 구조를 기반으로 합니다.

```
(|(parentid=search_base_id)(entryid=search_base_id)
```

검색된 특성(**entryid**,**parentid** 또는 **nsViewFilter** 의 속성)이 인덱싱되지 않은 경우 검색은 부분적으로 인덱싱되지 않고 Directory Server는 전체 디렉터리 트리를 검색하여 일치하는 항목을 검색합니다.

뷰 성능을 향상하려면 다음과 같이 인덱스를 만듭니다.To improve views performance, create the indexes as follows:

- **entryid** 에 대한 *같음 인덱스* (**eq**)를 만듭니다. **parentid** 속성은 기본적으로 시스템 인덱스에서 인덱싱됩니다.
- **nsViewFilter** 테스트의 필터가 존재하는 경우(**attribute=***) 테스트 중인 속성에 대한 *존재 인덱스* (**이전**)를 만듭니다. 디렉터리 항목의 소수성에 표시되는 속성에서만 이 인덱스 유형을 사용해야 합니다.
- **nsViewFilter** 의 필터가 같음(**attribute=value**)을 테스트 중인 경우 테스트 중인 속성에 대한 *같음 인덱스* (**eq**)를 만듭니다.
- **nsViewFilter** 의 필터가 하위 문자열(**attribute=value***)을 테스트하면 테스트 중인 속성에 대한 **하위 *문자열 인덱스* (하위**)를 만듭니다.
- **nsViewFilter** 의 필터가 approximation(**attribute~=value**)을 테스트 중인 경우 테스트 중인 속성에 대해 **대략적인** *인덱스* (대량)를 만듭니다.

예를 들어 다음 보기 필터를 사용하는 경우 다음을 수행합니다.

```
nsViewFilter: (&(objectClass=inetOrgPerson)(roomNumber=*66))
```

기본적으로 수행되는 *같음 인덱스* 인 **objectClass** 와 하위 문자열 *인덱스가 있는* ***roomNumber*** *가 있어야* 합니다.

사전 요구 사항

- view 필터에서 사용하는 속성을 알고 있습니다.

절차

1. 선택 사항: 백엔드를 나열하여 인덱스할 데이터베이스를 결정합니다.

```
# dsconf -D "cn=Directory Manager" instance_name backend suffix list
dc=example,dc=com (userroot)
```

   선택한 데이터베이스 이름(가져오기)을 기록해 둡니다.

2. 선택한 백엔드 데이터베이스에 대해 **dsconfig** 유틸리티를 사용하여 인덱스 구성을 생성합니다. 특히 국제화된 인스턴스의 경우 특성 이름, 인덱스 유형, 선택적으로 일치하는 규칙을 지정하여 데이터 정렬 순서(OID)를 설정합니다.

```
# dsconf -D "cn=Directory Manager" instance_name backend index add --attr
roomNumber --index-type sub userroot
```

   view 필터에 사용된 각 속성에 대해 이 단계를 반복합니다.

3. 새 인덱스를 적용하려면 데이터베이스를 다시 인덱싱합니다.

```
# dsconf -D "cn=Directory Manager" instance_name backend index reindex userroot
```

검증

1. 뷰에서 사용하는 것과 동일한 필터가 있는 표준 디렉터리 트리를 기반으로 하는 검색을 수행합니다.

```
# ldapsearch -D "cn=Directory Manager" -W -H ldap://server.example.com -x -b
dc=example,dc=com (&(objectClass=inetOrgPerson)(roomNumber=*66))
# ldapsearch -D "cn=Directory Manager" -W -H ldap://server.example.com -x -b
dc=example,dc=com "(&(objectClass=inetOrgPerson)(roomNumber=*66))"
```

2. **/var/log/dirsrv/slapd-***instance_name***/ access 에서 액세스 로그** 를 확인합니다. 검색 **RESULT** 에는 세부 정보에 **note=U** 또는 **Partially Unindexed 필터** 가 포함되어 있지 않아야 합니다.

추가 리소스

- 인덱스 관리

2.2. 웹 콘솔을 사용하여 뷰의 성능을 향상시키는 인덱스 생성

뷰는 지정된 필터를 기반으로 검색 결과에서 파생됩니다. 필터의 일부는 **nsViewFilter** 에서 명시적으로 제공되는 속성입니다. 나머지 필터는 보기에 포함된 실제 항목의 **entryid** 및 **parentid** 작동 속성을 찾고 있는 항목 계층 구조를 기반으로 합니다.

```
(|(parentid=search_base_id)(entryid=search_base_id)
```

검색된 특성(**entryid**,**parentid** 또는 **nsViewFilter** 의 속성)이 인덱싱되지 않은 경우 검색은 부분적으로 인덱싱되지 않고 Directory Server는 전체 디렉터리 트리를 검색하여 일치하는 항목을 검색합니다.

뷰 성능을 향상하려면 다음과 같이 인덱스를 만듭니다.To improve views performance, create the indexes as follows:

- **entryid** 에 대한 *같음 인덱스* (**eq**)를 만듭니다. **parentid** 속성은 기본적으로 시스템 인덱스에서 인덱싱됩니다.
- **nsViewFilter** 테스트의 필터가 존재하는 경우(**attribute=***) 테스트 중인 속성에 대한 *존재 인덱스* (**이전**)를 만듭니다. 디렉터리 항목의 소수성에 표시되는 속성에서만 이 인덱스 유형을 사용해야 합니다.
- **nsViewFilter** 의 필터가 같음(**attribute=value**)을 테스트 중인 경우 테스트 중인 속성에 대한 *같음 인덱스* (**eq**)를 만듭니다.
- **nsViewFilter** 의 필터가 하위 문자열(**attribute=value***)을 테스트하면 테스트 중인 속성에 대한 **하위 *문자열 인덱스*** (**하위**)를 만듭니다.
- **nsViewFilter** 의 필터가 approximation(**attribute~=value**)을 테스트 중인 경우 테스트 중인 속성에 대해**대략적인** *인덱스* (대량)를 만듭니다.

예를 들어 다음 보기 필터를 사용하는 경우 다음을 수행합니다.

```
nsViewFilter: (&(objectClass=inetOrgPerson)(roomNumber=*66))
```

기본적으로 수행되는 *같음 인덱스* 인 **objectClass** 와 하위 문자열 *인덱스가 있는* ***roomNumber*** *가 있어야* 합니다.

사전 요구 사항

- 웹 콘솔에서 인스턴스에 로그인되어 있습니다.
- view 필터에서 사용하는 속성을 알고 있습니다.

절차

1. **Database** 아래에서 인덱스를 생성할 구성 트리에서 접미사를 선택합니다.
2. **인덱스 및 데이터베이스 인덱스로 이동합니다**.
3. **인덱스 추가** 버튼을 클릭합니다.
4. 특성 이름을 입력하고 특성을 선택합니다.
5. 이 특성에 대해 생성해야 하는 **인덱스 유형을** 선택합니다.
6. 선택적으로 **일치하는 규칙을** 추가하여 특히 국제화된 인스턴스의 경우 데이터 정렬 순서(OID)를 지정합니다.
7. **생성 후 인덱스** 를 선택하여 나중에 인덱스를 다시 작성합니다.
8. **인덱스 생성을** 클릭합니다.
9. 인덱싱할 각 속성에 대해 단계를 반복합니다.

검증

- 추가된 속성의 이름을 입력하여 **인덱스를 필터링** 합니다.
- 새로 인덱싱된 특성이 결과에 표시되어야 합니다.

추가 리소스

- 인덱스 관리

3장. 긴 ID 목록을 로드할 때 성능을 개선하도록 인덱스 검사 제한 설정

대규모 디렉터리에서 검색 결과 목록은 대규모일 수 있습니다. 예를 들어 **inetorgperson** 속성이 있는 100만 개의 항목이 있는 디렉터리는 **(objectclass=inetorg person)** 과 같은 필터를 사용하여 검색에서 이러한 모든 항목을 반환합니다.

데이터베이스에서 긴 ID 목록을 로드하면 검색 성능이 크게 저하됩니다. ID 목록 검사 제한은 키가 전체 기본 인덱스와 일치하도록 간주되기 전에 ID Directory Server 읽기 수에 대한 제한을 설정합니다. 즉, Directory Server는 검색을 다른 리소스 제한 집합을 사용하여 인덱싱되지 않은 검색으로 처리합니다.

큰 인덱스의 경우 인덱스와 일치하는 모든 검색을 인덱싱되지 않은 검색으로 처리하는 것이 더 효율적입니다.For large indexes, it is actually more efficient to treat any search which matches the index as an unindexed search. 검색 작업은 거의 디렉터리의 크기와 디렉터리 자체인 인덱스를 검색하지 않고 결과를 처리하기 위해 한 곳에서만 확인해야 합니다.

전역적으로 또는 특정 데이터베이스에 대해 인덱스 검사 제한을 설정할 수 있습니다.

3.1. 명령줄을 사용하여 글로벌 인덱스 검사 제한 설정

기본적으로 Directory Server의 ID 목록 검사 제한은 **4000** 입니다. 대부분의 시나리오에서 이 값은 일반적인 데이터베이스 크기 및 액세스 패턴에 적합한 성능을 제공하며 기본값을 변경할 필요가 없습니다. 데이터베이스 인덱스가 4000 항목보다 약간 크지만 전체 디렉토리보다 훨씬 작으면 ID 목록 검사 제한을 늘리면 검색을 개선할 수 있습니다.

반면 제한을 낮추면 4000개의 항목 제한에 도달할 수 있는 검색 속도를 크게 높일 수 있지만 모든 항목을 스캔할 필요는 없습니다.

절차

1. ID 목록 검사 제한을 업데이트합니다.

   ```
   # dsconf -D "cn=Directory Manager" ldap://server.example.com backend config set --
   idlistscanlimit=8000
   ```

   이 명령은 제한을 **8000** 항목으로 설정합니다.

2. 인스턴스를 다시 시작합니다.

   ```
   # dsctl instance_name restart
   ```

3.2. 웹 콘솔을 사용하여 글로벌 인덱스 검사 제한 설정

기본적으로 Directory Server의 ID 목록 검사 제한은 **4000** 입니다. 대부분의 시나리오에서 이 값은 일반적인 데이터베이스 크기 및 액세스 패턴에 적합한 성능을 제공하며 기본값을 변경할 필요가 없습니다. 데이터베이스 인덱스가 4000 항목보다 약간 크지만 전체 디렉토리보다 훨씬 작으면 ID 목록 검사 제한을 늘리면 검색을 개선할 수 있습니다.

반면 제한을 낮추면 4000개의 항목 제한에 도달할 수 있는 검색 속도를 크게 높일 수 있지만 모든 항목을 스캔할 필요는 없습니다.

절차

1. **데이터베이스 글로벌 데이터베이스구성으로 이동합니다**.
2. **ID 목록 검사 제한** 필드를 업데이트합니다.
3. **Save Config** 를 클릭합니다.
4. 오른쪽 **상단에 있는 작업을** 클릭하고 **인스턴스 재시작** 을 선택합니다.

3.3. 명령줄을 사용하여 인덱스 검사 제한 설정

경우에 따라 특정 인덱스에 대한 제한을 정의하거나 ID 목록을 전혀 사용하지 않는 것이 유용합니다. 다양한 유형의 검색 필터에 대한 ID 목록 검사 제한에 대한 개별 설정을 구성할 수 있습니다.

예를 들어 **inetOrgPerson**. 개체 클래스를 포함하는 10 만 개의 항목이 있는 대규모 데이터베이스에서 **(&(objectClass=inetOrgPerson)(uid=user))** 필터는 **objectClass=inetOrgPerson** 과 일치하는 모든 10만 개의 ID가 포함된 ID 목록을 먼저 생성합니다. 데이터베이스가 필터의 두 번째 부분을 적용하면 결과 목록에서 **uid=user** 와 일치하는 오브젝트를 검색합니다. 이 경우 특정 인덱스에 대한 제한을 정의하거나 ID 목록을 전혀 사용하지 않는 것이 유용합니다.

이 절차에서는 AND 절에서 **objectClass=inetOrgPerson** 조건에 대한 ID 목록을 작성하도록 Directory Server를 구성하는 **방법을** 보여줍니다.

절차

- **nsIndexIDListScanLimit** 매개변수를 설정합니다.

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=objectclass,cn=index,cn=userRoot,cn=ldbm database,cn=plugins,cn=config
changetype: modify
replace: nsIndexIDListScanLimit
nsIndexIDListScanLimit: limit=0 type=eq flags=AND values=inetOrgPerson
```

이러한 설정을 사용하면 Directory Server에서 **AND** 절에 **objectClass=inetOrgPerson** 조건에 대한 ID 목록을 생성하지 않습니다. 다른 모든 상황에서 Directory Server는 글로벌 ID 목록 검사 제한 값을 적용합니다.

nsIndexIDListScanLimit 매개변수는 다음 구문을 사용합니다.

```
nsIndexIDListScanLimit: limit=NNN [type=eq[,sub,...]] [flags=AND[,XXX,...]]
[values=val[,val,...]]
```

  - **limit**: ID 목록의 최대 크기를 설정합니다. 유효한 값은 다음과 같습니다.
    - **-1**: 무제한
    - **0**: 인덱스를 사용하지 마십시오
    - **1** 의 최대 32 비트 정수 (**2147483647**): 최대 ID 수
  - **유형**: 선택 사항: 검사 제한 동작을 변경하는 플래그를 설정합니다. 유효한 값은 다음과 같습니다.
    - **AND** : 특성이 AND 절에 표시되는 검색에만 검사 제한을 적용합니다.
    - 또는 : 특성이 **OR** 절에 표시되는 검색에만 검사 제한을 적용합니다.

- **values**: 선택 사항: 제한을 적용하려면 검색 필터와 일치해야 하는 쉼표로 구분된 값 목록입니다. 일치는 한 번에 하나씩 수행되므로 일치하는 값이 있는 경우 값이 일치합니다.
  한 번에 하나의 유형에서만 값을 사용합니다. 값은 인덱스 유형 및 인덱스를 적용할 속성의 구문에 대응해야 합니다. 예를 들어 정수 기반 속성 **uidNumber** 를 지정하고 eq 유형에 대해 인덱싱되는 경우 **type= eq values=*abc*** 를 사용할 수 없습니다.

  값에 이스케이프가 필요한 공백, 쉼표, NULL 또는 기타 값이 포함된 경우 LDAP 필터 이스케이프 구문: 백슬래시(\) 다음에 문자의 2 16진수 코드를 사용합니다. 다음 예에서 DN 값의 쉼표는 **\2C** 로 이스케이프됩니다.

  ```
  nsIndexIDListScanLimit: limit=0 type=eq
  values=uid=user\2Cou=People\2Cdc=example\2Cdc=com
  ```

4장. 잠금 수 조정

Directory Server의 잠금 메커니즘은 동시에 실행할 수 있는 Directory Server 프로세스의 사본 수를 제어합니다. 예를 들어 가져오기 작업 중에 Directory Server는 **/run/lock/dirsrv/slapd-instance_name/imports/** 디렉터리에 잠금을 설정하여 **ns-slapd** Directory Server 프로세스, 다른 가져오기 또는 내보내기 작업이 실행되지 않도록 합니다.

서버가 사용 가능한 잠금이 부족하면 Directory Server는 **/var/log/dirsrv/slapd-instance_name/errors** 파일에 다음 오류를 기록합니다.

```
libdb: Lock table is out of available locks
```

그러나 Directory Server 기본 설정은 서버가 잠금 부족을 실행하여 데이터 손상을 방지하려고 합니다. 자세한 내용은 무료 데이터베이스 잠금을 모니터링하여 데이터 손상 방지를 참조하십시오.

4.1. 데이터베이스 잠금을 모니터링하여 데이터 손상을 방지

데이터베이스 잠금이 부족하면 데이터가 손상될 수 있습니다. 이를 방지하기 위해 Directory Server는 기본적으로 500밀리초마다 사용 가능한 나머지 수의 사용 가능한 데이터베이스 잠금을 모니터링하고 활성 데이터베이스 잠금 수가 90%보다 크거나 같으면 Directory Server가 모든 검색을 중단합니다.

이 절차에서는 간격을 **600** 밀리초로 변경하고 임계값을 **85%** 로 변경합니다.

참고

간격을 너무 높게 설정하면 다음 모니터링 검사가 발생하기 전에 서버가 잠금 부족을 실행할 수 있습니다. 너무 짧은 간격을 설정하면 서버가 느려질 수 있습니다.

절차

1. 간격 및 임계값을 설정합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend config set --locks-monitoring-enabled on --locks-monitoring-pause 600 --locks-monitoring-threshold 85
```

2. 인스턴스를 다시 시작합니다.

```
# dsctl instance_name restart
```

검증

- 잠금 모니터링 설정을 표시합니다.

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com backend config get | grep "nsslapd-db-locks-monitoring"
nsslapd-db-locks-monitoring-enabled: on
nsslapd-db-locks-monitoring-threshold: 85
nsslapd-db-locks-monitoring-pause: 600
```

4.2. 잠금 수를 수동으로 모니터링

디렉터리 서버는 cn=database,cn= **monitor,cn=ldbm 데이터베이스,cn=plugins,cn=config** 의 **nsslapd-db-max-locks** 속성의 **현재** 잠금 수를 추적합니다.

절차

- 잠금 수를 표시하려면 다음을 입력합니다.

```
# ldapsearch -D "cn=Directory Manager" -W -H ldap://server.example.com -x -s sub -b
"cn=database,cn=monitor,cn=ldbm database,cn=plugins,cn=config" nsslapd-db-
current-locks nsslapd-db-max-locks
...
nsslapd-db-current-locks: 37
nsslapd-db-max-locks: 39
```

4.3. 명령줄을 사용하여 잠금 수 설정

dsconf backend config set 명령을 사용하여 디렉터리 서버에서 사용할 수 있는 잠금 수를 업데이트합니다.

절차

1. 잠금 수를 설정합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend config set --
locks=20000
```

   이 명령은 잠금 수를 **20000** 으로 설정합니다.

2. 인스턴스를 다시 시작합니다.

```
# dsctl instance_name restart
```

검증

- **nsslapd-db-locks** 매개변수 값을 표시합니다.

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com backend config get |
grep "nsslapd-db-locks:"
nsslapd-db-locks: 20000
```

4.4. 웹 콘솔을 사용하여 잠금 수 설정

웹 콘솔의 글로벌 데이터베이스 구성에서 사용하는 잠금 디렉터리 서버 수를 설정할 수 있습니다.

사전 요구 사항

- 웹 콘솔에서 인스턴스에 로그인되어 있습니다.

절차

1. **데이터베이스 글로벌 데이터베이스구성으로 이동합니다.**

2. **고급 설정 표시**를 클릭합니다.
3. **모니터링 사용**을 선택하고 임계값 백분율 및 일시 중지 시간을 입력합니다.
4. **Save Config** 를 클릭합니다.
5. **Actions → Restart Instance** (인스턴스 재시작)를 클릭합니다.

검증

1. **데이터베이스 글로벌 데이터베이스구성으로 이동합니다**.
2. **고급 설정 표시**를 클릭합니다.
3. 잠금 모니터링 설정을 확인합니다.

5장. DIRECTORY SERVER 스레드 수 설정

동시 연결을 처리하는 데 사용하는 스레드 디렉터리 서버의 수는 서버의 성능에 영향을 미칩니다. 예를 들어 **추가** 작업과 같이 시간이 많이 걸리는 작업을 처리하는 모든 스레드가 사용 중인 경우 무료 스레드에서 요청을 처리할 때까지 들어오는 새 연결이 큐에 추가됩니다.

서버가 낮은 수의 CPU 스레드를 제공하는 경우 더 많은 수의 스레드를 구성하면 성능이 향상될 수 있습니다. 그러나 CPU 스레드가 많은 서버에서 너무 높은 값을 설정하면 성능이 더 향상되지 않습니다.

기본적으로 Directory Server는 스레드 수를 계산하는 자동 조정 설정을 사용합니다. 이 수는 인스턴스가 시작될 때 서버의 하드웨어 리소스를 기반으로 합니다.

주의

스레드 수를 수동으로 설정하지 마십시오. 대신 auto-tuning 설정을 사용합니다.

활성화된 자동 스레드 튜닝을 통해 Directory Server는 다음과 같은 최적화된 스레드 수를 사용합니다.

CPU 스레드 번호	디렉터리 서버 스레드 번호
1-16	16
17-512	Directory Server 스레드 번호는 시스템의 CPU 스레드 번호와 일치합니다. 예를 들어 시스템에 24개의 CPU 스레드가 있는 경우 Directory Server는 24개의 스레드를 사용합니다. 최대 Directory Server 스레드 수는 512입니다.
512 이상	512. Directory Server는 권장되는 최대 스레드 수를 적용합니다.

5.1. 명령줄을 사용하여 자동 스레드 튜닝 활성화

기본적으로 디렉터리 서버는 사용 가능한 하드웨어를 기반으로 스레드 수를 자동으로 설정합니다. 그러나 경우에 따라 명령줄을 사용하여 이 자동 튜닝 기능을 수동으로 활성화할 수 있습니다.

절차

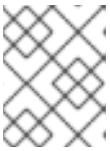
- auto-tuning 기능을 활성화하려면 다음과 같이 **nsslapd-threadnumber** 특성 값을 **-1** 로 설정합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace
nsslapd-threadnumber="-1"

Successfully replaced "nsslapd-threadnumber"
```

검증

- 이제 명령에서 Directory Server에서 사용하는 트랜젝션 수를 확인합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config get nsslapd-threadnumber

nsslapd-threadnumber: 16
```



참고

명령은 올바른 하드웨어 리소스를 기반으로 Directory Server가 계산된 스레드 수를 검색합니다.

추가 리소스

- **nsslapd-threadnumber** 속성 설명.

5.2. 웹 콘솔을 사용하여 자동 스레드 튜닝 활성화

기본적으로 디렉터리 서버는 사용 가능한 하드웨어를 기반으로 스레드 수를 자동으로 설정합니다. 그러나 경우에 따라 웹 콘솔을 사용하여 이 자동 튜닝 기능을 수동으로 활성화할 수 있습니다.

사전 요구 사항

- 웹 콘솔에서 인스턴스에 로그인되어 있습니다. 자세한 내용은 웹 콘솔을 사용하여 디렉터리 서버에 로그인을 참조하십시오.

절차

1. **Server → Tuning & Limits** 로 이동합니다.
2. **Number of Worker Threads 필드에서 스레드**수를 **-1** 로 설정합니다.
3. **Save Settings** 를 클릭합니다.

추가 리소스

- **nsslapd-threadnumber** 속성 설명.

5.3. 명령줄을 사용하여 수동으로 스레드 수 설정

경우에 따라 고정된 수의 Directory Server 스레드를 수동으로 설정해야 합니다. 예를 들어 자동 튜닝 설정을 사용하지 않고 가상 머신의 CPU 코어 수를 변경하지 않으면 Directory Server 스레드 수를 조정하면 성능이 향상될 수 있습니다.

또한 이전에 특정 개수의 스레드를 설정하는 경우 이 절차를 사용하여 자동 튜닝 설정을 다시 활성화할 수도 있습니다.

절차

- Directory Server에서 사용해야 하는 스레드 수를 설정합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace
nsslapd-threadnumber="64"

Successfully replaced "nsslapd-threadnumber"
```

nsslapd-threadnumber 매개변수를 **-1** 로 설정하여 자동 튜닝 설정을 활성화합니다.

5.4. 웹 콘솔을 사용하여 수동으로 스레드 수 설정

특정 상황에서는 수정 번호의 Directory Server 스레드를 수동으로 설정해야 합니다. 예를 들어 자동 튜닝 설정을 사용하지 않고 가상 머신의 CPU 코어 수를 변경하지 않으면 Directory Server 스레드 수를 조정하면 성능이 향상될 수 있습니다.

이전에 특정 개수의 스레드를 설정하는 경우 웹 콘솔을 사용하여 자동 튜닝 설정을 다시 활성화할 수 있습니다.

사전 요구 사항

- 웹 콘솔에서 인스턴스에 로그인되어 있습니다.

절차

1. **Server → Tuning & Limits** 로 이동합니다.
2. **Number of Worker Threads 필드에서 스레드**수를 설정합니다.
3. **Save Settings** 를 클릭합니다.

6장. 리소스 제한 튜닝

Directory Server는 인스턴스에서 사용하는 리소스의 양을 조정하는 몇 가지 설정을 제공합니다. 명령줄 또는 웹 콘솔을 사용하여 변경할 수 있습니다.

6.1. 명령줄을 사용하여 리소스 제한 설정 업데이트

이 섹션에서는 리소스 제한 설정을 변경하는 일반적인 절차를 설명합니다. 환경에 따라 설정을 조정합니다.

절차

1. 성능 설정을 업데이트합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace
parameter_name=value
```

   다음 매개변수를 설정할 수 있습니다.

   - **nsslapd-threadnumber**: 작업자 스레드 수를 설정합니다.
   - **nsslapd-maxdescriptors**: 파일 디스크립터의 최대 수를 설정합니다.
   - **nsslapd-timelimit**: 검색 시간 제한을 설정합니다.
   - **nsslapd-sizelimit**: 검색 크기 제한을 설정합니다.
   - **nsslapd-pagedsizelimit**: paged 검색 크기 제한을 설정합니다.
   - **nsslapd-idletimeout**: 유휴 연결 시간 초과를 설정합니다.
   - **nsslapd-ioblocktimeout**: 입력/출력(I/O) 블록 시간 초과를 설정합니다.
   - **nsslapd-ndn-cache-enabled**: 정규화된 DN 캐시를 활성화하거나 비활성화합니다.
   - **nsslapd-ndn-cache-max-size**: nsslapd-ndn-cache-enabled가 활성화된 경우 정규화된 DN 캐시 크기를 설정합니다.
   - **nsslapd-outbound-ldap-io-timeout**: 아웃바운드 I/O 시간 초과를 설정합니다.
   - **nsslapd-maxbersize**: 기본 인코딩 규칙 (BER)의 최대 크기를 설정합니다.
   - **nsslapd-maxsasliosize**: 최대 SASL (Simple Authentication and Security Layer) I/O 크기를 설정합니다.
   - **nsslapd-listen-backlog-size**: 들어오는 연결을 수신하는 데 사용할 수 있는 최대 소켓 수를 설정합니다.
   - **nsslapd-max-filter-nest-level**: 중첩된 최대 필터 수준을 설정합니다.
   - **nsslapd-ignore-virtual-attrs**: 가상 속성 조회를 활성화하거나 비활성화합니다.
   - **nsslapd-connection-nocanon**: 역방향 DNS 조회를 활성화하거나 비활성화합니다.
   - **nsslapd-enable-turbo-mode**: turbo 모드 기능을 활성화하거나 비활성화합니다.
     자세한 내용은 구성 및 스키마 참조 의 매개변수 설명을 참조하십시오.

2. 인스턴스를 다시 시작합니다.

```
# dsctl instance_name restart
```

6.2. 웹 콘솔을 사용하여 리소스 제한 설정 업데이트

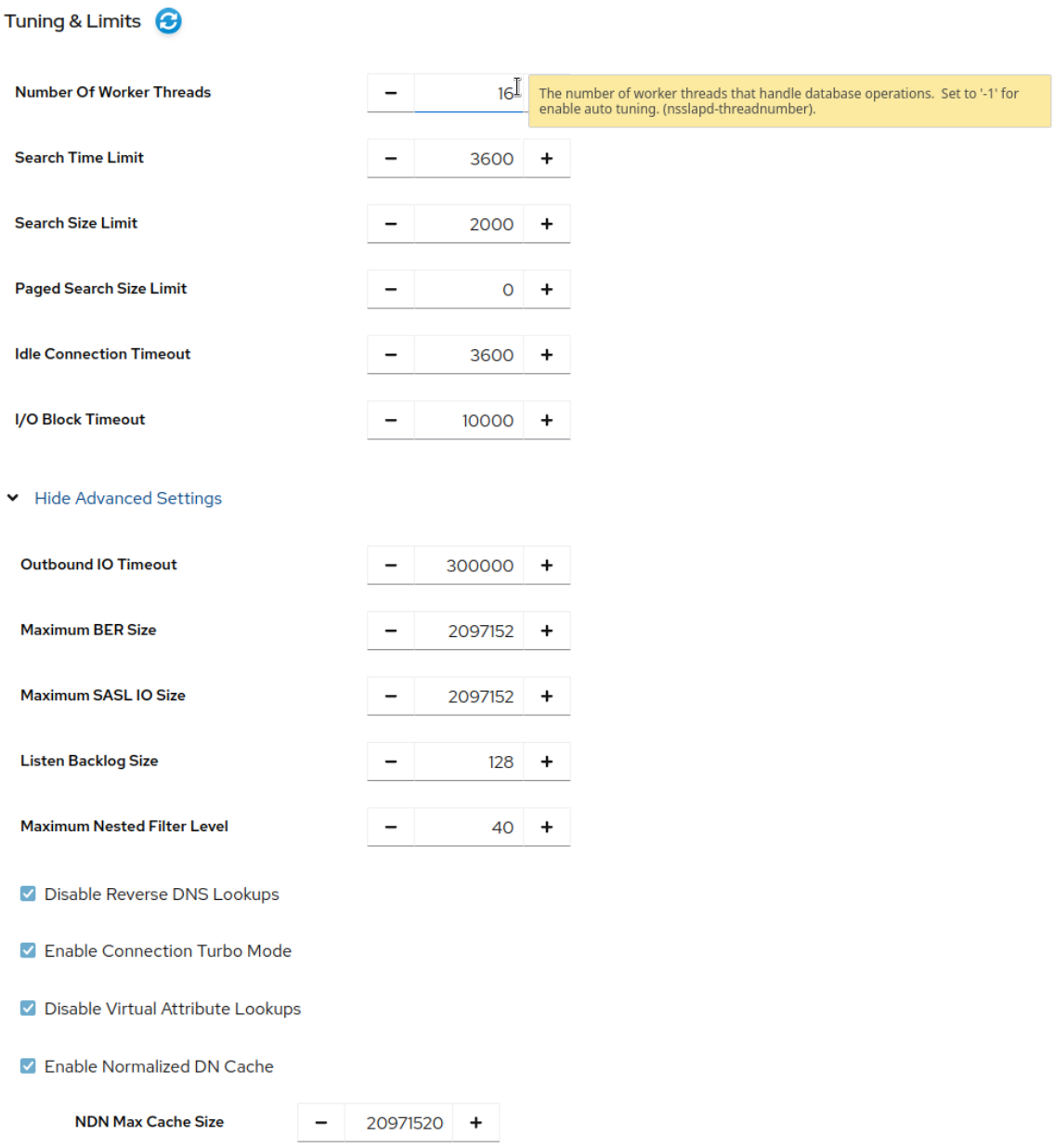
이 섹션에서는 리소스 제한 설정을 변경하는 일반적인 절차를 설명합니다. 환경에 따라 설정을 조정합니다.

사전 요구 사항

- 웹 콘솔에서 인스턴스에 로그인되어 있습니다.

절차

1. **Server → Tuning & Limits** 로 이동합니다.
2. 설정을 업데이트합니다. 필요한 경우 **고급 설정 표시**를 클릭하여 모든 설정을 표시합니다.



3. **Save Settings** 를 클릭합니다.

4. **Actions → Restart Instance** (인스턴스 재시작)를 클릭합니다.

6.3. 투명한 대규모 페이지 기능 비활성화

THP(Transparent Huge Pages)는 대규모 메모리 페이지를 사용하여 대용량 메모리가 있는 머신에서 TLB(Translation Lookaside Buffer) 검사를 가속화하는 Linux의 메모리 관리 기능입니다. THP 기능은 RHEL 시스템에서 기본적으로 활성화되어 있으며 2MB 메모리 페이지를 지원합니다.

그러나 THP 기능은 연속적인 대규모 할당 패턴에서 활성화되어 있으며 Red Hat Directory Server에 일반적인 작고 스파스 할당 패턴에서 성능을 저하시킬 수 있습니다. 프로세스의 상주 메모리 크기는 결국 제한을 초과하여 성능에 영향을 미치거나 OOM(메모리 부족) 킬러에 의해 종료될 수 있습니다.

중요

성능 및 메모리 사용 문제를 방지하려면 Red Hat Directory Server가 설치된 RHEL 시스템에서 THP를 비활성화합니다.

절차

1. THP의 현재 상태를 확인합니다.

```
# cat /sys/kernel/mm/transparent_hugepage/enabled
```

2. 투명한 대규모 페이지 기능이 활성화된 경우 부팅 시 또는 런타임 중 하나를 비활성화합니다.
   - **grub.conf** 파일의 커널 명령줄에 다음을 추가하여 부팅 시 투명한 대규모 페이지를 비활성화합니다.

```
transparent_hugepage=never
```

   - 다음 명령을 실행하여 런타임에 투명한 대규모 페이지를 비활성화합니다.

```
# echo never > /sys/kernel/mm/transparent_hugepage/enabled
# echo never > /sys/kernel/mm/transparent_hugepage/defrag
```

추가 리소스

- RHDS에서 THP(Transparent Huge Pages)의 부정적인 영향
- RHEL 7에서 투명한 대규모 페이지 구성

7장. 검색 작업당 통계 로깅

일부 검색 작업 중에 특히 **(cn=user*)** 와 같은 필터를 사용하면 서버가 작업을 수신한 후 결과 전송 **시간이**매우 길 수 있습니다.

검색 작업 중에 사용되는 인덱스와 관련된 정보를 사용하여 액세스 로그를 확장하면 **etime** 값이 리소스 비용이 많이 드는 이유를 진단하는 데 도움이 됩니다.

nsslapd-statlog-level 속성을 사용하여 서버에 미치는 영향을 최소화하면서 각 검색 작업에 대한 인덱스 조회 수(데이터베이스 읽기 작업) 및 전체 인덱스 조회 기간과 같은 통계 수집을 활성화합니다.

사전 요구 사항

- 액세스 로깅을 활성화했습니다.

절차

1. 검색 작업 메트릭을 활성화합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace
nsslapd-statlog-level=1
```

2. 인스턴스를 다시 시작합니다.

```
# dsctl instance_name restart
```

검증

1. 검색 작업을 수행합니다.

```
# ldapsearch -D "cn=Directory Manager" -H ldap://server.example.com -b
"dc=example,dc=com" -s sub -x "cn=user*"
```

2. 액세스 로그 파일을 보고 검색 통계 레코드를 찾습니다.

```
# cat /var/log/dirsrv/slapd-instance_name/access
...
[16/Nov/2022:11:34:11.834135997 +0100] conn=1 op=73 SRCH
base="dc=example,dc=com" scope=2 filter="(cn=user)"* attrs=ALL
[16/Nov/2022:11:34:11.835750508 +0100] conn=1 op=73 STAT read index:
attribute=objectclass key(eq)=referral --> count 0
[16/Nov/2022:11:34:11.836648697 +0100] conn=1 op=73 STAT read index: attribute=cn
key(sub)=er_ --> count 25
[16/Nov/2022:11:34:11.837538489 +0100] conn=1 op=73 STAT read index: attribute=cn
key(sub)=ser --> count 25
[16/Nov/2022:11:34:11.838814948 +0100] conn=1 op=73 STAT read index: attribute=cn
key(sub)=use --> count 25
[16/Nov/2022:11:34:11.841241531 +0100] conn=1 op=73 STAT read index: attribute=cn
key(sub)=^us --> count 25
[16/Nov/2022:11:34:11.842230318 +0100] conn=1 op=73 STAT read index: duration
0.000010276
[16/Nov/2022:11:34:11.843185322 +0100] conn=1 op=73 RESULT err=0 tag=101
nentries=24 wtime=0.000078414 optime=0.001614101 etime=0.001690742
...
```

-

추가 리소스

- nsslapd-statlog-level 설명에 대한 링크

8장. 디스크 부족에서 DIRECTORY SERVER를 종료하도록 로컬 디스크 모니터링

시스템에서 사용 가능한 디스크 공간이 너무 작아지면 Directory Server 프로세스가 종료됩니다. 결과적으로 데이터베이스를 손상시키거나 데이터를 손실할 위험이 있습니다. 이 문제를 방지하려면 구성, 트랜잭션 로그 및 데이터베이스 디렉터리가 포함된 파일 시스템에서 사용 가능한 디스크 공간을 모니터링하도록 Directory Server를 구성할 수 있습니다. 사용 가능한 공간이 구성된 임계값에 도달하면 Directory Server에서 인스턴스를 종료합니다.

8.1. 사용 가능한 디스크 공간 크기에 따라 디렉터리 서버 동작

모니터링을 구성할 때 Directory Server가 작동하는 방식은 남은 여유 공간 양에 따라 다릅니다.

- 사용 가능한 디스크 공간이 정의된 임계값에 도달하면 Directory Server입니다.
  - 자세한 로깅 비활성화
  - 액세스 액세스 로깅 비활성화
  - 보관된 로그 파일 삭제

참고

임계값에 도달한 경우에도 Directory Server는 항상 오류 로그를 계속 작성합니다.

- 사용 가능한 디스크 공간이 구성된 임계값의 절반보다 작으면 디렉터리 서버가 정의된 유예 기간 내에 종료됩니다.
- 사용 가능한 디스크 공간이 4KB보다 작으면 디렉터리 서버가 즉시 종료됩니다.

디스크 공간이 확보되면 Directory Server는 종료 프로세스를 중지하고 이전에 비활성화된 모든 로그 설정을 다시 활성화합니다.

8.2. 명령줄을 사용하여 로컬 디스크 모니터링 구성

Directory Server는 구성, 트랜잭션 로그 및 데이터베이스 디렉터리가 포함된 파일 시스템에서 사용 가능한 디스크 공간을 모니터링할 수 있습니다. 사용 가능한 나머지 공간에 따라 Directory Server는 특정 로깅 기능을 비활성화하거나 종료합니다.

절차

1. 디스크 모니터링 기능을 활성화하고 임계값 및 유예 기간을 설정합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace
nsslapd-disk-monitoring=on nsslapd-disk-monitoring-threshold=3221225472 nsslapd-
disk-monitoring-grace-period=60
```

   이 명령은 사용 가능한 디스크 공간 임계값을 3GB (3,221,225,472 바이트)로 설정하고 유예 기간을 60초로 설정합니다.

2. 선택 사항: Directory Server를 구성하여 액세스 로깅을 비활성화하거나 보관된 로그를 삭제하도록 구성합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace
nsslapd-disk-monitoring-logging-critical=on
```

3. 인스턴스를 다시 시작합니다.

```
# dsctl instance_name restart
```

8.3. 웹 콘솔을 사용하여 로컬 디스크 모니터링 구성

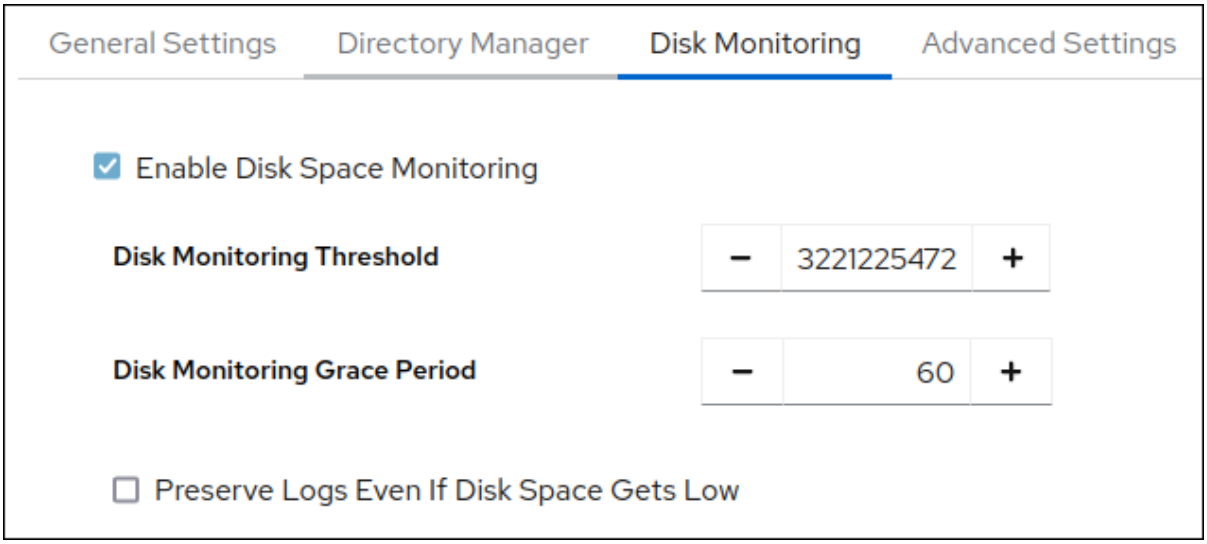
Directory Server는 구성, 트랜잭션 로그 및 데이터베이스 디렉터리가 포함된 파일 시스템에서 사용 가능한 디스크 공간을 모니터링할 수 있습니다. 사용 가능한 나머지 공간에 따라 Directory Server는 특정 로깅 기능을 비활성화하거나 종료합니다.

사전 요구 사항

- 웹 콘솔에서 인스턴스에 로그인되어 있습니다.

절차

1. **Server → Server Settings → Disk Monitoring** 으로 이동합니다.
2. **Enable Disk Space Monitoring** 을 선택합니다. 임계값을 바이트 단위로 설정하고 유예 기간을 분 단위로 설정합니다.



이 예에서는 모니터링 임계값을 3GB (3,221,225,472 바이트)로 설정하고, 임계값에 도달한 후 디렉터리 서버가 인스턴스를 종료하기 전의 시간을 60분으로 설정합니다.

3. 선택 사항: **디스크 공간이 낮게 되는 경우 Preserve Logs Even를 선택합니다.**
4. **Save Settings** 를 클릭합니다.
5. 오른쪽 **상단에 있는 작업을** 클릭하고 **인스턴스 재시작** 을 선택합니다.

9장. 트랜잭션 로깅 튜닝

모든 Directory Server 인스턴스에는 관리하는 데이터베이스 업데이트를 기록하는 트랜잭션 로그가 포함되어 있습니다. 수정 작업과 같은 디렉터리 데이터베이스 작업이 수행될 때마다 서버는 해당 LDAP 작업의 결과로 호출되는 모든 데이터베이스 작업에 대해 단일 데이터베이스 트랜잭션을 생성합니다. 여기에는 항목이 포함된 데이터베이스 파일의 항목 레코드를 업데이트하고 모든 속성 인덱스를 업데이트하는 작업이 포함됩니다. 모든 작업이 성공하면 서버는 트랜잭션을 커밋하고, 작업을 트랜잭션 로그에 쓰고, 전체 트랜잭션이 디스크에 기록되는지 확인합니다. 이러한 작업이 하나라도 실패하면 서버에서 트랜잭션을 롤백하고 모든 작업이 삭제됩니다. 이 all-or-nothing 접근 방식은 업데이트 작업이 atomic임을 보장합니다. 전체 작업이 영구적으로 성공하고 되돌릴 수 없거나 실패합니다.

주기적으로 Directory Server는 트랜잭션 로그의 내용을 실제 데이터베이스 인덱스 파일로 플러시하고 트랜잭션 로그에 트리밍이 필요한지 확인합니다.

서버에 정전과 같은 오류가 발생하고 비정상적으로 종료되는 경우 최근 디렉터리 변경에 대한 정보는 트랜잭션 로그에 의해 계속 저장됩니다. 서버가 다시 시작되면 서버는 오류 상태를 자동으로 감지하고 데이터베이스 트랜잭션 로그를 사용하여 데이터베이스를 복구합니다.

데이터베이스 트랜잭션 로깅, 플러시 데이터베이스, 트리밍 및 데이터베이스 복구는 개입이 필요한 자동 프로세스이지만 성능을 최적화하기 위해 데이터베이스 트랜잭션 로깅 속성 중 일부를 조정하는 것이 좋습니다.

9.1. 명령줄을 사용하여 데이터베이스 체크섬 간격 변경

정기적으로 Directory Server는 트랜잭션 로그에 기록된 트랜잭션을 데이터베이스 파일에 작성하고 데이터베이스 트랜잭션 로그에 Checkpoint 항목을 기록합니다.At regular intervals, Directory Server writes the transactions logged in the transaction log to the database files and logs a checkpoint entry in the database transaction log. 데이터베이스에 이미 작성된 변경 사항을 표시하여 체크포인트 항목은 트랜잭션 로그에서 복구를 시작할 위치를 표시하므로 복구 프로세스의 속도가 빨라집니다.

기본적으로 Directory Server는 60초마다 데이터베이스 트랜잭션 로그에 체크포인트 항목을 보냅니다. 체크포인트 간격을 늘리면 디렉터리 쓰기 작업의 성능이 향상될 수 있습니다. 그러나 무질서한 종료 후 디렉터리 데이터베이스를 복구하는 데 필요한 시간을 늘리고 데이터베이스 트랜잭션 로그 파일로 인해 더 많은 디스크 공간이 필요할 수도 있습니다.

절차

- 체크포인트 간격을 (예: 120초)로 변경합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend config set --checkpoint-interval=120
```

9.2. 웹 콘솔을 사용하여 데이터베이스 체크섬 간격 변경

정기적으로 Directory Server는 트랜잭션 로그에 기록된 트랜잭션을 데이터베이스 파일에 작성하고 데이터베이스 트랜잭션 로그에 Checkpoint 항목을 기록합니다.At regular intervals, Directory Server writes the transactions logged in the transaction log to the database files and logs a checkpoint entry in the database transaction log. 데이터베이스에 이미 작성된 변경 사항을 표시하여 체크포인트 항목은 트랜잭션 로그에서 복구를 시작할 위치를 표시하므로 복구 프로세스의 속도가 빨라집니다.

기본적으로 Directory Server는 60초마다 데이터베이스 트랜잭션 로그에 체크포인트 항목을 보냅니다. 체크포인트 간격을 늘리면 디렉터리 쓰기 작업의 성능이 향상될 수 있습니다. 그러나 무질서한 종료 후 디렉터리 데이터베이스를 복구하는 데 필요한 시간을 늘리고 데이터베이스 트랜잭션 로그 파일로 인해 더 많은 디스크 공간이 필요할 수도 있습니다.

사전 요구 사항

- 웹 콘솔에서 인스턴스에 로그인되어 있습니다.

절차

1. **데이터베이스 글로벌 데이터베이스구성으로 이동합니다**.
2. **고급 설정 표시를** 클릭합니다.
3. **Database Checkpoint Interval** 필드에서 값을 업데이트합니다.
4. **Save Configuration** 을 클릭합니다.

9.3. CRYOSTAT 트랜잭션 비활성화

Cryostat 트랜잭션 로깅은 트랜잭션의 일련의 데이터베이스 작업으로 구성된 각 LDAP 업데이트 작업이 디스크에 물리적으로 기록됨을 의미합니다. 각 LDAP 작업을 여러 데이터베이스 업데이트로 구성할 수 있지만 각 LDAP 작업은 단일 데이터베이스 트랜잭션으로 처리됩니다. 각 LDAP 작업은 atomic 및 Cryostat 입니다.

주의

Cryostat 트랜잭션을 활성화하면 데이터 손실 위험이 있는 Directory Server의 쓰기 성능이 향상될 수 있습니다.

Cryostat 트랜잭션 로깅을 비활성화하면 Directory Server는 모든 디렉터리 데이터베이스 작업을 데이터베이스 트랜잭션 로그 파일에 기록하지만 즉시 디스크에 물리적으로 기록되지 않을 수 있습니다. 디렉토리 변경이 논리 데이터베이스 트랜잭션 로그 파일에 기록되었지만 시스템 충돌 시 디스크에 물리적으로 기록되지 않은 경우 변경 사항을 복구할 수 없습니다. Cryostat 트랜잭션이 비활성화되면 복구된 데이터베이스는 일관되게 유지되지만 시스템 충돌 직전에 완료된 LDAP 쓰기 작업의 결과가 반영되지 않습니다.

Directory Server가 실행 중인 경우 **nsslapd-db-durable-** Cryostat 매개변수를 변경할 수 없습니다.

절차

1. 인스턴스를 중지합니다.

```
# dsctl instance_name stop
```

2. **/etc/dirsrv/slapd-*instance_name*/dse.ldif** 파일을 편집하고 cn=config,cn= **ldbm 데이터베이스,cn=plugins,cn=config** 항목을 **off** 로 **nsslapd-db-durable-** Cryostat 매개변수를 설정합니다.

```
dn: cn=config,cn=ldbm database,cn=plugins,cn=config
...
nsslapd-db-durable-transaction: off
...
```

3. 인스턴스를 시작합니다.

```
# dsctl instance_name start
```

10장. 캐시 설정 관리

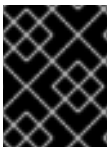
Directory Server는 다음 캐시를 사용합니다.

- 개별 디렉터리 항목이 포함된 항목 캐시입니다.
- DN(고유 이름) 캐시는 DN 및 RDN(유치 고유 이름)을 항목과 연결하는 데 사용됩니다.
- 데이터베이스 인덱스 파일 ***.db** 파일이 포함된 데이터베이스 캐시입니다.

성능 향상을 위해 모든 캐시 크기는 모든 레코드를 저장할 수 있어야 합니다. 권장 자동 크기 조정 기능을 사용하지 않고 사용 가능한 RAM이 충분하지 않은 경우 이전에 표시된 순서대로 캐시에 사용 가능한 메모리를 할당합니다.

10.1. CACHE-AUTOSIZE 및 CACHE-AUTOSIZE-SPLIT 매개변수가 데이터베이스 및 항목 캐시 크기에 미치는 영향

기본적으로 Directory Server는 자동 크기 조정 기능을 사용하여 인스턴스가 시작될 때 서버의 하드웨어 리소스에 대한 데이터베이스 및 항목 캐시의 크기를 최적화합니다.



중요

Red Hat은 자동 크기 조정 기능을 사용하여 캐시 크기를 수동으로 설정하지 않는 것이 좋습니다.

cn=config,cn=ldbm 데이터베이스,cn=plugins,cn=config 항목의 다음 매개변수는 auto-sizing을 제어합니다.

nsslapd-cache-autosize

이러한 설정은 데이터베이스 및 항목 캐시에 자동 크기 조정이 활성화되어 있는지 제어합니다. 자동 크기 조정이 활성화되어 있습니다.

- 데이터베이스 및 항목 캐시 모두에서 **nsslapd-cache-autosize** 매개변수가 **0** 보다 큰 값으로 설정된 경우
- 데이터베이스 캐시의 경우 **nsslapd-cache-autosize** 및 **nsslapd-dbcachesize** 매개변수가 **0** 으로 설정된 경우
- 항목 캐시의 경우 **nsslapd-cache-autosize** 및 **nsslapd-cachememsize** 매개변수가 **0** 으로 설정된 경우

nsslapd-cache-autosize-split

- 이 값은 Directory Server가 데이터베이스 캐시에 사용하는 RAM의 백분율을 설정합니다. 서버는 항목 캐시에 나머지 백분율을 사용합니다.
- 데이터베이스 캐시에 1.5GB 이상의 RAM을 사용하면 성능이 향상되지 않습니다. 따라서 Directory Server는 데이터베이스 캐시를 1.5GB로 제한합니다.

기본적으로 Directory Server는 다음 기본값을 사용합니다.

- **nsslapd-cache-autosize: 25**
- **nsslapd-cache-autosize-split: 25**

- **nsslapd-dbcachesize: 1,536MB**

이러한 설정을 사용하면 시스템의 사용 가능한 RAM의 25%가 사용됩니다(**nsslapd-cache-autosize**). 이 메모리에서 서버는 데이터베이스 캐시에 25%를 사용하고(**nsslapd-cache-autosize-split**) 및 나머지 75%는 항목 캐시에 사용합니다.

사용 가능한 RAM에 따라 다음과 같은 캐시 크기가 생성됩니다.

표 10.1. nsslapd-cache-autosize 및 nsslapd-cache-split에서 기본값을 사용하는 경우 캐시 크기

GB of free RAM	데이터베이스 캐시 크기	항목 캐시 크기
1GB	64MB	192MB
2GB	128MB	384MB
4GB	256MB	768MB
8GB	512MB	1,536MB
16GB	1,024MB	3,072MB
32GB	1,536MB	6,656MB
64GB	1,536MB	14,848MB
128GB	1,536MB	31232MB

10.2. 필요한 캐시 크기

dsconf monitor dbmon 명령을 사용하면 런타임 시 캐시 통계를 모니터링할 수 있습니다. 통계를 표시하려면 다음을 입력합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com monitor dbmon
DB Monitor Report: 2022-02-24 10:25:16
--------------------------------------------------------
Database Cache:
 - Cache Hit Ratio:     50%
 - Free Space:          397.31 KB
 - Free Percentage:     2.2%
 - RO Page Drops:       0
 - Pages In:            2934772
 - Pages Out:           219075

Normalized DN Cache:
 - Cache Hit Ratio:     60%
 - Free Space:          19.98 MB
 - Free Percentage:     99.9%
 - DN Count:            100000
 - Evictions:           9282348

Backends:
```

```
- dc=example,dc=com (userroot):
  - Entry Cache Hit Ratio:       66%
  - Entry Cache Count:           50000
  - Entry Cache Free Space:      2.0 KB
  - Entry Cache Free Percentage: 0.8%
  - Entry Cache Average Size:    8.9 KB
  - DN Cache Hit Ratio:          21%
  - DN Cache Count:              100000
  - DN Cache Free Space:         4.29 MB
  - DN Cache Free Percentage:    69.8%
  - DN Cache Average Size:       130.0 B
```

선택적으로 **-b *back_end*** 또는 **-x** 옵션을 명령에 전달하여 특정 백엔드 또는 인덱스의 통계를 표시합니다.

캐시 크기가 충분한 경우 **DN 캐시 수의 수는 캐시** 수 백엔드 항목의 값과 일치합니다. **또한 모든 항목과 DN이 해당 캐시에 적합한 경우 Entry Cache Count 값은 DN 캐시 수 값과 일치합니다.**

이 예제의 출력은 다음과 같습니다.

- **2.2%의 무료 데이터베이스 캐시만 남아 있습니다:**

```
Database Cache:
 ...
 - Free Space:         397.31 KB
 - Free Percentage:    2.2%
```

그러나 효율적으로 운영하려면 최소 15 % 이상의 무료 데이터베이스 캐시가 필요합니다. 데이터베이스 캐시의 최적 크기를 확인하려면 하위 디렉터리와 변경 로그 데이터베이스를 포함하여 /var/lib/dirsrv/slapd-*instance_name*/db/ 디렉토리에서 모든 *.db 파일의 크기를 계산하고 오버헤드에 12%를 추가합니다.

데이터베이스 캐시를 설정하려면 명령줄을 사용하여 데이터베이스 캐시 크기 설정을 참조하십시오.

- **userroot 데이터베이스의 DN 캐시는 다음과 같이 잘 선택됩니다.**

```
Backends:
 - dc=example,dc=com (userroot):
   ...
   - DN Cache Count:              100000
   - DN Cache Free Space:         4.29 MB
   - DN Cache Free Percentage:    69.8%
   - DN Cache Average Size:       130.0 B
```

데이터베이스의 DN 캐시에는 100000개의 레코드가 포함되어 있습니다. 캐시의 69, percent는 사용 불가능합니다. 각 메모리의 DN은 평균 Cryostat 바이트가 필요합니다.

DN 캐시를 설정하려면 명령줄을 사용하여 DN 캐시 크기 설정을 참조하십시오.

- userroot 데이터베이스의 항목 캐시의 통계는 성능 향상을 위해 항목 캐시 값을 늘려야 함을 나타냅니다.

```
Backends:
 - dc=example,dc=com (userroot):
 ...
  - Entry Cache Count:          50000
  - Entry Cache Free Space:     2.0 KB
  - Entry Cache Free Percentage: 0.8%
  - Entry Cache Average Size:    8.9 KB
```

이 데이터베이스 50000 레코드에는 항목 캐시에 포함되어 있으며 2 킬로바이트의 여유 공간만 남아 있습니다. Directory Server가 100000 DN을 모두 캐시할 수 있도록 하려면 캐시를 최소 890MB(00000 DNs * 8,9KB 평균 항목 크기)로 늘려야 합니다. 그러나 Red Hat은 필요한 최소 크기를 다음 최고 GB로 반올림하고 그 결과를 두 배로 늘릴 것을 권장합니다. 이 예에서는 항목 캐시를 2GB로 설정해야 합니다.

항목 캐시를 설정하려면 명령줄을 사용하여 항목 캐시 크기 설정을 참조하십시오.

10.3. 명령줄을 사용하여 데이터베이스 캐시 크기 설정

데이터베이스 캐시에는 데이터베이스의 Berkeley 데이터베이스 인덱스 파일이 포함되어 있습니다. 즉, 데이터베이스에서 특성 인덱싱에 사용되는 *.db 및 기타 모든 파일이 포함됩니다. 이 값은 Berkeley DB API 함수 set_cachesize() 에 전달됩니다. 이 캐시 크기는 항목 캐시 크기보다 Directory Server 성능에 미치는 영향은 줄어들지만 항목 캐시 크기가 설정된 후 사용 가능한 RAM이 있는 경우 데이터베이스 캐시에 할당된 메모리 양을 늘립니다.

절차

1. 자동 캐시 튜닝 비활성화

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend config set --cache-autosize=0
```

2. 데이터베이스 캐시 크기를 수동으로 설정합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend config set --
dbcachesize=268435456
```

데이터베이스 캐시 크기를 바이트 단위로 지정합니다. 이 예제에서 명령은 데이터베이스 캐시를 256MB로 설정합니다.

3. 인스턴스를 다시 시작합니다.

```
# dsctl instance_name restart
```

10.4. 웹 콘솔을 사용하여 데이터베이스 캐시 크기 설정

데이터베이스 캐시에는 데이터베이스의 Berkeley 데이터베이스 인덱스 파일이 포함되어 있습니다. 즉, 데이터베이스에서 특성 인덱싱에 사용되는 *.db 및 기타 모든 파일이 포함됩니다. 이 값은 Berkeley DB API 함수 set_cachesize() 에 전달됩니다. 이 캐시 크기는 항목 캐시 크기보다 Directory Server 성능에 미치는 영향은 줄어들지만 항목 캐시 크기가 설정된 후 사용 가능한 RAM이 있는 경우 데이터베이스 캐시에 할당된 메모리 양을 늘립니다.

사전 요구 사항

- 웹 콘솔에서 인스턴스에 로그인되어 있습니다.

절차

1. 데이터베이스 글로벌 데이터베이스구성으로 이동합니다.
2. 자동 캐시 튜닝을 선택 해제합니다.
3. Save Config 를 클릭합니다.
4. 256MB의 268435456 과 같은 데이터베이스 캐시 크기를 바이트 단위로 입력합니다.
5. Save Config 를 클릭합니다.

6. 오른쪽 상단에 있는 작업을 클릭하고 인스턴스 재시작 을 선택합니다.

10.5. 명령줄을 사용하여 DN 캐시 크기 설정

Directory Server는 진입점 인덱스를 사용하여 고유 이름(DN) 및 RDN(relative 고유 이름)을 항목과 연결합니다. 이를 통해 서버는 하위 트리의 이름을 효율적으로 변경하고 항목을 이동하고 moddn 작업을 수행할 수 있습니다. 서버는 DN 캐시를 사용하여 입력된 인덱스의 메모리 내 표현을 캐시하여 비용이 많이 드는 파일 I/O 및 변환 작업을 방지합니다.

특히 항목 이름 변경 및 작업 이동과 같이 최상의 성능을 위해 자동 튜닝 기능을 사용하지 않는 경우 DN 캐시를 데이터베이스의 모든 DN을 캐시할 수 있는 크기로 설정합니다.

DN이 캐시에 저장되지 않은 경우 Directory Server는 entryrdn.db 인덱스 데이터베이스 파일에서 DN을 읽고 디스크의 DN을 디스크 형식에서 메모리 내 형식으로 변환합니다. 캐시에 저장된 DNS를 사용하면 서버가 디스크 I/O 및 변환 단계를 건너뛸 수 있습니다.

절차

1. 접미사와 해당 백엔드를 표시합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com suffix list
dc=example,dc=com (userroot)
```

이 명령은 각 접미사 옆에 백엔드 데이터베이스의 이름을 표시합니다. 다음 단계에서 접미사의 데이터베이스 이름이 필요합니다.

2. DN 캐시 크기를 설정합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend suffix set --
dncache-memsize=20971520 userRoot
```

이 명령은 사용자Root 데이터베이스의 DN 캐시를 20MB로 설정합니다.

3. 인스턴스를 다시 시작합니다.

```
# dsctl instance_name restart
```

10.6. 웹 콘솔을 사용하여 DN 캐시 크기 설정

Directory Server는 진입점 인덱스를 사용하여 고유 이름(DN) 및 RDN(relative 고유 이름)을 항목과 연결합니다. 이를 통해 서버는 하위 트리의 이름을 효율적으로 변경하고 항목을 이동하고 moddn 작업을 수행할 수 있습니다. 서버는 DN 캐시를 사용하여 입력된 인덱스의 메모리 내 표현을 캐시하여 비용이 많이 드는 파일 I/O 및 변환 작업을 방지합니다.

특히 항목 이름 변경 및 작업 이동과 같이 최상의 성능을 위해 자동 튜닝 기능을 사용하지 않는 경우 DN 캐시를 데이터베이스의 모든 DN을 캐시할 수 있는 크기로 설정합니다.

DN이 캐시에 저장되지 않은 경우 Directory Server는 entryrdn.db 인덱스 데이터베이스 파일에서 DN을 읽고 디스크의 DN을 디스크 형식에서 메모리 내 형식으로 변환합니다. 캐시에 저장된 DNS를 사용하면 서버가 디스크 I/O 및 변환 단계를 건너뛸 수 있습니다.

사전 요구 사항

- 웹 콘솔에서 인스턴스에 로그인되어 있습니다.

절차

1. Database → Suffixes → *suffix_name* 으로 이동합니다.
2. DN 캐시 크기를 바이트 단위로 DN 캐시 크기 필드에 입력합니다.
3. Save Configuration 을 클릭합니다.
4. 오른쪽 상단에 있는 작업을 클릭하고 인스턴스 재시작 을 선택합니다.

10.7. 명령줄을 사용하여 항목 캐시 크기 설정

Directory Server는 항목 캐시를 사용하여 검색 및 읽기 작업 중에 사용되는 디렉터리 항목을 저장합니다. 항목 캐시를 모든 레코드를 저장할 수 있는 크기로 설정하면 검색 작업에 가장 큰 영향을 미칩니다.

항목 캐싱이 구성되지 않은 경우 Directory Server는 id2entry.db 데이터베이스 파일에서 항목을 읽고 고유 이름(DN)을 디스크상의 형식에서 메모리 내 형식으로 변환합니다. 캐시에 저장된 항목을 사용하면

서버가 디스크 I/O 및 변환 단계를 건너뛸 수 있습니다.

절차

1. 자동 캐시 튜닝을 비활성화합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend config set --cache-autosize=0
```

2. 접미사와 해당 백엔드를 표시합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com suffix list
dc=example,dc=com (userroot)
```

이 명령은 각 접미사 옆에 백엔드 데이터베이스의 이름을 표시합니다. 다음 단계에서 접미사의 데이터베이스 이름이 필요합니다.

3. 데이터베이스의 항목 캐시 크기를 바이트 단위로 설정합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend suffix set --cache-memsize=2147483648 userRoot
```

이 명령은 사용자Root 데이터베이스의 항목 캐시를 2GB로 설정합니다.

4. 인스턴스를 다시 시작합니다.

```
# dsctl instance_name restart
```

10.8. 웹 콘솔을 사용하여 항목 캐시 크기 설정

Directory Server는 항목 캐시를 사용하여 검색 및 읽기 작업 중에 사용되는 디렉터리 항목을 저장합니다. 항목 캐시를 모든 레코드를 저장할 수 있는 크기로 설정하면 검색 작업에 가장 큰 영향을 미칩니다.

항목 캐싱이 구성되지 않은 경우 Directory Server는 id2entry.db 데이터베이스 파일에서 항목을 읽고 고유 이름(DN)을 디스크상의 형식에서 메모리 내 형식으로 변환합니다. 캐시에 저장된 항목을 사용하면 서버가 디스크 I/O 및 변환 단계를 건너뛸 수 있습니다.

사전 요구 사항

- 웹 콘솔에서 인스턴스에 로그인되어 있습니다.

절차

1. **Database** → **Suffixes** → ***suffix_name*** → **Settings** 로 이동합니다.
2. 자동 캐시 튜닝 설정을 비활성화합니다.
3. **Save Configuration** 을 클릭합니다.
4. 오른쪽 상단에 있는 작업을 클릭하고 인스턴스 재시작 을 선택합니다.
5. **Database** → **Suffixes** → ***suffix_name*** → **Settings** 로 이동합니다.
6. '암호화 캐시 크기 필드에서 데이터베이스 캐시 크기를 설정합니다.
7. **Save Configuration** 을 클릭합니다.
8. 오른쪽 상단에 있는 작업을 클릭하고 인스턴스 재시작 을 선택합니다.

11장. 가져오기 성능 개선

매우 큰 특성 크기 또는 많은 항목이 가져오기 작업 중에 서버 성능에 부정적인 영향을 미칠 수 있습니다. 이 섹션에서는 가져오기 성능을 개선하기 위해 **Directory Server** 설정을 조정하는 방법에 대해 설명합니다.

11.1. 큰 특성 값을 사용하여 큰 데이터베이스 가져오기 및 가져오기를 위해 디렉터리 서버 튜닝

다음과 같은 작업에 가져오기 캐시 자동 크기 조정 기능을 사용합니다.

- 매우 큰 데이터베이스 가져오기
- 인증서 체인 또는 이미지를 저장하는 바이너리 속성과 같은 대규모 속성이 있는 데이터베이스 가져오기

참고

오프라인 가져오기는 온라인 가져오기보다 빠릅니다. 가능한 경우 오프라인 가져오기를 사용하는 것이 좋습니다.

nsslapd-import-cache-autosize 특성을 사용하여 가져오기 캐시 자동 크기 조정을 구성할 수 있습니다. 기본적으로 **Directory Server**에서는 **ldif2db** 작업에 대해서만 가져오기 캐시 자동 크기 조정을 활성화하고 가져오기 캐시에 사용 가능한 실제 메모리의 **50%**를 자동으로 할당합니다.

자세한 내용은 구성, 명령 및 파일 참조 문서의 **nsslapd-import-cache-autosize** 속성에 대한 설명을 참조하십시오.

12장. 서버 연결 관리 튜닝

nsslapd-numlisteners 속성은 디렉터리 서버가 설정된 연결을 모니터링하는 데 사용할 수 있는 리스너 스레드 수를 지정합니다. 특성 값을 늘리면 서버에서 많은 수의 클라이언트 연결이 발생할 때 응답 시간을 개선할 수 있습니다.

12.1. 명령줄을 사용하여 연결 리스너 스레드 수 관리

명령줄을 사용하여 연결 리스너 스레드 수를 관리할 수 있습니다. 기본값은 **1** 입니다.

절차

1. 연결 리스너 스레드 수를 나열합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config get nsslapd-numlisteners
```

2. 연결 리스너 스레드 수를 수정합니다.

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-numlisteners=4
```

3. 인스턴스를 다시 시작합니다.

```
# dsctl instance_name restart
```