Red Hat Enterprise Linux 7

SELinux 사용자 및 관리자 가이드

SELinux(Security-Enhanced Linux)의 기본 및 고급 구성

Last Updated: 2023-02-23

Red Hat Enterprise Linux 7 SELinux 사용자 및 관리자 가이드

SELinux(Security-Enhanced Linux)의 기본 및 고급 구성

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

법적 공지

초록

이 문서는 다음 두 부분으로 구성되어 있습니다. SELinux및 제한된 서비스 관리. 전자는 SELinux 기능을 사용하는 기본 원리와 원칙을 설명하고, 후자는 다양한 서비스를 설정하고 구성하는 실용적인 작업에 더욱 중점을 둡니다.

차례

I 부. SELINUX

이 문서에서는 SELinux(Security Enhanced Linux)가 작동하는 기본 및 원칙에 대해 설명합니다.

1장. 소개

SELinux(Security Enhanced Linux)는 추가 시스템 보안 계층을 제공합니다. SELinux는 기본적으로 질문에 대답합니다. "may <subject> do <action> to <object>"는 다음과 같습니다. "사용자의 홈 디렉토리에 있는 웹 서버 액세스 파일일 수 있습니까?".

DAC(Degretionary Access Control)라고 하는 사용자, 그룹 및 기타 권한을 기반으로 하는 표준 액세스 정책은 시스템 관리자가 로그 파일을 볼 수 있도록 특정 애플리케이션을 제한하는 것과 같은 포괄적이고 세분화된 보안 정책을 생성할 수 없습니다. 또한 다른 애플리케이션에서는 로그 파일에 새 데이터를 추가할 수 없습니다.

SELinux는 MAC(강제적 액세스 제어)를 구현합니다. 모든 프로세스 및 시스템 리소스에는 *SELinux 컨텍스트*라는 특수 보안 레이블이 있습니다. SELinux *레이블이라고도 하는 SELinux* 컨텍스트는 시스템 수준 세부 정보를 추상화하고 엔터티의 보안 속성에 중점을 두는 식별자입니다. 이렇게 하면 SELinux 정책에서 오브젝트를 참조하는 일관된 방법을 제공할 뿐만 아니라 다른 식별 방법에서 찾을 수 있는 모호성을 제거합니다. 예를 들어 파일은 바인드 마운트를 사용하는 시스템에서 여러 개의 유효한 경로 이름을 가질 수 있습니다.

SELinux 정책은 프로세스가 서로 상호 작용하는 방법과 다양한 시스템 리소스와 상호 작용하는 방법을 정의하는 일련의 규칙에서 이러한 컨텍스트를 사용합니다. 기본적으로 정책은 규칙에서 명시적으로 액세스 권한을 부여하지 않는 한 상호 작용을 허용하지 않습니다.

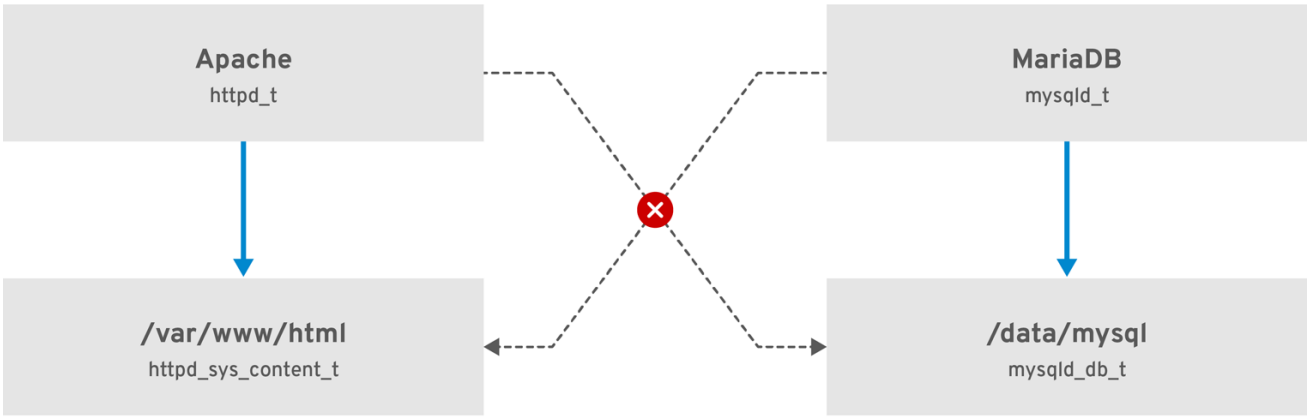
참고

DAC 규칙 다음에 SELinux 정책 규칙이 확인된다는 점을 기억하는 것이 중요합니다. DAC 규칙이 먼저 액세스를 거부하면 SELinux 정책 규칙이 사용되지 않습니다. 즉, 기존 DAC 규칙이 액세스를 차단하는 경우 SELinux 거부가 기록되지 않습니다.

SELinux 컨텍스트에는 사용자, 역할, 유형, 보안 수준 등 여러 필드가 있습니다. SELinux 유형 정보는 전체 SELinux 컨텍스트가 아니라 프로세스와 시스템 리소스 간에 허용되는 상호 작용을 정의하는 가장 일반적인 정책 규칙이므로 SELinux 정책과 관련하여 가장 중요한 정보일 수 있습니다. SELinux 유형은 일반적으로 **_t** 로 끝납니다. 예를 들어 웹 서버의 유형 이름은 **httpd_t** 입니다. **/var/www/html/** 에서 일반적으로 발견되는 파일 및 디렉토리의 유형 컨텍스트는 **httpd_sys_content_t** 입니다. /tmp 및 **/var/tmp/ 에서 일반적으로 발견되는 파일 및 디렉토리에 대한 유형 컨텍스트는 tmp_t 입니다**. 웹 서버 포트의 유형 컨텍스트는 **http_port_t** 입니다.

예를 들어 Apache(httpd **_t로 실행되는 웹 서버 프로세스)가 /var/www/html/ 및 기타 웹 서버 디렉터리 (httpd_sys_content_t**)에서 일반적으로 발견되는 컨텍스트의 파일 및 디렉터리에 액세스할 수 있도록 허용하는 정책 규칙이 있습니다. **/tmp 및 /var/tmp/ 에서 일반적으로 발견되는 파일에 대한 정책에 허용 규칙이 없으므로** 액세스가 허용되지 않습니다. SELinux를 사용하면 Apache가 손상되어 악의적인 스크립트에서 액세스를 제공하더라도 **/tmp** 디렉토리에 액세스할 수 없습니다.

그림 1.1. SELinux를 사용하면 httpd_t로 실행하는 Apache 프로세스가 /var/www/html/ 디렉터리에 액세스할 수 있으며 httpd_t 및 mysqld_db_t 유형 컨텍스트에 대한 허용 규칙이 없기 때문에 동일한 프로세스가 /data/mysql/ 디렉터리에 액세스할 수 있습니다. 반면 mysqld_t로 실행되는 MariaDB 프로세스는 /data/mysql/ 디렉터리에 액세스할 수 있으며 SELinux도 mysqld_t 유형의 프로세스를 올바르게 거부하여 httpd_sys_content_t로 레이블이 지정된 /var/www/html/ 디렉터리에 액세스합니다.



[D]

추가 리소스

자세한 내용은 다음 설명서를 참조하십시오.

- **apropos selinux 명령으로 나열된 selinux(8)** 도움말 페이지 및 도움말 페이지.
- selinux-policy-doc 패키지를 설치할 때 **man -k _selinux** 명령으로 나열된 도움말 페이지. 자세한 내용은 11.3.3절. "서비스 수동 페이지"을 참조하십시오.
- SELinux 컬러링북
- SELinux Wiki FAQ

1.1. SELINUX 실행의 이점

SELinux는 다음과 같은 이점을 제공합니다.

- 모든 프로세스와 파일에 레이블이 지정됩니다. SELinux 정책 규칙은 프로세스가 파일과 상호 작용하는 방법과 프로세스가 서로 상호 작용하는 방식을 정의합니다. 액세스는 특별히 허용하는 SELinux 정책 규칙이 있는 경우에만 허용됩니다.
- 세부적인 액세스 제어. 사용자 재량에 Linux 사용자 및 그룹 ID에 따라 제어되는 기존 UNIX 권한을 벗어나 SELinux 액세스 결정은 SELinux 사용자, 역할, 유형 및 선택적으로 보안 수준과 같은 사용 가능한 모든 정보를 기반으로 합니다.
- SELinux 정책은 관리 방식으로 정의되고 시스템 전체에 적용됩니다.
- 권한 에스컬레이션 공격에 대한 완화 개선. 프로세스는 도메인에서 실행되므로 서로 분리됩니다. SELinux 정책 규칙은 프로세스가 파일 및 기타 프로세스에 액세스하는 방법을 정의합니다. 프로세스가 손상되면 공격자는 해당 프로세스의 일반 기능에만 액세스할 수 있으며 프로세스가 액세스하도록 구성된 파일에만 액세스할 수 있습니다. 예를 들어 Apache HTTP 서버가 손상되면 공격자는 특정 SELinux 정책 규칙을 추가하거나 이러한 액세스를 허용하도록 구성하지 않은 한 해당 프로세스를 사용하여 사용자 홈 디렉토리의 파일을 읽을 수 없습니다.

- SELinux를 사용하여 데이터 기밀성 및 무결성을 적용하고 신뢰할 수 없는 입력에서 프로세스를 보호할 수 있습니다.

그러나 SELinux는 다음이 아닙니다.

- 확대/축소 소프트웨어,
- 암호, 방화벽 및 기타 보안 시스템 대신
- 올인원 보안 솔루션.

SELinux는 기존 보안 솔루션을 개선하도록 설계되었으며 이를 대체하지 않습니다. SELinux를 실행하는 경우에도 분석하기 어려운 암호 또는 방화벽을 사용하여 소프트웨어를 최신 상태로 유지하는 등 모범적인 보안 사례를 계속 따르는 것이 중요합니다.

1.2. 예

다음 예제에서는 SELinux가 보안을 강화하는 방법을 보여줍니다.

- 기본 작업은 deny입니다. 파일을 여는 프로세스와 같이 액세스를 허용하기 위한 SELinux 정책 규칙이 없으면 액세스가 거부됩니다.
- SELinux는 Linux 사용자를 제한할 수 있습니다. SELinux 정책에는 많은 제한된 SELinux 사용자가 있습니다. 제한된 SELinux 사용자에게 Linux 사용자를 매핑하여 보안 규칙 및 메커니즘을 활용할 수 있습니다. 예를 들어 Linux 사용자를 SELinux **user_u** 사용자에게 매핑하면 **sudo** 및 **su** 와 같이 사용자 ID(setuid) 애플리케이션을 실행할 수 없는 Linux 사용자가 생성됩니다. 자세한 내용은 3.3절. "제한된 사용자 및 제한되지 않은 사용자"을 참조하십시오.
- 프로세스 및 데이터 분리 증가. 프로세스는 자체 도메인에서 실행되므로 프로세스가 다른 프로세스에서 사용하는 파일에 액세스하지 못하도록 방지하고 프로세스가 다른 프로세스에 액세스하지 못하도록 합니다. 예를 들어, SELinux를 실행할 때 공격자는 Samba 서버를 손상시킬 수 없으며 MariaDB 데이터베이스와 같은 다른 프로세스에서 사용하는 파일을 읽고 쓸 수 있는 공격 벡터로 해당 Samba 서버를 사용할 수 없습니다.
- SELinux를 사용하면 구성 실수로 인한 손상을 완화할 수 있습니다. DNS(Domain Name System) 서버는 종종 영역 전송이라고 하는 항목에서 서로 간에 정보를 복제합니다. 공격자는 영역 전송을 사용하여 false 정보로 DNS 서버를 업데이트할 수 있습니다. Red Hat Enterprise Linux에서 Berkeley Internet Name Domain(BIND)을 DNS 서버로 실행하는 경우 관리자가 영역 전송을 수행할 수 있는 서버를 제한하지 않아도 기본 SELinux 정책은 영역 파일을 금지합니다. [1] 에서 영역 전송을 사용하여, 데몬 자체 **라는** BIND 및 기타 프로세스를 통해 업데이트할 수 있습니다.
- NetworkWorld.com 문서, 서버 소프트웨어에 대한 이해력: SELinux로 실제 공격 차단 [2], SELinux에 대한 배경 정보와 SELinux가 차단한 다양한 위협에 대한 정보.

1.3. SELINUX 아키텍처

SELinux는 Linux 커널에 빌드된 Linux 보안 모듈(LSM)입니다. 커널의 SELinux 하위 시스템은 관리자가 제어하고 부팅 시 로드하는 보안 정책에 의해 구동됩니다. 시스템의 모든 보안 관련 커널 수준 액세스 작업은 SELinux에서 가로채고 로드된 보안 정책 컨텍스트에서 검사합니다. 로드된 정책에서 작업을 허용하면 작업을 계속합니다. 그렇지 않으면 작업이 차단되고 프로세스가 오류를 수신합니다.

액세스 허용 또는 허용하지 않기와 같은 SELinux 결정은 캐시됩니다. 이 캐시를 AVC(액세스 벡터 캐시)라고 합니다. 이러한 캐시된 의사 결정을 사용할 때는 SELinux 정책 규칙을 보다 적게 확인하여 성능이 향상되어야 합니다. DAC 규칙이 먼저 액세스를 거부하면 SELinux 정책 규칙이 적용되지 않습니다.

1.4. SELINUX 상태 및 모드

SELinux는 비활성화, 허용 또는 강제의 세 가지 모드 중 하나로 실행될 수 있습니다.

비활성화 모드는 강력하지 않습니다. 시스템이 SELinux 정책을 적용하지 않을 뿐만 아니라 파일과 같은 영구 오브젝트의 레이블을 방지하여 나중에 SELinux를 활성화하기가 어렵습니다.

허용 모드에서 시스템은 SELinux가 개체에 레이블을 지정하고 로그에서 액세스 거부 항목을 내보내는 등 로드된 보안 정책을 적용하는 것처럼 작동하지만 실제로는 작업을 거부하지 않습니다. 프로덕션 시스템에는 권장되지 않지만 허용 모드는 SELinux 정책 개발에 유용할 수 있습니다.

강제 모드는 기본 및 권장되는 작업 모드입니다. 강제 모드에서 SELinux는 정상적으로 작동하여 전체 시스템에서 로드된 보안 정책을 적용합니다.

setenforce 유틸리티를 사용하여 강제 모드와 허용 모드 간에 변경합니다. **setenforce** 를 사용한 변경 사항은 재부팅 시 유지되지 않습니다. 강제 모드로 변경하려면 Linux root 사용자로 **setenforce 1** 명령을 입력합니다. 허용 모드로 변경하려면 **setenforce 0** 명령을 입력합니다. **getenforce** 유틸리티를 사용하여 현재 SELinux 모드를 확인합니다.

```
~]# getenforce
Enforcing
```

```
~]# setenforce 0
~]# getenforce
Permissive
```

```
~]# setenforce 1
~]# getenforce
Enforcing
```

Red Hat Enterprise Linux에서는 개별 도메인을 허용 모드로 설정할 수 있으며 강제 모드에서 시스템을 실행할 수 있습니다. 예를 들어 **httpd_t** 도메인을 허용하려면 다음을 수행합니다.

```
~]# semanage permissive -a httpd_t
```

자세한 내용은 11.3.4절. "허용 도메인"을 참조하십시오.

참고

영구 상태 및 모드 변경 사항은 4.4절. "SELinux 상태 및 모드의 영구 변경" 에서 다룹니다.

1.5. 추가 리소스

Red Hat IdM(Identity Management)은 SELinux 사용자 맵을 정의하는 중앙 집중식 솔루션을 제공합니다. 자세한 내용은 Linux 도메인 ID, 인증 및 정책 가이드에서 SELinux 사용자 맵 정의를 참조하십시오 .

---

[1] DNS 서버에서 사용하는 호스트 이름 및 IP 주소 매핑과 같은 정보를 포함하는 텍스트 파일.

[2] Marti, Don. "서버 소프트웨어에 대한 확신: SELinux는 실제 위협을 차단합니다.". 2008년 2월 24일 게시됨. 2009년 8월 27일 액세스: http://www.networkworld.com/article/2283723/lan-wan/a-seatbelt-for-server-software--selinux-blocks-real-world-exploits.html.

2장. SELINUX 컨텍스트

프로세스 및 파일에는 SELinux 사용자, 역할, 유형, 수준(선택 사항)과 같은 추가 정보가 포함된 SELinux 컨텍스트로 레이블이 지정됩니다. SELinux를 실행할 때 이 모든 정보를 사용하여 액세스 제어 결정을 내립니다. Red Hat Enterprise Linux에서 SELinux는 역할 기반 액세스 제어(RBAC),TE(유형 강제), 선택적으로 XML(Multi-Level Security)의 조합을 제공합니다.

다음은 SELinux 컨텍스트를 보여주는 예입니다. SELinux 컨텍스트는 SELinux를 실행하는 Linux 운영 체제의 프로세스, Linux 사용자 및 파일에서 사용됩니다. 다음 명령을 사용하여 파일 및 디렉터리의 SELinux 컨텍스트를 확인합니다.

```
~]$ ls -Z file1
-rwxrw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0      file1
```

SELinux 컨텍스트는 *SELinux 사용자:role:type:level* 구문을 따릅니다. 필드는 다음과 같습니다.

SELinux 사용자

SELinux 사용자 ID는 특정 역할 집합과 특정 MLS/MCS 범위에 대해 인증된 정책에 알려진 ID입니다. 각 Linux 사용자는 SELinux 정책을 사용하여 SELinux 사용자에게 매핑됩니다. 이를 통해 Linux 사용자는 SELinux 사용자에게 지정된 제한 사항을 상속할 수 있습니다. 매핑된 SELinux 사용자 ID는 입력할 수 있는 역할과 수준을 정의하기 위해 해당 세션의 프로세스에 대해 SELinux 컨텍스트에서 사용됩니다. SELinux 및 Linux 사용자 계정 간 매핑 목록을 보려면 root로 다음 명령을 입력합니다(policycoreutils-python 패키지가 설치되어 있어야 함).

```
~]# semanage login -l
Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

출력은 시스템마다 약간 다를 수 있습니다.

- **로그인 이름** 열에는 Linux 사용자가 나열됩니다.
- **SELinux User(SELinux 사용자)** 열에는 Linux 사용자가 에 매핑되는 SELinux 사용자가 나열됩니다. 프로세스의 경우 SELinux 사용자는 액세스할 수 있는 역할과 수준을 제한합니다.
- **MLS/MCS 범위** 열은 MLS(Multi-Level Security) 및 MCS(Multi-Category Security)에서 사용하는 수준입니다.
- **서비스** 열은 Linux 사용자가 시스템에 로그인해야 하는 올바른 SELinux 컨텍스트를 결정합니다. 기본적으로 별표(*) 문자가 사용되며 모든 서비스를 나타냅니다.

역할

SELinux의 일부는 역할 기반 액세스 제어(RBAC) 보안 모델입니다. 역할은 RBAC의 속성입니다. SELinux 사용자에게는 역할에 대한 권한이 부여되며, 도메인에 대한 역할이 인증됩니다. 역할은 도메인과 SELinux 사용자 간의 중간 역할을 합니다. 입력할 수 있는 역할은 입력할 수 있는 도메인을 결정합니다. 궁극적으로 액세스할 수 있는 오브젝트 유형을 제어합니다. 이를 통해 권한 상승 공격의 취약점을 줄일 수 있습니다.

type

유형은 Type Enforcement의 속성입니다. 유형은 프로세스의 도메인 및 파일의 유형을 정의합니다. SELinux 정책 규칙은 유형에 액세스하는 도메인이든 다른 도메인에 액세스하는 도메인이든 관계없이

유형이 서로 액세스할 수 있는 방법을 정의합니다. 액세스는 허용하는 특정 SELinux 정책 규칙이 있는 경우에만 허용됩니다.

level

수준은 MLS 및 MCS의 속성입니다. MLS 범위는 레벨이 다를 경우 *낮은 수준-고수준으로 작성되며 수준이 동일한 경우 낮은 수준* (**s0-s0은 s** 0)으로 작성됩니다 **.** 각 수준은 민감도 범주가 선택 사항인 민감도 범주 쌍입니다. 범주가 있는 경우 수준은 *민감도:category-set* 로 작성됩니다. 범주가 없으면 *민감도* 로 작성됩니다.

범주 세트가 연속된 시리즈인 경우 약어로 표시할 수 있습니다. 예를 들어 **c0.c3** 은 **c0,c1,c2,c3** 과 동일합니다. **/etc/selinux/targeted/setrans.conf** 파일은 수준(s**0:c0)을 사람이 읽을 수 있는 양식(즉, CompanyConfidential)**으로 매핑합니다. Red Hat Enterprise Linux에서는 대상 지정 정책은 MCS를 적용하며 MCS에서는 민감도 한 개만 존재합니다. **s0**. Red Hat Enterprise Linux의 MCS는 1024가지 카테고리(c **0** ~c **1023**)를 지원합니다.**s0-s0:c0.c1023** 은 민감도 **s0** 이며 모든 범주에 대해 인증되었습니다.

MLS는 LSPP(Labeled Security Protection Profile) 환경에서 사용됩니다. MLS 제한을 사용하려면 selinux-policy-mls 패키지를 설치하고 MLS를 기본 SELinux 정책으로 구성합니다. Red Hat Enterprise Linux와 함께 제공되는 MLS 정책은 평가된 구성의 일부가 아닌 여러 프로그램 도메인을 생략하므로 데스크탑 워크스테이션의 MLS를 사용할 수 없습니다(X Window System은 지원하지 않음). 그러나 모든 프로그램 도메인을 포함하는 업스트림 SELinux 참조 정책 의 MLS 정책을 구축할 수 있습니다. MLS 구성에 대한 자세한 내용은 4.13절. "다단계 보안(MLS)" 을 참조하십시오.

2.1. 도메인 전환

한 도메인의 프로세스는 새 도메인의 **진입점** 유형이 있는 애플리케이션을 실행하여 다른 도메인으로 전환합니다. **진입점** 권한은 SELinux 정책에서 사용되며 도메인을 입력하는 데 사용할 수 있는 애플리케이션을 제어합니다. 다음 예제에서는 도메인 전환을 보여줍니다.

절차 2.1. 도메인 전환의 예

1. 사용자가 암호를 변경하려고 합니다. 이렇게 하려면 **passwd** 유틸리티를 실행합니다. **/usr/bin/passwd** 실행 파일은 **passwd_exec_t 유형으로** 레이블이 지정됩니다.

```
~]$ ls -Z /usr/bin/passwd
-rwsr-xr-x  root root system_u:object_r:passwd_exec_t:s0 /usr/bin/passwd
```

passwd 유틸리티는 **shadow_t** 유형으로 레이블이 지정된 **/etc/shadow** 에 액세스합니다.

```
~]$ ls -Z /etc/shadow
-r--------. root root system_u:object_r:shadow_t:s0    /etc/shadow
```

2. SELinux 정책 규칙은 **passwd_t 도메인에서 실행되는 프로세스가 shadow_t** 유형으로 레이블이 지정된 파일을 읽고 쓸 수 있음을 나타냅니다. **shadow_t** 유형은 암호 변경에 필요한 파일에만 적용됩니다. 여기에는 **/etc/gshadow**,**/etc/shadow** 및 해당 백업 파일이 포함됩니다.

3. SELinux 정책 규칙은 passwd **_t 도메인에 passwd_ exec_t** 유형으로 설정된 **진입점** 권한이 있음을 나타냅니다.

4. 사용자가 **passwd** 유틸리티를 실행하면 사용자의 쉘 프로세스가 **passwd_t 도메인으로** 전환됩니다. SELinux에서는 기본 조치가 거부되고 **passwd_t 도메인에서 실행 중인 애플리케이션이 shadow_t** 유형으로 레이블이 지정된 파일에 액세스할 수 있도록 허용하는 규칙이 있으므로 **passwd** 애플리케이션에서 **/etc/shadow** 에 액세스하고 사용자 암호를 업데이트할 수 있습니다.

이 예제는 전체가 아니며 도메인 전환을 설명하는 데 기본 예제로 사용됩니다. **passwd_t 도메인에서 실행 중인 주체가 shadow_t** 파일 유형으로 레이블이 지정된 개체에 액세스할 수 있도록 하는 실제 규칙이 있지만, 제목이 새 도메인으로 전환되기 전에 다른 SELinux 정책 규칙을 충족해야 합니다. 이 예제에서는 적용 유형으로 다음을 보장합니다.

- **passwd_t 도메인은 passwd_ exec_t 유형으로 레이블이 지정된 애플리케이션을 실행하여 입력할 수 있습니다. 는 lib_t** 유형과 같은 권한 있는 공유 라이브러리에서만 실행할 수 있으며 다른 애플리케이션을 실행할 수 없습니다.

- **passwd_t와 같은 승인된 도메인만 shadow_t** 유형으로 레이블이 지정된 파일에 쓸 수 있습니다. 다른 프로세스가 수퍼유저 권한으로 실행 중인 경우에도 해당 프로세스는 **passwd_t 도메인에서 실행되지 않으므로 shadow_t 유형으로 레이블이 지정된 파일에 쓸 수 없습니다**.

- 인증된 도메인만 **passwd_t** 도메인으로 전환할 수 있습니다. 예를 들어 **sendmail _t 도메인에서 실행되는 sendmail** 프로세스에 passwd를 실행하는 합법적인 이유가 없으므로 **passwd _t** 도메인으로 전환할 수 없습니다.

- **passwd_t** 도메인에서 실행 중인 프로세스는 **etc_t 또는 shadow_t** 유형으로 레이블이 지정된 파일처럼 권한 있는 유형만 읽고 쓸 수 있습니다. 이렇게 하면 **passwd** 애플리케이션이 임의 파일을 읽거나 쓸 수 없습니다.

2.2. 프로세스의 SELINUX 컨텍스트

ps -eZ 명령을 사용하여 프로세스의 SELinux 컨텍스트를 확인합니다. 예를 들어 다음과 같습니다.

절차 2.2. passwd 유틸리티의 SELinux 컨텍스트 보기

1. **Applications → System Tools → Terminal(시스템 도구 터미널)과 같은 터미널**을 엽니다.

2. **passwd** 유틸리티를 실행합니다. 새 암호를 입력하지 마십시오.

```
~]$ passwd
Changing password for user user_name.
Changing password for user_name.
(current) UNIX password:
```

3. 새 탭 또는 다른 터미널을 열고 다음 명령을 입력합니다. 출력은 다음과 유사합니다.

```
~]$ ps -eZ | grep passwd
unconfined_u:unconfined_r:passwd_t:s0-s0:c0.c1023 13212 pts/1 00:00:00 passwd
```

4. 첫 번째 탭/터미널에서 **Ctrl+C** 를 눌러 **passwd** 유틸리티를 취소합니다.

이 예에서 **passwd** 유틸리티(passwd **_exec_t** 유형으로 레이블이 지정됨)가 실행되면 사용자의 쉘 프로세스가 **passwd_t 도메인으로** 전환됩니다. 유형은 프로세스의 도메인 및 파일의 유형을 정의합니다.

실행 중인 모든 프로세스에 대한 SELinux 컨텍스트를 보려면 **ps** 유틸리티를 다시 실행합니다. 아래에 출력이 잘린 예가 있으며 시스템에 따라 다를 수 있습니다.

```
]$ ps -eZ
system_u:system_r:dhcpc_t:s0              1869 ?  00:00:00 dhclient
system_u:system_r:sshd_t:s0-s0:c0.c1023  1882 ?  00:00:00 sshd
system_u:system_r:gpm_t:s0                1964 ?  00:00:00 gpm
```

```
system_u:system_r:crond_t:s0-s0:c0.c1023 1973 ?  00:00:00 crond
system_u:system_r:kerneloops_t:s0        1983 ?  00:00:05 kerneloops
system_u:system_r:crond_t:s0-s0:c0.c1023 1991 ?  00:00:00 atd
```

system_r 역할은 데몬과 같은 시스템 프로세스에 사용됩니다. Enforcement(강제)를 입력하고 각 도메인을 분리합니다.

2.3. 사용자의 SELINUX 컨텍스트

다음 명령을 사용하여 Linux 사용자와 연결된 SELinux 컨텍스트를 확인합니다.

```
~]$ id -Z
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Red Hat Enterprise Linux에서 Linux 사용자는 기본적으로 제한 없이 실행됩니다. 이 SELinux 컨텍스트는 Linux 사용자가 SELinux **unconfined_u** 사용자에 매핑되어 **unconfined_r** 역할로 실행되며 unconfined **_t 도메인에서 실행되고 있음을 보여줍니다. s0-s0** 은 MLS 범위이며, 이 경우 **s0** 과 동일합니다. 사용자가 액세스할 수 있는 카테고리는 모든 카테고리(c **0~c1023)인c0.c 1023**으로 정의됩니다.

3장. 대상 지정 정책

대상 지정 정책은 Red Hat Enterprise Linux에서 사용되는 기본 SELinux 정책입니다. 타겟 정책을 사용하는 경우 대상이 되는 프로세스는 제한된 도메인에서 실행되며, 타겟이 지정되지 않은 프로세스는 제한되지 않은 도메인에서 실행됩니다. 예를 들어, 기본적으로 로그인한 사용자는 **unconfined_t 도메인에서** 실행되며 init에 의해 시작된 시스템 프로세스는 **unconfined_service_t 도메인에서 실행됩니다**. 두 도메인은 모두 제한되지 않습니다.

실행 가능 및 쓰기 가능한 메모리 검사는 제한된 도메인과 제한되지 않은 도메인 모두에 적용될 수 있습니다. 그러나 기본적으로 제한되지 않은 도메인에서 실행 중인 주체는 쓰기 가능한 메모리를 할당하고 실행할 수 있습니다. 이러한 메모리 검사는 런타임 시 SELinux 정책을 수정할 수 있는 부울을 설정하여 활성화할 수 있습니다. 부울 구성은 나중에 설명합니다.

3.1. 제한된 프로세스

sshd 또는 **httpd** 와 같은 네트워크에서 수신 대기하는 거의 모든 서비스가 Red Hat Enterprise Linux에 제한되어 있습니다. 또한 root 사용자로 실행되고 사용자의 작업을 수행하는 대부분의 프로세스(예: **passwd** 유틸리티)가 제한됩니다. 프로세스가 제한된 경우 httpd **_t** 도메인에서 실행되는 **httpd** 프로세스 등 자체 도메인에서 실행됩니다. SELinux 정책 구성에 따라 공격자가 제한된 프로세스가 손상되면 공격자가 리소스에 대한 액세스와 가능한 손상을 제한합니다.

SELinux가 활성화되어 시스템이 다음 예제를 수행할 준비가 되었는지 확인하려면 다음 절차를 완료합니다.

절차 3.1. SELinux 상태를 확인하는 방법

1. SELinux가 활성화되어 있고 강제 모드로 실행 중이며 타겟 정책이 사용 중인지 확인합니다. 올바른 출력은 아래 출력과 유사해야 합니다.

```
~]$ sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Max kernel policy version:      30
```

   SELinux 모드 변경에 대한 자세한 내용은 4.4절. "SELinux 상태 및 모드의 영구 변경" 을 참조하십시오.

2. root로 **/var/www/html/** 디렉터리에 파일을 생성합니다.

```
~]# touch /var/www/html/testfile
```

3. 새로 생성된 파일의 SELinux 컨텍스트를 보려면 다음 명령을 입력합니다.

```
~]$ ls -Z /var/www/html/testfile
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 /var/www/html/testfile
```

   기본적으로 Linux 사용자는 Red Hat Enterprise Linux에서 제한 없이 실행되므로 **testfile** 파일에 SELinux **unconfined_u** 사용자로 레이블이 지정됩니다. RBAC는 파일이 아닌 프로세스에 사용됩

니다. 역할에는 파일에 대한 의미가 없습니다. **object_r** 역할은 영구저장장치 및 네트워크 파일 시스템에서 파일에 사용되는 일반 역할입니다. **/proc** 디렉터리에서 프로세스와 관련된 파일은 **system_r** 역할을 사용할 수 있습니다. **httpd_sys_content_t** 유형을 사용하면 **httpd** 프로세스가 이 파일에 액세스할 수 있습니다.

다음 예제에서는 SELinux가 Apache HTTP Server(httpd)가 Samba에서 사용할 파일과 같이 올바르게 레이블이 지정되지 않은 파일을 읽지 못하도록 하는 방법을 보여줍니다. 이는 예제이며 프로덕션에서 사용해서는 안 됩니다. httpd 및 wget 패키지가 설치되어 있고, SELinux 대상 지정 정책이 사용되며 SELinux가 강제 모드로 실행 중이라고 가정합니다.

절차 3.2. 제한된 프로세스의 예

1. root로 **httpd** 데몬을 시작합니다.

```
~]# systemctl start httpd.service
```

서비스가 실행 중인지 확인합니다. 출력에는 아래 정보가 포함되어야 합니다(시간 스탬프만 다릅니다).

```
~]$ systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: active (running) since Mon 2013-08-05 14:00:55 CEST; 8s ago
```

2. Linux 사용자가 에 대한 쓰기 액세스 권한이 있는 디렉터리로 변경하고 다음 명령을 입력합니다. 기본 구성을 변경하지 않는 한 이 명령은 성공합니다.

```
~]$ wget http://localhost/testfile
--2009-11-06 17:43:01--  http://localhost/testfile
Resolving localhost... 127.0.0.1
Connecting to localhost|127.0.0.1|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 0 [text/plain]
Saving to: `testfile'

[ <=>                                 ] 0      --.-K/s   in 0s

2009-11-06 17:43:01 (0.00 B/s) - `testfile' saved [0/0]
```

3. **chcon** 명령은 파일의 레이블을 다시 지정합니다. 그러나 파일 시스템에 레이블을 다시 지정하면 이러한 레이블 변경 사항이 유지되지 않습니다. 파일 시스템의 레이블 변경 후에도 유지되는 영구 변경의 경우 **semanage** 유틸리티를 사용합니다. 이 유틸리티는 나중에 설명합니다. root로 다음 명령을 입력하여 유형을 Samba에서 사용하는 유형으로 변경합니다.

```
~]# chcon -t samba_share_t /var/www/html/testfile
```

변경 사항을 보려면 다음 명령을 입력합니다.

```
~]$ ls -Z /var/www/html/testfile
-rw-r--r--  root root unconfined_u:object_r:samba_share_t:s0 /var/www/html/testfile
```

4. 현재 DAC 권한을 통해 **httpd** 프로세스에서 **테스트 파일에 액세스할 수 있습니다**. 사용자가 쓰기 액세스 권한이 있는 디렉터리로 변경하고 다음 명령을 입력합니다. 기본 구성을 변경하지 않는 한 이 명령은 실패합니다.

```
~]$ wget http://localhost/testfile
--2009-11-06 14:11:23--  http://localhost/testfile
Resolving localhost... 127.0.0.1
Connecting to localhost|127.0.0.1|:80... connected.
HTTP request sent, awaiting response... 403 Forbidden
2009-11-06 14:11:23 ERROR 403: Forbidden.
```

5. root 권한으로 **testfile** 을 제거합니다.

```
~]# rm -i /var/www/html/testfile
```

6. **httpd** 를 실행할 필요가 없는 경우 root로 다음 명령을 입력하여 중지합니다.

```
~]# systemctl stop httpd.service
```

이 예에서는 SELinux에서 추가된 추가 보안을 보여줍니다. DAC 규칙은 **httpd** 프로세스에서 액세스할 수 없는 유형으로 레이블이 지정되었기 때문에 2 **단계의 테스트** 파일에 **httpd** 프로세스 액세스를 허용했지만 SELinux에서 액세스를 거부했습니다.

auditd 데몬이 실행 중인 경우 다음과 유사한 오류가 **/var/log/audit/audit.log에 기록됩니다.**

```
type=AVC msg=audit(1220706212.937:70): avc:  denied  { getattr } for  pid=1904 comm="httpd"
path="/var/www/html/testfile" dev=sda5 ino=247576 scontext=unconfined_u:system_r:httpd_t:s0
tcontext=unconfined_u:object_r:samba_share_t:s0  tclass=file

type=SYSCALL msg=audit(1220706212.937:70): arch=40000003 syscall=196 success=no exit=-13
a0=b9e21da0 a1=bf9581dc a2=555ff4 a3=2008171 items=0 ppid=1902 pid=1904 auid=500 uid=48
gid=48 euid=48 suid=48 fsuid=48 egid=48 sgid=48 fsgid=48 tty=(none) ses=1 comm="httpd"
exe="/usr/sbin/httpd" subj=unconfined_u:system_r:httpd_t:s0 key=(null)
```

또한 다음과 유사한 오류가 **/var/log/httpd/error_log** 에 기록됩니다.

```
[Wed May 06 23:00:54 2009] [error] [client 127.0.0.1] (13)Permission denied: access to /testfile
denied
```

3.2. 제한되지 않은 프로세스

제한되지 않은 프로세스는 제한되지 않은 도메인에서 실행됩니다. 예를 들어 **init** 에서 실행한 제한되지 않은 서비스는 결국 **unconfined_service_t** 도메인, 커널에서 실행되는 제한되지 않은 서비스는 **kernel_t** 도메인에서 실행되고, unconfined Linux 사용자가 실행한 제한되지 않은 서비스는 결국 **unconfined_t 도메인에서 실행됩니다**. 제한되지 않은 프로세스의 경우 SELinux 정책 규칙이 적용되지만 제한되지 않은 도메인에서 실행되는 프로세스가 거의 모든 액세스를 허용하는 정책 규칙이 있습니다. 제한되지 않은 도메인에서 실행되는 프로세스는 DAC 규칙을 독점적으로 사용하는 것으로 대체됩니다. 제한되지 않은 프로세스가 손상되면 SELinux를 통해 공격자가 시스템 리소스 및 데이터에 대한 액세스 권한을 얻지 못하지만 물론 DAC 규칙이 계속 사용됩니다. SELinux는 DAC 규칙의 보안 개선 사항이므로 대체되지 않습니다.

SELinux가 활성화되고 시스템이 다음 예제를 수행할 준비가 되었는지 확인하려면 3.1절. "제한된 프로세스" 에 설명된 절차 3.1. "SELinux 상태를 확인하는 방법" 을(를) 완료합니다.

다음 예제에서는 제한 없이 실행할 때 Apache HTTP Server(**httpd**)가 Samba에서 사용하도록 의도한 데이터에 액세스하는 방법을 보여줍니다. Red Hat Enterprise Linux에서 **httpd** 프로세스는 기본적으로 제한된 **httpd_t** 도메인에서 실행됩니다. 이는 예제이며 프로덕션에서 사용해서는 안 됩니다. httpd,wget,dbus 및 audit 패키지가 설치되어 있고, SELinux 대상 지정 정책이 사용되며 SELinux가 강제 모드로 실행 중이라고 가정합니다.

절차 3.3. 제한되지 않은 프로세스의 예

1. **chcon** 명령은 파일의 레이블을 다시 지정합니다. 그러나 파일 시스템에 레이블을 다시 지정하면 이러한 레이블 변경 사항이 유지되지 않습니다. 파일 시스템의 레이블 변경 후에도 유지되는 영구 변경의 경우 **semanage** 유틸리티를 사용합니다. 이 유틸리티는 나중에 설명합니다. root 사용자로 다음 명령을 입력하여 유형을 Samba에서 사용하는 유형으로 변경합니다.

```
~]# chcon -t samba_share_t /var/www/html/testfile
```

변경 사항을 확인합니다.

```
~]$ ls -Z /var/www/html/testfile
-rw-r--r--  root root unconfined_u:object_r:samba_share_t:s0 /var/www/html/testfile
```

2. 다음 명령을 입력하여 **httpd** 프로세스가 실행 중이 아닌지 확인합니다.

```
~]$ systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: inactive (dead)
```

출력이 다르면 root로 다음 명령을 입력하여 **httpd** 프로세스를 중지합니다.

```
~]# systemctl stop httpd.service
```

3. **httpd** 프로세스를 제한 없이 실행하려면 root로 다음 명령을 입력하여 **/usr/sbin/httpd** 파일의 유형을 제한된 도메인으로 전환하지 않는 유형으로 변경합니다.

```
~]# chcon -t bin_t /usr/sbin/httpd
```

4. **/usr/sbin/httpd** 가 **bin_t** 유형으로 레이블이 지정되어 있는지 확인합니다.

```
~]$ ls -Z /usr/sbin/httpd
-rwxr-xr-x. root root system_u:object_r:bin_t:s0       /usr/sbin/httpd
```

5. root로 **httpd** 프로세스를 시작하고 성공적으로 시작되었는지 확인합니다.

```
~]# systemctl start httpd.service
```

```
~]# systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: active (running) since Thu 2013-08-15 11:17:01 CEST; 5s ago
```

6. 다음 명령을 입력하여 **unconfined_service_t** 도메인에서 실행 중인 **httpd** 를 확인합니다.

```
~]$ ps -eZ | grep httpd
system_u:system_r:unconfined_service_t:s0 11884 ? 00:00:00 httpd
system_u:system_r:unconfined_service_t:s0 11885 ? 00:00:00 httpd
system_u:system_r:unconfined_service_t:s0 11886 ? 00:00:00 httpd
system_u:system_r:unconfined_service_t:s0 11887 ? 00:00:00 httpd
system_u:system_r:unconfined_service_t:s0 11888 ? 00:00:00 httpd
system_u:system_r:unconfined_service_t:s0 11889 ? 00:00:00 httpd
```

7. Linux 사용자가 에 대한 쓰기 액세스 권한이 있는 디렉터리로 변경하고 다음 명령을 입력합니다. 기본 구성을 변경하지 않는 한 이 명령은 성공합니다.

```
~]$ wget http://localhost/testfile
--2009-05-07 01:41:10--  http://localhost/testfile
Resolving localhost... 127.0.0.1
Connecting to localhost|127.0.0.1|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 0 [text/plain]
Saving to: `testfile'

[ <=>                                 ]--.-K/s   in 0s

2009-05-07 01:41:10 (0.00 B/s) - `testfile' saved [0/0]
```

httpd 프로세스가 **samba_share_t** 유형으로 레이블이 지정된 파일에 액세스할 수 없지만 **httpd** 는 제한되지 않은 **unconfined_service_t** 도메인에서 실행되며 DAC 규칙 사용을 대체합니다. 따라서 **wget** 명령은 성공합니다. **httpd** 가 제한된 **httpd_t** 도메인에서 실행 중이면 **wget** 명령이 실패했습니다.

8. **restorecon** 유틸리티는 파일의 기본 SELinux 컨텍스트를 복원합니다. root로 **/usr/sbin/httpd** 의 기본 SELinux 컨텍스트를 복원하려면 다음 명령을 입력합니다.

```
~]# restorecon -v /usr/sbin/httpd
restorecon reset /usr/sbin/httpd context system_u:object_r:unconfined_exec_t:s0-
>system_u:object_r:httpd_exec_t:s0
```

/usr/sbin/httpd 에 **httpd_exec_t** 유형이 지정되었는지 확인합니다.

```
~]$ ls -Z /usr/sbin/httpd
-rwxr-xr-x  root root system_u:object_r:httpd_exec_t:s0 /usr/sbin/httpd
```

9. root로 다음 명령을 입력하여 **httpd** 를 다시 시작합니다. 재시작 후 httpd_t 도메인이 제한된 **httpd _t** 도메인에서 실행 중인지 확인합니다.

```
~]# systemctl restart httpd.service
```

```
~]$ ps -eZ | grep httpd
system_u:system_r:httpd_t:s0    8883 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    8884 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    8885 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    8886 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    8887 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    8888 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    8889 ?        00:00:00 httpd
```

10. root 권한으로 **testfile** 을 제거합니다.

```
~]# rm -i /var/www/html/testfile
rm: remove regular empty file `/var/www/html/testfile'? y
```

11. **httpd** 를 실행할 필요가 없는 경우 root로 다음 명령을 입력하여 **httpd** 를 중지합니다.

```
~]# systemctl stop httpd.service
```

이 섹션의 예제에서는 손상된 제한된 프로세스(SELinux로 보호)로부터 데이터를 보호할 수 있는 방법과 공격자가 손상된 제한 프로세스(SELinux에 의해 보호되지 않음)로부터 데이터를 더 효과적으로 액세스할 수 있는 방법을 보여줍니다.

3.3. 제한된 사용자 및 제한되지 않은 사용자

각 Linux 사용자는 SELinux 정책을 사용하여 SELinux 사용자에게 매핑됩니다. 이를 통해 Linux 사용자는 SELinux 사용자에 대한 제한을 상속할 수 있습니다. 이 Linux 사용자 매핑은 **semanage login -l** 명령을 root로 실행하면 표시됩니다.

```
~]# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

Red Hat Enterprise Linux에서 Linux 사용자는 기본적으로 SELinux **unconfined_u** 사용자에 매핑되는 SELinux **__default__** 로그인에 매핑됩니다. 다음 행은 기본 매핑을 정의합니다.

```
__default__             unconfined_u            s0-s0:c0.c1023
```

다음 절차에서는 새 Linux 사용자를 시스템에 추가하는 방법과 해당 사용자를 SELinux **unconfined_u** 사용자에게 매핑하는 방법을 보여줍니다. root 사용자가 Red Hat Enterprise Linux에서 기본적으로 수행되므로 제한되지 않은 것으로 가정합니다.

절차 3.4. 새 Linux 사용자를 SELinux unconfined_u 사용자에 매핑

1. root로 다음 명령을 입력하여 new **user라는 새** Linux 사용자를 생성합니다.

```
~]# useradd newuser
```

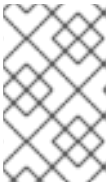
2. Linux **newuser** 사용자에게 암호를 할당하려면 다음을 수행합니다. root로 다음 명령을 입력합니다.

```
~]# passwd newuser
Changing password for user newuser.
New UNIX password: Enter a password
Retype new UNIX password: Enter the same password again
passwd: all authentication tokens updated successfully.
```

3. 현재 세션에서 로그아웃한 다음 Linux **newuser** 사용자로 로그인합니다. 로그인하면

pam_selinux PAM 모듈은 Linux 사용자를 SELinux 사용자(이 경우 **unconfined_u**)에 자동으로 매핑하고 결과 SELinux 컨텍스트를 설정합니다. 그런 다음 Linux 사용자의 쉘이 이 컨텍스트를 사용하여 시작됩니다. 다음 명령을 입력하여 Linux 사용자의 컨텍스트를 확인합니다.

```
[newuser@localhost ~]$ id -Z
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```



참고

시스템에서 **newuser** 사용자가 더 이상 필요하지 않은 경우 Linux **newuser** 의 세션에서 로그아웃한 후 계정으로 로그인한 다음 **userdel -r newuser** 명령을 root로 실행합니다. 그러면 홈 디렉터리와 함께 **newuser** 가 제거됩니다.

제한된 Linux 사용자 및 제한되지 않은 Linux 사용자는 실행 가능하고 쓰기 가능한 메모리 검사의 영향을 받으며 MCS 또는 MLS에도 제한됩니다.

사용 가능한 SELinux 사용자를 나열하려면 다음 명령을 입력합니다.

```
~]$seinfo -u
Users: 8
   sysadm_u
   system_u
   xguest_u
   root
   guest_u
   staff_u
   user_u
   unconfined_u
```

이러한 **info** 명령은 기본적으로 설치되지 않는 setools-console 패키지에서 제공합니다.

제한되지 않은 Linux 사용자가 SELinux 정책에서 **unconfined_t 도메인에서** 자체 제한된 도메인으로 전환할 수 있는 애플리케이션으로 정의하는 애플리케이션을 실행하는 경우 unconfined Linux 사용자는 여전히 제한된 도메인의 제한 사항을 받습니다. 이 경우의 보안 이점은 Linux 사용자가 제한되지 않고 있는 경우에도 애플리케이션이 제한되어 있다는 것입니다. 따라서 애플리케이션에서 결함을 악용하는 것은 정책에 의해 제한될 수 있습니다.

마찬가지로 제한된 사용자에게 이러한 검사를 적용할 수 있습니다. 제한된 각 Linux 사용자는 제한된 사용자 도메인에 의해 제한됩니다. SELinux 정책은 제한된 사용자 도메인에서 대상 제한 도메인으로의 전환을 정의할 수도 있습니다. 이러한 경우 제한된 Linux 사용자에게는 해당 대상 제한된 도메인의 제한 사항이 적용됩니다. 주요 점은 특별 권한이 역할에 따라 제한된 사용자와 연결되어 있다는 것입니다. 아래 표에서는 Red Hat Enterprise Linux의 Linux 사용자를 위한 기본 제한된 도메인의 예를 확인할 수 있습니다.

표 3.1. SELinux 사용자 기능

사용자	Role	도메인	X 윈도우 시스템	su 또는 sudo	홈 디렉토리 및 /tmp에서 실행 (기본값)	네트워킹
sysadm_u	sysadm_r	sysadm_t	제공됨	**su** 및 **sudo**	제공됨	제공됨
staff_u	staff_r	staff_t	제공됨	**sudo**만	제공됨	제공됨

사용자	Role	도메인	X 윈도우 시스템	su 또는 sudo	홈 디렉토리 및 /tmp에서 실행 (기본값)	네트워킹
user_u	user_r	user_t	제공됨	제공되지 않음	제공됨	제공됨
guest_u	guest_r	guest_t	제공되지 않음	제공되지 않음	제공됨	제공되지 않음
xguest_u	xguest_r	xguest_t	제공됨	제공되지 않음	제공됨	Firefox만

- **user_t,guest_t 및 xguest_t** 도메인의 Linux 사용자는 SELinux 정책에서 허용하는 경우에만 setuid(setuid) 애플리케이션만 실행할 수 있습니다(예: **passwd**). 이러한 사용자는 **su** 및 **sudo** setuid 애플리케이션을 실행할 수 없으므로 이러한 애플리케이션을 사용하여 루트가 될 수 없습니다.

- **sysadm_t,staff_t,user_t 및 xguest_t** 도메인의 Linux 사용자는 X Window 시스템 및 터미널을 사용하여 로그인할 수 있습니다.

- 기본적으로 **staff_t,user_t,guest_t** 도메인의 Linux 사용자는 홈 디렉터리 및 **/tmp** 에서 애플리케이션을 실행할 수 있습니다. **사용자의 권한을 상속하는 애플리케이션을 실행하지 못하도록 하려면 guest_exec_content 및 xguest_exec_content부울을 off 로 설정합니다. 이를 통해 결함이 있는 애플리케이션 또는 악성 애플리케이션이 사용자의 파일을 수정하지 못하도록 방지할 수 있습니다.**

  홈 디렉토리 및 /tmp 에서 사용자가 애플리케이션을 실행하지 못하도록 허용 및 방지하는 방법에 대한 정보는 6.6절. "사용자 애플리케이션 실행 부울" 을 참조하십시오.

- **xguest_t 도메인에 있는 유일한 네트워크 액세스 Linux 사용자는 Firefox 에서 웹 페이지에 연결하는 것입니다.**

system_u 는 시스템 프로세스 및 개체의 특수 사용자 ID입니다. Linux 사용자와 연결하면 안 됩니다. 또한 unconfined_u 및 root 는 제한되지 않은 사용자입니다. 이러한 이유로 앞서 언급한 SELinux 사용자 기능 테이블에 포함되지 않습니다.

이미 언급한 SELinux 사용자와 함께 해당 사용자에게 매핑할 수 있는 특별한 역할이 있습니다. 이러한 역할은 SELinux에서 사용자가 수행할 수 있는 작업을 결정합니다.

- **webadm_r 은 Apache HTTP 서버와 관련된 SELinux 유형만 관리할 수 있습니다. 자세한 내용은 13.2절. "유형" 을 참조하십시오.**

- **dbadm_r** 은 **MariaDB** 데이터베이스 및 **PostgreSQL** 데이터베이스 관리 시스템과 관련된 **SELinux** 유형만 관리할 수 있습니다. 자세한 내용은 20.2절. "유형" 및 21.2절. "유형" 을 참조하십시오.

- **logadm_r** 은 **syslog** 및 **auditlog** 프로세스와 관련된 **SELinux** 유형만 관리할 수 있습니다.

- **secadm_r** 은 **SELinux**만 관리할 수 있습니다.

- **auditadm_r** 은 감사 하위 시스템과 관련된 프로세스만 관리할 수 있습니다.

사용 가능한 역할을 모두 나열하려면 다음 명령을 입력합니다.

```
~]$ seinfo -r
```

앞에서 언급한 것처럼, 이러한**info** 명령은 기본적으로 설치되지 않는 **setools-console** 패키지에서 제공합니다.

3.3.1. sudo 전환 및 SELinux 역할

경우에 따라 제한된 사용자는 **root** 권한이 필요한 관리 작업을 수행해야 합니다. 그렇게 하려면 제한된 사용자가 **sudo** 명령을 사용하여 *제한된 관리자* **SELinux** 역할을 얻어야 합니다. **sudo** 명령은 신뢰할 수 있는 사용자에게 관리 액세스 권한을 제공하는 데 사용됩니다. 사용자가 **sudo** 를 사용하여 관리 명령에 우선하면 *자신의* 암호를 묻는 메시지가 표시됩니다. 그런 다음, 인증되고 명령이 허용되었다고 가정하면 **root** 사용자인 것처럼 관리 명령이 실행됩니다.

표 3.1. "SELinux 사용자 기능" 에 표시된 대로 **staff_u** 및 **sysadm_u SELinux** 제한 사용자만 기본적으로 **sudo** 를 사용할 수 있습니다. 이러한 사용자가 **sudo** 를 사용하여 명령을 실행하는 경우 **/etc/sudoers** 구성 파일에 지정된 규칙이나 해당 파일이 있는 경우 **/etc/ sudoers.d/** 디렉터리에 있는 해당 파일에 따라 역할을 변경할 수 있습니다.

sudo 에 대한 자세한 내용은 **Red Hat Enterprise Linux 7** 시스템 관리자 가이드의 *권한 얻기* 섹션을 참조하십시오.

절차 3.5. sudo 전환 구성

다음 절차에서는 새로 생성된 ***SELinux_user_u*** 제한된 사용자를 ***default_role_r*** 에서 관리자_r *관리*

자 역할로 전환하기 위해 **sudo** 를 설정하는 방법을 보여줍니다.

참고

기존 **SELinux** 사용자에 대해 제한된 관리자 역할을 구성하려면 처음 두 단계를 건너뜁니다.

1. 새 **SELinux** 사용자를 생성하고 이 사용자에 대한 기본 **SELinux** 역할 및 제한된 관리자 역할을 지정합니다.

```
~]# semanage user -a -r s0-s0:c0.c1023 -R "default_role_r administrator_r" SELinux_user_u
```

2. 기본 **SElinux** 정책 컨텍스트 파일을 설정합니다. 예를 들어 **staff_u SELinux** 사용자와 동일한 **SELinux** 규칙을 보유하려면 **staff_u** 컨텍스트 파일을 복사합니다.

```
~]# cp /etc/selinux/targeted/contexts/users/staff_u
/etc/selinux/targeted/contexts/users/SELinux_user_u
```

3. 새로 생성된 **SELinux** 사용자를 기존 **Linux** 사용자에게 매핑합니다.

```
semanage login -a -s SELinux_user_u -rs0:c0.c1023 linux_user
```

4. **/etc/sudoers.d/** 디렉토리에 **Linux** 사용자와 이름이 같은 새 구성 파일을 만들고 다음 문자열을 추가합니다.

```
~]# echo "linux_user ALL=(ALL) TYPE=administrator_t ROLE=administrator_r /bin/bash " >
/etc/sudoers.d/linux_user
```

5. **restorecon** 유틸리티를 사용하여 ***linux_user*** 홈 디렉터리에 레이블을 다시 지정합니다.

```
~]# restorecon -FR -v /home/linux_user
```

6. 새로 생성된 **Linux** 사용자로 시스템에 로그인하고 사용자가 기본 **SELinux** 역할로 레이블이 지정되어 있는지 확인합니다.

```
~]$ id -Z
SELinux_user_u:default_role_r:SELinux_user_t:s0:c0.c1023
```

7.

sudo 를 실행하여 사용자의 **SELinux** 컨텍스트를 **/etc/sudoers.d/*linux_user*** 에 지정된 대로 보조 **SELinux** 역할로 변경합니다**. sudo** 와 함께 사용되는 **-i** 옵션을 사용하면 대화형 쉘이 실행됩니다**.**

```
~]$ sudo -i
~]# id -Z
SELinux_user_u:administrator_r:administrator_t:s0:c0.c1023
```

default_role_r 또는 ***administrator_r*** 와 같은 자리 표시자를 더 잘 이해하려면 다음 예제를 참조하십시오**.**

예 3.1. sudo 전환 구성

이 예제에서는 기본 역할 **staff _ r** 이 할당된 새 **SELinux** 사용자를 만들고 **sudo** 를 사용하여 **staff_r** 에서 **webadm _r** 로 제한됨**_u** 의 역할을 변경하도록 구성합니다**.**

- **tekton _r** 또는 **unconfined_r** 역할에 **root** 사용자로 다음 명령을 모두 입력합니다**.**

```
~]# semanage user -a -r s0-s0:c0.c1023 -R "staff_r webadm_r" confined_u
~]# cp /etc/selinux/targeted/contexts/users/staff_u
/etc/selinux/targeted/contexts/users/confined_u
~]# semanage login -a -s confined_u -rs0:c0.c1023 linux_user
~]# restorecon -FR -v /home/linux_user
~]# echo "linux_user ALL=(ALL) ROLE=webadm_r TYPE=webadm_t /bin/bash " >
/etc/sudoers.d/linux_user
```

- 새로 생성된 **Linux** 사용자로 시스템에 로그인하고 사용자가 기본 **SELinux** 역할로 레이블이 지정되어 있는지 확인합니다**.**

```
~]$ id -Z
confined_u:staff_r:staff_t:s0:c0.c1023
~]$ sudo -i
~]# id -Z
confined_u:webadm_r:webadm_t:s0:c0.c1023
```

4장. SELINUX 작업

다음 섹션에서는 Red Hat Enterprise Linux의 기본 SELinux 패키지, 패키지 설치 및 업데이트, 로그 파일, 기본 SELinux 구성 파일, SELinux 모드 활성화 및 비활성화, 부울 구성, 파일 및 디렉터리 레이블의 일시적 변경, NFS 볼륨 마운트, 파일 및 디렉터리 복사 및 보관 시 SELinux 컨텍스트 보존 방법에 대해 간략하게 설명합니다.

4.1. SELINUX 패키지

Red Hat Enterprise Linux 전체 설치에서 SELinux 패키지는 설치 중에 수동으로 제외되지 않는 한 기본적으로 설치됩니다. 텍스트 모드에서 최소 설치를 수행하는 경우 policycoreutils-python 및 policycoreutils-gui 패키지가 기본적으로 설치되지 않습니다. 또한 기본적으로 SELinux는 강제 모드로 실행되며 SELinux 대상 지정 정책이 사용됩니다. 기본적으로 다음 SELinux 패키지가 시스템에 설치되어 있습니다.

- policycoreutils 는 SELinux를 운영 및 관리하기 위해 restorecon,secon,setfiles,semodule,load_policy 및 setsebool 과 같은 유틸리티를 제공합니다.
- selinux-policy 는 기본 디렉터리 구조, selinux-policy.conf 파일 및 RPM 매크로를 제공합니다.
- selinux-policy-targeted 는 SELinux 대상 정책을 제공합니다.
- libselinux - SELinux 애플리케이션을 위한 API를 제공합니다.
- libselinux-utils 는 avcstat,getenforce,getsebool,matchpathcon,selinuxconlist,selinuxdefcon,selinuxenabled 및 setenforce 유틸리티를 제공합니다.
- libselinux-python 은 SELinux 애플리케이션을 개발하기 위한 Python 바인딩을 제공합니다.

다음 패키지는 기본적으로 설치되지 않지만 yum install *<package-name>* 명령을 실행하여 선택적으로 설치할 수 있습니다.

-

selinux-policy-devel 은 사용자 지정 **SELinux** 정책 및 정책 모듈을 생성하는 유틸리티를 제공합니다.

- **selinux-policy-doc** 는 **SELinux**를 다양한 서비스와 함께 구성하는 방법을 설명하는 도움말 페이지를 제공합니다.
- **selinux-policy-✓s** 는 **MLS(Multi-Level Security) SELinux** 정책을 제공합니다.
- **setroubleshoot-server** 는 **SELinux**에서 액세스를 거부할 때 생성되는 거부 메시지를 이 패키지에서도 제공하는 **sealert** 유틸리티에서 볼 수 있는 자세한 설명으로 변환합니다.
- **setools-console** 은 **Tresys Technology SETools** 배포, 정책 분석 및 쿼리, 감사 로그 모니터링 및 보고, 파일 컨텍스트 관리를 위한 다양한 유틸리티 및 라이브러리를 제공합니다. **setools** 패키지는 **SETools**의 메타 패키지입니다. **setools-gui** 패키지는 **apol** 및 **seaudit** 유틸리티를 제공합니다. **setools-console** 패키지는 **sechecker,sediff,seinfo,sesearch, findcon** 명령줄 유틸리티를 제공합니다. 이러한 유틸리티에 대한 자세한 내용은 **Tresys Technology SETools** 페이지를 참조하십시오. **setools** 및 **setools-gui** 패키지는 **Red Hat Network Optional** 채널이 활성화된 경우에만 사용할 수 있습니다. 자세한 내용은 적용 범위 상세 정보를 참조하십시오.
- **mcstrans** 은 **s0-s0:c0.c1023** 과 같은 수준을 **SystemLow-SystemHigh** 와 같이 읽기 쉬운 형태로 변환합니다.
- **policycoreutils-python** 은 **SELinux**의 운영 및 관리를 위한 **semanage,audit2allow, audit2why** 및 **chcat** 과 같은 유틸리티를 제공합니다.
- **policycoreutils-gui** 는 **SELinux**를 관리하기 위한 그래픽 유틸리티인 **system-config-selinux** 를 제공합니다.

4.2. 사용된 로그 파일은 무엇입니까?

Red Hat Enterprise Linux에서는 기본 패키지 선택에서 제거되지 않는 한 **dbus** 및 **audit** 패키지가 기본적으로 설치됩니다. **Yum**을 사용하여 **setroubleshoot-server** 를 설치해야 합니다(**yum install setroubleshoot-server** 명령 사용).

auditd 데몬이 실행 중인 경우 다음과 같은 **SELinux** 거부 메시지가 기본적으로 **/var/log/audit/audit.log** 에 기록됩니다.

```
type=AVC msg=audit(1223024155.684:49): avc:  denied  { getattr } for  pid=2000 comm="httpd"
path="/var/www/html/file1" dev=dm-0 ino=399185 scontext=unconfined_u:system_r:httpd_t:s0
tcontext=system_u:object_r:samba_share_t:s0 tclass=file
```

또한 **/var/log/message** 파일에 다음과 유사한 메시지가 작성됩니다.

```
May 7 18:55:56 localhost setroubleshoot: SELinux is preventing httpd (httpd_t) "getattr" to
/var/www/html/file1 (samba_share_t). For complete SELinux messages. run sealert -l de7e30d6-
5488-466d-a606-92c9f40d316d
```

Red Hat Enterprise Linux 7에서 **setroubleshootd** 는 더 이상 서비스로 지속적으로 실행되지 않습니다. 그러나 여전히 **AVC** 메시지를 분석하는 데 사용됩니다. 두 개의 새 프로그램이 필요할 때 **setroubleshoot** 를 시작하는 방법으로 작동합니다.

- **sedispatch** 유틸리티는 감사 하위 시스템의 일부로 실행됩니다. **AVC** 거부 메시지가 반환되면 **sedispatch** 는 **dbus** 를 사용하여 메시지를 보냅니다. 이미 실행 중인 경우 이 메시지는 **setroubleshootd** 로 바로 이동합니다. 실행되고 있지 않으면 **sedispatch** 가 자동으로 시작됩니다.

- **seapplet** 유틸리티는 시스템 도구 모음에서 실행되며 **setroubleshootd** 의 **dbus** 메시지를 기다립니다. 알림 도구가 시작되어 사용자가 **AVC** 메시지를 검토할 수 있습니다.

절차 4.1. 자동으로 데몬 시작

1. 부팅 시 자동으로 시작되도록 **auditd** 및 **rsyslog** 데몬을 구성하려면 **root** 사용자로 다음 명령을 입력합니다.

```
~]# systemctl enable auditd.service
```

```
~]# systemctl enable rsyslog.service
```

2. 데몬이 활성화되었는지 확인하려면 쉘 프롬프트에 다음 명령을 입력합니다.

```
~]$ systemctl is-enabled auditd
enabled
```

```
~]$ systemctl is-enabled rsyslog
enabled
```

- 또는 **systemctl status *service-name*.service** 명령을 사용하고 명령 출력에서 활성화된 키워드를 검색합니다. 예를 들면 다음과 같습니다.

```
~]$ systemctl status auditd.service | grep enabled
auditd.service - Security Auditing Service
   Loaded: loaded (/usr/lib/systemd/system/auditd.service; enabled)
```

systemd 데몬이 시스템 서비스를 관리하는 방법에 대한 자세한 내용은 시스템 관리자 가이드의 시스템 서비스 관리 장을 참조하십시오.

4.3. 기본 설정 파일

/etc/selinux/config 파일은 기본 **SELinux** 구성 파일입니다. **SELinux**가 활성화되었는지 또는 비활성화되었는지 여부와 사용되는 **SELinux** 모드 및 **SELinux** 정책을 제어합니다.

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of these two values:
#       targeted - Targeted processes are protected,
#       mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

SELINUX=

SELINUX 옵션은 **SELinux**가 비활성화되었는지 여부와 강제 또는 허용 모드(강제 또는 허용)로 설정합니다.

- **SELINUX=enforcing** 을 사용하면 **SELinux** 정책이 적용되고 **SELinux**는 **SELinux** 정책 규칙을 기반으로 액세스를 거부합니다. 거부 메시지가 기록됩니다.
- **SELINUX=permissive** 를 사용하면 **SELinux** 정책이 적용되지 않습니다. **SELinux**는 액세스를 거부하지 않지만 강제 모드에서 **SELinux**를 실행하는 경우 거부된 작업에 대해 거부된 작업에 로깅됩니다.
- **SELINUX=disabled** 를 사용하면 **SELinux**가 비활성화되고, **SELinux** 모듈이 **Linux** 커널에 등록되지 않으며 **DAC** 규칙만 사용됩니다.

SELINUXTYPE=

SELINUXTYPE 옵션은 사용할 **SELinux** 정책을 설정합니다. 타겟 정책은 기본 정책입니다. **MLS** 정책을 사용하려면 이 옵션만 변경하십시오. **MLS** 정책을 활성화하는 방법에 대한 자세한 내용은 **4.13.2절. "SELinux에서 MLS 활성화"** 을 참조하십시오.

4.4. SELINUX 상태 및 모드의 영구 변경

1.4절. "SELinux 상태 및 모드" 에서 설명한 대로 **SELinux**를 활성화하거나 비활성화할 수 있습니다. 활성화되는 경우 **SELinux**에는 강제 및 허용 모드의 두 가지 모드가 있습니다.

getenforce 또는 **sestatus** 명령을 사용하여 **SELinux**가 실행 중인 모드를 확인합니다. **getenforce** 명령은 **Enforcing,Permissive** 또는 **Disabled** 를 반환합니다.

sestatus 명령은 **SELinux** 상태 및 사용 중인 **SELinux** 정책을 반환합니다.

```
~]$ sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Max kernel policy version:      30
```

참고

시스템이 허용 모드에서 **SELinux**를 실행하는 경우 사용자는 파일에 잘못 레이블을 지정할 수 있습니다. **SELinux**가 비활성화된 상태에서 생성된 파일에는 레이블이 전혀 지정되지 않습니다. 이 동작은 파일에 잘못 레이블되거나 레이블이 전혀 레이블되지 않았으므로 강제 모드로 변경할 때 문제가 발생합니다. 레이블이 잘못 지정되고 레이블이 지정되지 않은 파일의 문제가 발생하지 않도록 하려면 비활성화 상태에서 허용 또는 강제 모드로 변경할 때 파일 시스템에 자동으로 레이블이 다시 지정됩니다.

4.4.1. SELinux 활성화

활성화되면 SELinux를 강제 또는 허용 모드의 두 가지 모드 중 하나로 실행할 수 있습니다. 다음 섹션에서는 이러한 모드로 영구적으로 변경하는 방법을 보여줍니다.

이전에 비활성화한 시스템에서 SELinux를 활성화하는 동안 시스템 부팅 또는 프로세스 실패와 같은 문제를 방지하려면 다음 절차를 따르는 것이 좋습니다.

1. 허용 모드에서 SELinux를 활성화합니다. 자세한 내용은 4.4.1.1절. "허용 모드" 의 내용을 참조하십시오.

2. 시스템을 재부팅합니다.

3. SELinux 거부 메시지가 있는지 확인합니다. 자세한 내용은 11.3.5절. "거부 검색 및 보기" 의 내용을 참조하십시오.

4. 거부가 없는 경우 강제 모드로 전환합니다. 자세한 내용은 4.4.1.2절. "강제 모드" 의 내용을 참조하십시오.

SELinux를 강제 모드로 사용자 지정 애플리케이션을 실행하려면 다음 시나리오 중 하나를 선택하십시오.

- unconfined_service_t 도메인에서 애플리케이션을 실행합니다. 자세한 내용은 3.2절. "제한되지 않은 프로세스"을 참조하십시오.

- 애플리케이션에 대한 새 정책을 작성합니다. 자세한 내용은 사용자 지정 SELinux 정책 지식 베이스 작성 문서를 참조하십시오.

4.4.1.1. 허용 모드

SELinux가 허용 모드로 실행 중이면 SELinux 정책이 적용되지 않습니다. 시스템이 작동 상태로 남아 있으며 SELinux는 작업을 거부하지 않고 AVC 메시지만 기록합니다. 그러면 문제 해결, 디버깅 및 SELinux 정책 개선에 사용할 수 있습니다. 각 AVC는 이 경우 한 번만 로깅됩니다.

모드를 허용으로 영구적으로 변경하려면 다음 절차를 따르십시오.

절차 4.2. 허용 모드로 변경

1. 다음과 같이 **/etc/selinux/config** 파일을 편집합니다.

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
SELINUX=permissive
# SELINUXTYPE= can take one of these two values:
#       targeted - Targeted processes are protected,
#       mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

2. 시스템을 재부팅합니다.

```
~]# reboot
```

4.4.1.2. 강제 모드

SELinux가 강제 모드에서 실행 중인 경우 **SELinux** 정책을 적용하고 **SELinux** 정책 규칙에 따라 액세스를 거부합니다. **Red Hat Enterprise Linux**에서는 **SELinux**를 사용하여 처음 시스템을 설치할 때 강제 모드가 기본적으로 활성화됩니다.

SELinux가 비활성화된 경우 아래 절차에 따라 모드를 강제 모드로 다시 변경합니다.

절차 4.3. 강제 모드로 변경

이 절차에서는 **selinux-policy-targeted,selinux- policy,libselinux,libselinux -python,libselinux-utils,policycoreutils** 및 **policycoreutils-python** 패키지가 설치되어 있다고 가정합니다. 패키지가 설치되었는지 확인하려면 다음 명령을 사용합니다.

```
rpm -q package_name
```

1. 다음과 같이 **/etc/selinux/config** 파일을 편집합니다.

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of these two values:
```

```
#       targeted - Targeted processes are protected,
#       mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

2. 시스템을 재부팅합니다.

```
~]# reboot
```

다음 부팅 시 **SELinux**는 시스템 내의 모든 파일과 디렉터리의 레이블을 다시 지정하고 **SELinux**가 비활성화될 때 생성된 파일과 디렉토리에 대해 **SELinux** 컨텍스트를 추가합니다.

참고

강제 모드로 변경한 후 **SELinux**는 올바르지 않거나 누락된 **SELinux** 정책 규칙으로 인해 일부 작업을 거부할 수 있습니다. **SELinux** 거부 작업을 보려면 **root**로 다음 명령을 입력합니다.

```
~]# ausearch -m AVC,USER_AVC,SELINUX_ERR -ts today
```

또는 **setroubleshoot-server** 패키지가 설치된 상태에서 **root**로 다음 명령을 입력합니다.

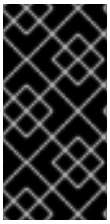
```
~]# grep "SELinux is preventing" /var/log/messages
```

SELinux에서 일부 작업을 거부하는 경우 문제 해결에 대한 정보는 11장. *문제 해결*을 참조하십시오.

모드의 일시적인 변경 사항은 1.4절. “SELinux 상태 및 모드” 에서 다룹니다.

4.4.2. SELinux 비활성화

SELinux가 비활성화되면 **SELinux** 정책이 전혀 로드되지 않습니다. 강제 적용되지 않으며 **AVC** 메시지가 기록되지 않습니다. 따라서 1.1절. “SELinux 실행의 이점” 에 나열된 **SELinux**를 실행하는 모든 이점이 손실됩니다.



중요

Red Hat은 **SELinux**를 영구적으로 비활성화하는 대신 허용 모드를 사용하도록 권장합니다. 허용 모드에 대한 자세한 내용은 **4.4.1.1**절. “허용 모드” 을 참조하십시오.

SELinux를 영구적으로 비활성화하려면 다음 절차를 따르십시오.

절차 **4.4. SELinux** 비활성화

1. **/etc/selinux/config** 파일에서 **SELINUX=disabled** 를 구성합니다.

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
SELINUX=disabled
# SELINUXTYPE= can take one of these two values:
#       targeted - Targeted processes are protected,
#       mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

2. 시스템을 재부팅합니다. 재부팅 후 **getenforce** 명령이 **Disabled** 를 반환하는지 확인합니다.

```
~]$ getenforce
Disabled
```

4.5. 부팅 시 SELINUX 모드 변경

부팅 시 **SELinux** 실행 방식을 변경하기 위해 여러 커널 매개변수를 설정할 수 있습니다.

enforcing=0

이 매개 변수를 설정하면 시스템이 허용 모드로 시작되므로 문제를 해결할 때 유용합니다. 파일 시스템이 너무 손상된 경우 허용 모드를 사용하는 것이 문제를 감지하는 유일한 옵션일 수 있습니다. 또한 허용 모드에서 시스템은 라벨을 올바르게 만듭니다. 이 모드에서 생성되는 **AVC** 메시지는 강제 모드에서와 다를 수 있습니다.

허용 모드에서는 일련의 동일한 거부 중 첫 번째 거부만 보고됩니다. 그러나 강제 모드에서는 디

렉터리 읽기와 관련된 거부가 발생할 수 있으며 애플리케이션이 중지될 수 있습니다. 허용 모드에서는 동일한 **AVC** 메시지가 표시되지만 애플리케이션은 디렉토리에 있는 파일을 계속 읽고 각 거부에 대해 **AVC**도 받습니다.

selinux=0

이 매개 변수를 사용하면 커널이 **SELinux** 인프라의 일부를 로드하지 않습니다. **init** 스크립트는 시스템이 **selinux=0** 매개 변수로 부팅되었으며 **/.autorelabel** 파일을 터치합니다. 이로 인해 **SELinux**를 활성화한 다음 부팅할 때 시스템이 자동으로 레이블을 다시 지정합니다.

중요

Red Hat은 **selinux=0** 매개 변수를 사용하지 않는 것이 좋습니다. 시스템을 디버깅하려면 허용 모드를 사용하는 것이 좋습니다.

autorelabel=1

이 매개변수는 시스템에서 다음 명령과 유사하게 레이블을 다시 지정하도록 강제 적용합니다.

```
~]# touch /.autorelabel
~]# reboot
```

시스템 레이블링에 많은 양의 오류가 포함된 경우 자동 레이블에 성공하기 위해 허용 모드로 부팅해야 할 수 있습니다.

checkreqprot 와 같은 추가 **SELinux** 관련 커널 부팅 매개 변수는 **/usr/share/doc/kernel-doc-*<KERNEL_VER>*/Documentation/kernel-parameters.txt** 파일을 참조하십시오. 이 문서는 **kernel-doc** 패키지와 함께 설치됩니다. ***<KERNEL_VER>*** 문자열을 설치된 커널의 버전 번호로 바꿉니다. 예를 들면 다음과 같습니다.

```
~]# yum install kernel-doc
~]$ less /usr/share/doc/kernel-doc-3.10.0/Documentation/kernel-parameters.txt
```

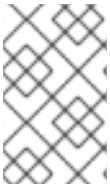
4.6. 부울

부울을 사용하면 **SELinux** 정책 작성에 대한 지식 없이 런타임 시 **SELinux** 정책 부분을 변경할 수 있습니다. 이렇게 하면 **SELinux** 정책을 다시 로드하거나 다시 컴파일하지 않고도 **NFS** 볼륨에 서비스 액세스 허용 등의 변경 사항을 허용합니다.

4.6.1. 부울 나열

부울 목록의 경우 각 항목에 대한 설명과 **on** 또는 **off** 여부에 대한 설명은 **Linux root** 사용자로 **semanage boolean -l** 명령을 실행합니다. 다음 예제는 모든 부울을 나열하지 않으며 출력은 간결성을 위해 단축됩니다.

```
~]# semanage boolean -l
SELinux boolean                 State  Default Description


smartmon_3ware                  (off  ,  off)  Determine whether smartmon can...
mpd_enable_homedirs             (off  ,  off)  Determine whether mpd can traverse...
```



참고

자세한 설명을 보려면 **selinux-policy-devel** 패키지를 설치합니다.

SELinux 부울 열에는 부울 이름이 나열됩니다. **Description** (설명) 열에는 부울이 켜져 있는지 또는 꺼져 있는지 여부와 해당 작업이 나열됩니다.

getsebool -a 명령은 **on** 또는 **off** 여부에 관계없이 부울을 나열하지만 각각에 대한 설명은 제공하지 않습니다. 다음 예제는 모든 부울을 나열하지 않습니다.

```
~]$ getsebool -a
cvs_read_shadow --> off
daemons_dump_core --> on
```

getsebool *boolean-name* 명령을 실행하여 **boolean *-name* *부울*** 의 상태만 나열합니다.

```
~]$ getsebool cvs_read_shadow
cvs_read_shadow --> off
```

공백으로 구분된 목록을 사용하여 여러 부울을 나열합니다.

```
~]$ getsebool cvs_read_shadow daemons_dump_core
cvs_read_shadow --> off
daemons_dump_core --> on
```

4.6.2. 부울 구성

setsebool 유틸리티를 **setsebool *boolean_name* on/off** 양식에서 실행하여 부울을 활성화하거나 비활성화합니다.

다음 예제에서는 **httpd_can_network_connect_db** 부울 구성을 보여줍니다.

절차 4.5. 부울 구성

1. 기본적으로 **httpd_can_network_connect_db** 부울이 꺼서 **Apache HTTP** 서버 스크립트 및 모듈이 데이터베이스 서버에 연결되지 않습니다.

```
~]$ getsebool httpd_can_network_connect_db
httpd_can_network_connect_db --> off
```

2. **Apache HTTP Server** 스크립트 및 모듈을 일시적으로 활성화하여 데이터베이스 서버에 연결하려면 **root**로 다음 명령을 입력합니다.

```
~]# setsebool httpd_can_network_connect_db on
```

3. **getsebool** 유틸리티를 사용하여 부울이 활성화되었는지 확인합니다.

```
~]$ getsebool httpd_can_network_connect_db
httpd_can_network_connect_db --> on
```

이렇게 하면 **Apache HTTP Server** 스크립트 및 모듈이 데이터베이스 서버에 연결할 수 있습니다.

4. 이 변경 사항은 재부팅 시 유지되지 않습니다. 재부팅 시 변경 사항이 지속되려면 루트로 명령에서 **setsebool -P *boolean-name*** 을 실행합니다.[3]

```
~]# setsebool -P httpd_can_network_connect_db on
```

4.6.3. 쉘 자동 완성

getsebool,setsebool 및 **semanage** 유틸리티로 쉘 자동 완성을 사용할 수 있습니다. **getsebool** 및 **setsebool** 로 자동 완성을 사용하여 명령줄 매개 변수와 부울을 모두 완료합니다. 명령줄 매개 변수만 나열하려면 명령 이름 뒤에 하이픈 문자(**"-"**)를 추가하고 **Tab** 키를 누릅니다.

```
~]# setsebool -[Tab]
-P
```

부울을 완료하려면 부울 이름 쓰기를 시작한 다음 **Tab:**을 누릅니다.

```
~]$ getsebool samba_[Tab]
samba_create_home_dirs   samba_export_all_ro      samba_run_unconfined
samba_domain_controller  samba_export_all_rw      samba_share_fusefs
samba_enable_home_dirs   samba_portmapper         samba_share_nfs
```

```
~]# setsebool -P virt_use_[Tab]
virt_use_comm     virt_use_nfs      virt_use_sanlock
virt_use_execmem  virt_use_rawip    virt_use_usb
virt_use_fusefs   virt_use_samba    virt_use_xserver
```

semanage 유틸리티는 하나씩 완성되는 여러 명령줄 인수와 함께 사용됩니다. **semanage** 명령의 첫 번째 인수는 **SELinux** 정책의 어떤 부분을 관리할지 지정하는 옵션입니다.

```
~]# semanage [Tab]
boolean    export     import     login      node       port
dontaudit  fcontext   interface  module     permissive  user
```

그런 다음 하나 이상의 명령줄 매개변수는 다음과 같습니다.

```
~]# semanage fcontext -[Tab]
-a          -D          --equal      --help       -m           -o
--add       --delete    -f           -l           --modify     -S
-C          --deleteall  --ftype     --list       -n           -t
-d          -e          -h           --locallist  --noheading  --type
```

마지막으로 부울, **SELinux** 사용자, 도메인 또는 기타와 같은 특정 **SELinux** 항목의 이름을 완료합니다. 항목을 입력하고 탭을 누릅니다:

```
~]# semanage fcontext -a -t samba<tab>
samba_etc_t                  samba_secrets_t
sambagui_exec_t              samba_share_t
samba_initrc_exec_t          samba_unconfined_script_exec_t
samba_log_t                  samba_unit_file_t
samba_net_exec_t
```

명령줄 매개 변수는 명령으로 연결할 수 있습니다.

```
~]# semanage port -a -t http_port_t -p tcp 81
```

4.7. SELINUX 컨텍스트 - 파일 레이블 지정

SELinux를 실행하는 시스템에서는 보안 관련 정보를 나타내는 방식으로 모든 프로세스와 파일에 레이블이 지정됩니다. 이 정보를 **SELinux** 컨텍스트라고 합니다. 파일의 경우 **ls -Z** 명령을 사용하여 확인할 수 있습니다.

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
```

이 예에서 **SELinux**는 사용자(**unconfined_u**), 역할(**object_r**), 유형(**user_home_t**) 및 수준(**s0**)을 제공합니다. 이 정보는 액세스 제어 결정을 내리는 데 사용됩니다. **DAC** 시스템에서는 **Linux** 사용자 및 그룹 **ID**를 기반으로 액세스가 제어됩니다. **SELinux** 정책 규칙은 **DAC** 규칙 후에 확인됩니다. **DAC** 규칙이 먼저 액세스를 거부하면 **SELinux** 정책 규칙이 사용되지 않습니다.

참고

기본적으로 새로 생성된 파일과 디렉터리는 상위 디렉터리의 **SELinux** 유형을 상속합니다. 예를 들어 **etc_t** 유형으로 레이블이 지정된 **/etc** 디렉토리에 새 파일을 만들 때 새 파일은 동일한 유형을 상속합니다.

```
~]$ ls -dZ - /etc
drwxr-xr-x. root root system_u:object_r:etc_t:s0       /etc
```

```
~]# touch /etc/file1
```

```
~]# ls -lZ /etc/file1
-rw-r--r--. root root unconfined_u:object_r:etc_t:s0   /etc/file1
```

SELinux는 **chcon,semanage fcontext,restorecon, matchpathcon** 과 같은 파일 시스템 레이블을 관리하기 위해 여러 명령을 제공합니다.

4.7.1. 임시 변경 사항: chcon

chcon 명령은 파일의 **SELinux** 컨텍스트를 변경합니다. 그러나 **chcon** 명령으로 변경한 내용은 파일 시스템 레이블 재레이블 또는 **restorecon** 명령 실행에서 지속되지 않습니다. **SELinux** 정책은 사용자가 지정된 파일에 대한 **SELinux** 컨텍스트를 수정할 수 있는지 여부를 제어합니다. **chcon** 을 사용하는 경우 사용자는 변경할 **SELinux** 컨텍스트의 전체 또는 일부를 제공합니다. 잘못된 파일 유형은 액세스를 거부하는 **SELinux**의 일반적인 원인입니다.

참고 자료

- **chcon -t *type file-name*** 명령을 실행하여 파일 유형을 변경합니다. 여기서 ***type*** 은 **httpd_sys_content_t** 와 같은 **SELinux** 유형이며 ***file-name*** 은 파일 또는 디렉토리 이름입니다.

```
~]$ chcon -t httpd_sys_content_t file-name
```

- **chcon -R -t *type directory-name*** 명령을 실행하여 디렉터리 및 해당 내용의 유형을 변경합니다. 여기서 ***type*** 은 **httpd_sys_content_t** 와 같은 **SELinux** 유형이며 ***directory-name*** 은 디렉터리 이름입니다.

```
~]$ chcon -R -t httpd_sys_content_t directory-name
```

절차 4.6. 파일 또는 디렉토리 유형 변경

다음 절차에서는 유형 및 **SELinux** 컨텍스트의 다른 특성을 변경하는 방법을 보여줍니다. 이 섹션의 예제에서는 디렉터리에 대해 동일한 작업을 수행합니다(예: **file1** 이 디렉터리인 경우).

1. 홈 디렉터리로 변경합니다.

2. 새 파일을 생성하고 **SELinux** 컨텍스트를 봅니다.

```
~]$ touch file1
```

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
```

이 예에서 **file1** 의 **SELinux** 컨텍스트에는 **SELinux unconfined_u** 사용자, **object_r** 역할, **user_home_t** 유형 및 **s0** 수준이 포함됩니다. **SELinux** 컨텍스트의 각 부분에 대한 설명은 **2장. *SELinux 컨텍스트*** 의 내용을 참조하십시오.

3. 다음 명령을 입력하여 유형을 **samba_share_t** 로 변경합니다. **t** 옵션은 유형만 변경합니다. 그런 다음 변경 사항을 확인합니다.

```
~]$ chcon -t samba_share_t file1
```

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:samba_share_t:s0 file1
```

4. 다음 명령을 사용하여 **file1** 파일의 **SELinux** 컨텍스트를 복원합니다. **v** 옵션을 사용하여 변경 사항을 확인하십시오.

```
~]$ restorecon -v file1
restorecon reset file1 context unconfined_u:object_r:samba_share_t:s0-
>system_u:object_r:user_home_t:s0
```

이 예제에서는 이전 유형 **samba_share_t** 가 올바른 **user_home_t** 유형으로 복원됩니다. 타겟 정책(**Red Hat Enterprise Linux**의 기본 **SELinux** 정책)을 사용하는 경우 **restorecon** 명령은 **/etc/selinux/targeted/contexts/files/** 디렉터리에 있는 파일을 읽고 어떤 **SELinux** 컨텍스트 파일을 보유할지 확인합니다.

절차 4.7. 디렉토리 및 콘텐츠 유형 변경

다음 예제에서는 새 디렉터리를 만들고 디렉터리의 파일 유형을 **Apache HTTP** 서버에서 사용하는 유형으로 변경하는 방법을 보여줍니다. 이 예제의 구성은 **Apache HTTP** 서버가 **/var/www/html/**대신 다른 문서 루트를 사용하려는 경우에 사용됩니다.

1. **root** 사용자로 이 디렉터리에 새 **web/** 디렉터리를 만든 다음 **3**개의 빈 파일(**file1,file2** 및 **file3**)을 만듭니다. 이 디렉터리의 **web/** 디렉토리와 파일에는 **default_t** 유형으로 레이블이 지정됩니다.

```
~]# mkdir /web
```

```
~]# touch /web/file{1,2,3}
```

```
~]# ls -dZ /web
drwxr-xr-x  root root unconfined_u:object_r:default_t:s0 /web
```

```
~]# ls -lZ /web
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file1
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file2
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file3
```

2. **root**로 다음 명령을 입력하여 **web/** 디렉토리(및 해당 콘텐츠)의 유형을 **httpd_sys_content_t** 로 변경합니다.

```
~]# chcon -R -t httpd_sys_content_t /web/
```

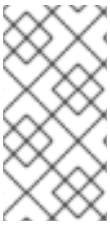
```
~]# ls -dZ /web/
drwxr-xr-x  root root unconfined_u:object_r:httpd_sys_content_t:s0 /web/
```

```
~]# ls -lZ /web/
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file1
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file2
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file3
```

3. 기본 **SELinux** 컨텍스트를 복원하려면 **root**로 **restorecon** 유틸리티를 사용합니다.

```
~]# restorecon -R -v /web/
restorecon reset /web context unconfined_u:object_r:httpd_sys_content_t:s0-
>system_u:object_r:default_t:s0
restorecon reset /web/file2 context unconfined_u:object_r:httpd_sys_content_t:s0-
>system_u:object_r:default_t:s0
restorecon reset /web/file3 context unconfined_u:object_r:httpd_sys_content_t:s0-
>system_u:object_r:default_t:s0
restorecon reset /web/file1 context unconfined_u:object_r:httpd_sys_content_t:s0-
>system_u:object_r:default_t:s0
```

chcon 에 대한 자세한 내용은 **chcon(1)** 도움말 페이지를 참조하십시오.



참고

유형 적용은 **SELinux** 대상 정책에 사용되는 기본 권한 제어입니다. 대부분의 경우 **SELinux** 사용자 및 역할은 무시할 수 있습니다.

4.7.2. 영구적인 변경 사항: semanage fcontext

semanage fcontext 명령은 파일의 **SELinux** 컨텍스트를 변경하는 데 사용됩니다. 새로 생성된 파일 및 디렉터리에 대한 컨텍스트를 표시하려면 **root**로 다음 명령을 입력합니다.

```
~]# semanage fcontext -C -l
```

semanage fcontext로 변경한 내용은 다음 유틸리티에서 사용합니다. **setfiles** 유틸리티는 파일 시스템에 레이블이 다시 지정되고 **restorecon** 유틸리티가 기본 **SELinux** 컨텍스트를 복원할 때 사용됩니다. 즉, 파일 시스템에 레이블을 다시 지정하는 경우에도 **semanage fcontext** 로 변경한 내용이 지속됩니다. **SELinux** 정책은 사용자가 지정된 파일에 대한 **SELinux** 컨텍스트를 수정할 수 있는지 여부를 제어합니다.

참고 자료

파일 시스템 레이블 변경 후에도 유지되는 **SELinux** 컨텍스트 변경을 수행하려면 다음을 수행합니다.

1. 파일 또는 디렉토리의 전체 경로를 사용하도록 다음 명령을 입력합니다.

```
~]# semanage fcontext -a options file-name|directory-name
```

2. **restorecon** 유틸리티를 사용하여 컨텍스트 변경 사항을 적용합니다.

```
~]# restorecon -v file-name|directory-name
```

semanage fcontext와 함께 정규 표현식 사용

semanage fcontext 명령이 올바르게 작동하려면 정규화된 경로 또는 **Perl** 호환 정규 표현식(**PCRE**)을 사용할 수 있습니다. 사용 중인 유일한 ***PCRE*** 플래그는 **PCRE2_DOTALL** 이므로 **.** 와일드카드가 새 행을 포함하여 모든 항목과 일치합니다. 경로를 나타내는 문자열은 바이트로 처리됩니다. 즉 **ASCII**가 아닌 문자는 단일 와일드카드와 일치하지 않습니다.

semanage fcontext 를 사용하여 지정된 파일 컨텍스트 정의는 해당 정의 방법의 역순으로 평가됩니다. 최신 항목은 줄기 길이에 관계없이 먼저 평가됩니다. **file_contexts.local** 에 저장된 로컬 파일 컨텍스트 수정은 정책 모듈에 지정된 것보다 우선 순위가 높습니다. 즉, **file _contexts.local**에서 지정된 파일 경로와 일치하는 항목이 있을 때마다 다른 파일 컨텍스트 정의는 고려되지 않습니다.

중요

semanage fcontext 명령을 사용하여 지정된 파일 컨텍스트 정의는 다른 모든 파일 컨텍스트 정의를 효과적으로 재정의합니다. 따라서 모든 정규 표현식은 의도치 않게 파일 시스템의 다른 부분에 영향을 주지 않도록 가능한 한 구체적이어야 합니다.

적용된 **file-context** 정의 및 플래그에 사용되는 정규식 유형에 대한 자세한 내용은 **semanage-fcontext(8)** 도움말 페이지를 참조하십시오.

절차 **4.8.** 파일 또는 디렉토리 유형 변경

다음 예제에서는 파일 유형 및 **SELinux** 컨텍스트의 다른 특성을 변경하는 방법을 보여줍니다. 이 예제는 디렉토리에 대해 동일한 작동(예: **file1** 이 디렉터리인 경우).

1. **root** 사용자로 **/etc** 디렉토리에 새 파일을 만듭니다. 기본적으로 **/etc** 에서 새로 생성된 파일은 **etc_t** 유형으로 레이블이 지정됩니다.

```
~]# touch /etc/file1
```

```
~]$ ls -Z /etc/file1
-rw-r--r--  root root unconfined_u:object_r:etc_t:s0       /etc/file1
```

디렉터리에 대한 정보를 나열하려면 다음 명령을 사용합니다.

```
~]$ ls -dZ directory_name
```

2. **root**로 다음 명령을 입력하여 **file1** 유형을 **samba_share_t** 로 변경합니다. **a** 옵션은 새 레코드를 추가하고 **-t** 옵션은 유형을 정의합니다(**samba_share_t**). 이 명령을 실행하면 유형을 직접 변경하지 않습니다. **file1** 은 여전히 **etc_t** 유형으로 레이블이 지정됩니다.

```
~]# semanage fcontext -a -t samba_share_t /etc/file1
```

```
~]# ls -Z /etc/file1
-rw-r--r--  root root unconfined_u:object_r:etc_t:s0       /etc/file1
```

```
~]$ semanage fcontext -C -l
/etc/file1    unconfined_u:object_r:samba_share_t:s0
```

3. **root**로 **restorecon** 유틸리티를 사용하여 유형을 변경합니다. **semanage** 가 **/etc/file1** 의 **file_contexts.local** 에 항목을 추가했기 때문에**restorecon** 은 유형을 **samba_share_t** 로 변경합니다.

```
~]# restorecon -v /etc/file1
restorecon reset /etc/file1 context unconfined_u:object_r:etc_t:s0-
>system_u:object_r:samba_share_t:s0
```

절차 4.9. 디렉토리 및 콘텐츠 유형 변경

다음 예제에서는 새 디렉터리를 만들고 디렉터리의 파일 유형을 해당 내용과 함께 **Apache HTTP** 서버에서 사용하는 유형으로 변경하는 방법을 보여줍니다. 이 예제의 설정은 **Apache HTTP** 서버가 **/var/www/html/** 대신 다른 문서 루트를 사용하려는 경우 사용됩니다.

1.

root 사용자로 이 디렉터리에 새 **web/** 디렉터리를 만든 다음 **3**개의 빈 파일**(file1,file2** 및 **file3)**을 만듭니다**.** 이 디렉터리의 **web/** 디렉토리와 파일에는 **default_t** 유형으로 레이블이 지정됩니다**.**

```
~]# mkdir /web
```

```
~]# touch /web/file{1,2,3}
```

```
~]# ls -dZ /web
drwxr-xr-x  root root unconfined_u:object_r:default_t:s0 /web
```

```
~]# ls -lZ /web
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file1
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file2
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file3
```

2.

root로 다음 명령을 입력하여 웹**/** 디렉터리의 유형과 해당 파일 유형을 **httpd_sys_content_t** 로 변경합니다**. a** 옵션은 새 레코드를 추가하고 **-t** 옵션은 유형 **(httpd_sys_content_t)**을 정의합니다**. "/web(/.*)?"** 정규 표현식을 사용하면 **semanage** 가 웹**/** 및 파일에 변경 사항을 적용합니다**.** 이 명령을 실행하면 유형을 직접 변경하지 않습니다**.** 웹**/** 및 파일은 여전히 **default_t** 유형으로 레이블이 지정됩니다**.**

```
~]# semanage fcontext -a -t httpd_sys_content_t "/web(/.*)?"
```

```
~]$ ls -dZ /web
drwxr-xr-x  root root unconfined_u:object_r:default_t:s0 /web
```

```
~]$ ls -lZ /web
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file1
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file2
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file3
```

semanage fcontext -a -t httpd_sys_content_t "/web(/.*)?" 명령은 **/etc/selinux/targeted/contexts/files/file_contexts.local**에 다음 항목을 추가합니다**.**

```
/web(/.*)?    system_u:object_r:httpd_sys_content_t:s0
```

3.

root로서 **restorecon** 유틸리티를 사용하여 **web/** 의 유형과 여기에 있는 모든 파일을 변경합니다**. R**은 **recursive(**재귀**)**용 입니다**.** 즉**, web/** 아래의 모든 파일과 디렉터리에 **httpd_sys_content_t** 유형으로 레이블이 지정됩니다**. semanage** 가 **/web(/ .*)**에 대해 **file.contexts.local** 에 항목을 추가했기 때문에 **restorecon** 은 유형을 **httpd_sys_content_t** 로 변경합니다**.**

```
~]# restorecon -R -v /web
```

```
restorecon reset /web context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
restorecon reset /web/file2 context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
restorecon reset /web/file3 context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
restorecon reset /web/file1 context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
```

기본적으로 새로 생성된 파일과 디렉터리는 상위 디렉터리의 **SELinux** 유형을 상속받습니다.

절차 4.10. 추가된 컨텍스트 삭제

다음 예제에서는 **SELinux** 컨텍스트 추가 및 제거를 보여줍니다. 컨텍스트가 정규 표현식의 일부인 경우**(**예**: /web(/.*)?)** 정규 표현식 관련 따옴표를 사용합니다.

```
~]# semanage fcontext -d "/web(/.*)?"
```

1. 컨텍스트를 제거하려면 **root**로 다음 명령을 입력합니다. 여기서 ***file-name*** **|*****directory-name*** 은 **file_contexts.local** 의 첫 번째 부분입니다 **:**

```
~]# semanage fcontext -d file-name|directory-name
```

다음은 **file_contexts.local** 의 컨텍스트의 예입니다 **:**

```
/test    system_u:object_r:httpd_sys_content_t:s0
```

첫 번째 부분이 테스트를 통해**. restorecon** 을 실행한 후 **test/** 디렉터리에 **httpd_sys_content_t** 로 레이블이 지정되지 않도록 또는 파일 시스템의 레이블을 다시 지정한 후 다음 명령을 루트로 입력하여 **file_contexts.local**에서 컨텍스트를 삭제합니다.

```
~]# semanage fcontext -d /test
```

2. **root**로 **restorecon** 유틸리티를 사용하여 기본 **SELinux** 컨텍스트를 복원합니다.

semanage 에 대한 자세한 내용은 **semanage(8)** 및 **semanage-fcontext(8)** 매뉴얼 페이지를 참조하십시오.

중요

semanage fcontext -a 로 **SELinux** 컨텍스트를 변경할 때 파일 또는 디렉토리의 전체 경로를 사용하여 파일 시스템 레이블 변경 후 또는 **restorecon** 명령이 실행된 후 파일의 레이블이 잘못 지정되지 않도록 합니다.

4.7.3. 파일 문맥을 확인하는 방법

파일 컨텍스트를 결정하는 것은 시스템 보안 정책(**.fc** 파일)에 지정된 파일 컨텍스트 정의를 기반으로 합니다. 시스템 정책에 따라 **semanage** 는 **file_contexts.homedirs** 및 **file_contexts** 파일을 생성합니다.

시스템 관리자는 **semanage fcontext** 명령을 사용하여 파일 컨텍스트 정의를 사용자 지정할 수 있습니다. 이러한 사용자 지정은 **file_contexts.local** 파일에 저장됩니다.

matchpathcon 또는 **restorecon** 과 같은 레이블 유틸리티가 지정된 경로에 적절한 레이블을 결정하는 경우 먼저 로컬 변경 사항(**file_contexts.local**)을 검색합니다. 유틸리티가 일치하는 패턴을 찾지 못하면 **file_contexts.homedirs** 파일과 마지막으로 **file_contexts** 파일을 검색합니다. 그러나 지정된 파일 경로와 일치하는 항목이 있을 때마다 검색이 종료되고 유틸리티는 추가 파일 컨텍스트 정의를 찾습니다. 즉, 홈 디렉터리 관련 파일 컨텍스트가 나머지보다 우선 순위가 높으며 로컬 사용자 지정은 시스템 정책을 재정의합니다.

시스템 정책 (**file_contexts .homedirs** 및 **file_contexts** 파일의 내용)으로 지정된 파일 컨텍스트 정의는 평가 전에 줄기 길이(와일드카드 앞에 경로 접두사)를 기준으로 정렬됩니다. 이는 가장 구체적인 경로가 선택됨을 의미합니다. 그러나 **semanage fcontext** 를 사용하여 지정된 파일 컨텍스트 정의는 해당 정의 방법의 역순으로 평가됩니다. 최신 항목은 줄기 길이에 관계없이 먼저 평가됩니다.

자세한 내용은 다음을 참조하십시오.

- **chcon** 을 사용하여 파일의 컨텍스트 변경은 4.7.1절. “임시 변경 사항: chcon” 을 참조하십시오.
- **semanage fcontext** 를 사용하여 파일 컨텍스트 정의 변경 및 추가는 4.7.2절. “영구적인 변경 사항: semanage fcontext” 을 참조하십시오.
- 시스템 정책 작업을 통해 파일 컨텍스트 정의 변경 및 추가는 4.10절. “SELinux 레이블 유지 관리” 또는 4.12절. “SELinux 정책 모듈 우선순위 및 비활성화” 을 참조하십시오.

4.8. FILE_T 및 DEFAULT_T 유형

확장 속성(EA)을 지원하는 파일 시스템을 사용하는 경우 **file_t** 유형은 아직 **EA** 값이 할당되지 않은 파일의 기본 유형입니다. 이 유형은 이 목적을 위해서만 사용되며 올바른 레이블 파일 시스템에 존재하지 않습니다. **SELinux**를 실행하는 시스템의 모든 파일에 적절한 **SELinux** 컨텍스트가 있어야 하고 **file_t** 유형은 파일 컨텍스트에서 사용되지 않기 때문입니다.[4].

default_t 유형은 **file-context** 구성의 패턴과 일치하지 않는 파일에 사용됩니다. 이러한 파일은 디스크에 컨텍스트가 없는 파일과 구분될 수 있으며 일반적으로 제한된 도메인에 액세스할 수 없습니다. 예를 들어 **mydirectory/** 와 같은 새 최상위 디렉터리를 만드는 경우 이 디렉터리는 **default_t** 유형으로 레이블이 지정될 수 있습니다. 서비스에서 이 디렉토리에 액세스해야 하는 경우 이 위치에 대한 **file-contexts** 구성을 업데이트해야 합니다. 파일 컨텍스트 구성에 컨텍스트를 추가하는 방법에 대한 자세한 내용은 4.7.2절. “영구적인 변경 사항: semanage fcontext” 을 참조하십시오.

4.9. 파일 시스템 마운트

기본적으로 확장 속성을 지원하는 파일 시스템이 마운트되면 각 파일의 보안 컨텍스트는 파일의 ***security.selinux*** 확장 특성에서 가져옵니다. 확장 특성을 지원하지 않는 파일 시스템의 파일에는 파일 시스템 유형에 따라 정책 구성에서 하나의 기본 보안 컨텍스트가 할당됩니다.

mount -o context 명령을 사용하여 기존 확장 속성을 덮어쓰거나 확장 속성을 지원하지 않는 파일 시스템에 대해 다른 기본 컨텍스트를 지정합니다. 이 기능은 여러 시스템에서 사용되는 이동식 미디어와 같이 올바른 속성을 제공하기 위해 파일 시스템을 신뢰하지 않는 경우에 유용합니다. **mount -o context** 명령을 사용하여 **FAT**(파일 할당 테이블) 또는 **NFS** 볼륨과 같은 확장된 속성을 지원하지 않는 파일 시스템의 레이블을 지원할 수도 있습니다. 컨텍스트 옵션으로 지정된 컨텍스트 는 디스크에 작성되지 않습니다. 원래 컨텍스트가 유지되며, 파일 시스템에 첫 번째 위치에서 확장된 속성이 있는 경우 컨텍스트 없이 마운트할 때 표시됩니다.

파일 시스템 레이블 지정에 대한 자세한 내용은 **James Morris**의 **"SELinux**의 파일 시스템 레이블 지정**"** 문서를 참조하십시오 **http://www.linuxjournal.com/article/7426**.

4.9.1. 컨텍스트 마운트

지정된 컨텍스트를 사용하여 파일 시스템을 마운트하려면 기존 컨텍스트가 있는 경우 재정의하거나 확장 속성을 지원하지 않는 파일 시스템에 다른 기본 컨텍스트를 지정하려면 루트 사용자로 **mount -o context=*SELinux_user:role:type:level*** 명령을 사용합니다. 컨텍스트 변경 사항은 디스크에 기록되지 않습니다. 기본적으로 클라이언트쪽의 **NFS** 마운트는 **NFS** 볼륨의 정책에서 정의한 기본 컨텍스트로 레이블이 지정됩니다. 일반적인 정책에서 이 기본 컨텍스트는 **nfs_t** 유형을 사용합니다. 추가 마운트 옵션이 없으면 **Apache HTTP** 서버와 같은 다른 서비스를 사용하여 **NFS** 볼륨을 공유하지 못하게 할 수 있습니다. 다음 예제에서는 **Apache HTTP** 서버를 사용하여 공유할 수 있도록 **NFS** 볼륨을 마운트합니다.

```
~]# mount server:/export /local/mount/point -o \ context="system_u:object_r:httpd_sys_content_t:s0"
```

이 파일 시스템에서 새로 생성된 파일과 디렉터리에 **-o** 컨텍스트로 지정된 **SELinux** 컨텍스트가 있는 것처럼 나타납니다. 그러나 이러한 변경 사항은 디스크에 작성되지 않으므로 이 옵션과 함께 지정된 컨텍스트는 마운트 간에 유지되지 않습니다. 따라서 필요한 컨텍스트를 유지하려면 마운트할 때마다 이 옵션을 동일한 컨텍스트와 함께 사용해야 합니다. 컨텍스트 마운트를 영구적으로 만드는 방법에 대한 자세한 내용은 4.9.5절. “컨텍스트 마운트 영구 설정” 을 참조하십시오.

유형 적용은 **SELinux** 대상 정책에 사용되는 기본 권한 제어입니다. 대부분의 경우 **SELinux** 사용자 및 역할은 무시할 수 있으므로 **SELinux** 컨텍스트를 **-o** 컨텍스트 로 재정의할 때 **SELinux system_u** 사용자 및 **object_r** 역할을 사용하고 유형에 집중할 수 있습니다. **MLS** 정책 또는 다중 범주 보안을 사용하지 않는 경우 **s0** 수준을 사용합니다.

참고

파일 시스템을 컨텍스트 옵션으로 마운트하면 사용자와 프로세스의 컨텍스트 변경이 금지됩니다. 예를 들어 컨텍스트 옵션을 사용하여 마운트된 파일 시스템에서 **chcon** 명령을 실행하면 **Operation**이 지원되지 않습니다.

4.9.2. 기본 문맥 변경

4.8절. “file_t 및 default_t 유형” 에서 언급했듯이 확장 속성을 지원하는 파일 시스템에서는 디스크의 **SELinux** 컨텍스트가 없는 파일에 액세스하면 **SELinux** 정책에 정의된 기본 컨텍스트가 있는 것처럼 처리됩니다. 일반적인 정책에서 이 기본 컨텍스트는 **file_t** 유형을 사용합니다. 다른 기본 컨텍스트를 사용하려면 **defcontext** 옵션을 사용하여 파일 시스템을 마운트합니다.

다음 예제에서는 새로 생성된 파일 시스템을 **/dev/sda2** 에 새로 생성된 **test/** 디렉터리에 마운트합니다. **/etc/selinux/targeted/contexts/files/**에 **test/** 디렉터리에 대한 컨텍스트를 정의하는 규칙이 없다고 가정합니다.

```
~]# mount /dev/sda2 /test/ -o defcontext="system_u:object_r:samba_share_t:s0"
```

이 예에서는 다음을 수행합니다.

- **defcontext** 옵션은 **system_u:object_r:samba_share_t:s0** 이 "레이블되지 않은 파일의 기본 보안 컨텍스트"를 정의합니다.[5].

- 마운트되면 파일 시스템의 루트 디렉토리(테스트/)는 **defcontext** 에서 지정한 컨텍스트로 레이블이 지정된 것처럼 처리됩니다(이 레이블은 디스크에 저장되지 않음). 이는 **test/**: 새 파일에 생성된 파일의 레이블 지정에 영향을 미치며 새 파일은 **samba_share_t** 유형을 상속하고 이러한 레이블은 디스크에 저장됩니다.

- **test/** 에서 만든 파일은 **defcontext** 옵션으로 마운트된 반면, 해당 레이블을 유지합니다.

4.9.3. NFS 볼륨 마운트

기본적으로 클라이언트쪽의 **NFS** 마운트는 **NFS** 볼륨의 정책에서 정의한 기본 컨텍스트로 레이블이 지정됩니다. 일반적인 정책에서 이 기본 컨텍스트는 **nfs_t** 유형을 사용합니다. 정책 구성에 따라 **Apache HTTP** 서버 및 **MariaDB**와 같은 서비스는 **nfs_t** 유형으로 레이블이 지정된 파일을 읽을 수 없습니다. 이렇게 하면 이 유형으로 레이블이 지정된 파일 시스템이 마운트되지 않은 다음 다른 서비스에서 읽거나 내보내지 못하게 할 수 있습니다.

NFS 볼륨을 마운트하고 다른 서비스로 해당 파일 시스템을 읽거나 내보내려면 마운트 시 컨텍스트 옵션을 사용하여 **nfs_t** 유형을 재정의합니다. 다음 컨텍스트 옵션을 사용하여 **Apache HTTP** 서버를 사용하여 공유할 수 있도록 **NFS** 볼륨을 마운트합니다.

```
~]# mount server:/export /local/mount/point -o context="system_u:object_r:httpd_sys_content_t:s0"
```

이러한 변경 사항은 디스크에 작성되지 않으므로 이 옵션과 함께 지정된 컨텍스트는 마운트 간에 유지되지 않습니다. 따라서 필요한 컨텍스트를 유지하려면 마운트할 때마다 이 옵션을 동일한 컨텍스트와 함께 사용해야 합니다. 컨텍스트 마운트를 영구적으로 만드는 방법에 대한 자세한 내용은 4.9.5절. “컨텍스트 마운트 영구 설정” 을 참조하십시오.

컨텍스트 옵션을 사용하여 파일 시스템을 마운트하는 대신, 부울을 활성화하여 **nfs_t** 유형으로 레이블이 지정된 파일 시스템에 액세스할 수 있습니다. **nfs_t** 유형에 대한 서비스 액세스를 허용하도록 부울을 구성하는 방법에 대한 지침은 II 부. 제한된 서비스 관리 을 참조하십시오.

4.9.4. 여러 NFS 마운트

동일한 **NFS** 내보내기에서 여러 마운트를 마운트할 때 각 마운트의 **SELinux** 컨텍스트를 다른 컨텍스트로 재정의하려고 하면 후속 마운트 명령이 실패합니다. 다음 예에서 **NFS** 서버에는 두 개의 하위 디렉터리인 **web/** 및 **database/**가 있는 단일 내보내기 **export/** 가 있습니다. 다음 명령은 단일 **NFS** 내보내기에서 두 개의 마운트를 시도하고 각각에 대한 컨텍스트를 재정의합니다.

```
~]# mount server:/export/web /local/web -o context="system_u:object_r:httpd_sys_content_t:s0"
```

```
~]# mount server:/export/database /local/database -o context="system_u:object_r:mysqld_db_t:s0"
```

두 번째 **mount** 명령이 실패하고 다음은 **/var/log/messages** 에 기록됩니다.

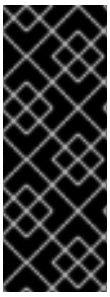
```
kernel: SELinux: mount invalid.  Same superblock, different security settings for (dev 0:15, type nfs)
```

마운트마다 다른 컨텍스트가 있는 단일 **NFS** 내보내기에서 여러 마운트를 마운트하려면 **-o nosharecache,context** 옵션을 사용합니다. 다음 예제에서는 마운트마다 다른 컨텍스트를 사용하여 단일 **NFS** 내보내기에서 여러 마운트를 마운트합니다(각 각각에 단일 서비스 액세스 허용).

```
~]# mount server:/export/web /local/web -o
nosharecache,context="system_u:object_r:httpd_sys_content_t:s0"
```

```
~]# mount server:/export/database /local/database -o \
nosharecache,context="system_u:object_r:mysqld_db_t:s0"
```

이 예에서 **server:/export/web** 은 **/local/web/** 디렉터리에 로컬로 마운트되며 모든 파일에 **httpd_sys_content_t** 유형으로 레이블이 지정되어 **Apache HTTP Server** 액세스를 허용합니다. **server:/export/database** 는 **/local/database/** 에 로컬로 마운트되며 모든 파일이 **mysqld_db_t** 유형으로 레이블이 지정되어 **MariaDB** 액세스를 허용합니다. 이러한 유형 변경 사항은 디스크에 기록되지 않습니다.



중요

nosharecache 옵션을 사용하면 **/export/web/** 을 여러 번 마운트하는 것과 같은 컨텍스트를 사용하여 동일한 내보내기의 하위 디렉터리를 여러 번 마운트할 수 있습니다. 다른 컨텍스트에서 파일에 액세스할 수 있는 중복 마운트를 생성하므로 내보내기에서 동일한 하위 디렉터리를 여러 번 마운트하지 마십시오.

4.9.5. 컨텍스트 마운트 영구 설정

다시 마운트하고 재부팅할 때마다 컨텍스트를 영구적으로 마운트하려면 **/etc/fstab** 파일에서 파일 시스템에 대한 항목을 추가하거나 자동 마운터 맵을 추가하고 필요한 컨텍스트를 마운트 옵션으로 사용합니다. 다음 예제에서는 **NFS** 컨텍스트 마운트에 사용할 항목을 **/etc/fstab** 에 추가합니다.

```
server:/export /local/mount/ nfs context="system_u:object_r:httpd_sys_content_t:s0" 0 0
```

4.10. SELINUX 레이블 유지 관리

이 섹션에서는 파일 및 디렉터리를 복사, 이동 및 보관할 때 **SELinux** 컨텍스트에 발생하는 사항에 대해 설명합니다. 또한 복사 및 보관 시 컨텍스트를 보존하는 방법을 설명합니다.

4.10.1. 파일 및 디렉토리 복사

파일 또는 디렉터리가 복사되면 새 파일 또는 디렉터리가 없는 경우 새로 생성됩니다. 새 파일 또는 디렉터리의 컨텍스트는 원래 컨텍스트를 유지하는 데 옵션을 사용하지 않는 한 원본 파일이나 디렉터리의

컨텍스트가 아닌 기본 레이블 규칙을 기반으로 합니다. 예를 들어 사용자 홈 디렉터리에 생성된 파일은 user_home_t 유형으로 레이블이 지정됩니다.

```
~]$ touch file1
```

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
```

이러한 파일이 /etc와 같은 다른 디렉토리에 복사되면 /etc 의 default-labeling 규칙에 따라 새 파일이 생성됩니다 . 추가 옵션 없이 파일을 복사하면 원래 컨텍스트가 유지되지 않을 수 있습니다.

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
```

```
~]# cp file1 /etc/
```

```
~]$ ls -Z /etc/file1
-rw-r--r--  root root unconfined_u:object_r:etc_t:s0   /etc/file1
```

file1 을 /etc/ file1에 복사하면 /etc/file1 이 없으면 /etc/file1 이 새 파일로 생성됩니다. 위 예제와 같이 /etc/file1 은 기본 레이블 규칙에 따라 etc_t 유형으로 레이블이 지정됩니다.

파일이 기존 파일을 통해 복사되면 사용자가 --preserve=context 와 같은 원본 파일의 컨텍스트를 보존하기 위해 지정한 cp 옵션이 아닌 한 기존 파일의 컨텍스트가 보존됩니다. SELinux 정책은 복사 중에 컨텍스트가 유지되지 않도록 할 수 있습니다.

절차 4.11. SELinux 컨텍스트를 보존하지 않고 복사

이 절차에서는 파일을 cp 명령으로 복사할 때 옵션을 지정하지 않으면 타겟 상위 디렉터리에서 유형을 상속함을 보여줍니다.

1. **사용자의 홈 디렉터리에서 파일을 만듭니다. 파일은 user_home_t 유형으로 레이블이 지정됩니다.**

```
~]$ touch file1
```

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
```

2. **/var/www/html/ 디렉터리에 다음 명령과 함께 표시된 대로 httpd_sys_content_t 유형으로 레이블이 지정됩니다.**

```
~]$ ls -dZ /var/www/html/
drwxr-xr-x  root root system_u:object_r:httpd_sys_content_t:s0 /var/www/html/
```

3. **file1** 이 **/var/www/html/** 에 복사되면 **httpd_sys_content_t** 유형을 상속합니다.

```
~]# cp file1 /var/www/html/
```

```
~]$ ls -Z /var/www/html/file1
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 /var/www/html/file1
```

절차 4.12. 복사 시 SELinux 컨텍스트 보존

다음 절차에서는 복사 시 컨텍스트를 보존하기 위해 **--preserve=context** 옵션을 사용하는 방법을 보여줍니다.

1. 사용자의 홈 디렉터리에서 파일을 만듭니다. 파일은 **user_home_t** 유형으로 레이블이 지정됩니다.

```
~]$ touch file1
```

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
```

2. **/var/www/html/** 디렉터리에 다음 명령과 함께 표시된 대로 **httpd_sys_content_t** 유형으로 레이블이 지정됩니다.

```
~]$ ls -dZ /var/www/html/
drwxr-xr-x  root root system_u:object_r:httpd_sys_content_t:s0 /var/www/html/
```

3. **preserve =context** 옵션을 사용하면 복사 작업 중에 **SELinux** 컨텍스트가 유지됩니다. 아래 표시된 대로 **file1** 의 **user_home_t** 유형은 **/var/www/html/** 에 파일을 복사할 때 보존됩니다.

```
~]# cp --preserve=context file1 /var/www/html/
```

```
~]$ ls -Z /var/www/html/file1
-rw-r--r--  root root unconfined_u:object_r:user_home_t:s0 /var/www/html/file1
```

절차 4.13. 문맥 복사 및 변경

다음 절차에서는 **--context** 옵션을 사용하여 대상 복사의 컨텍스트를 변경하는 방법을 보여줍니다. 다음 예는 사용자의 홈 디렉터리에서 수행됩니다.

1. 사용자의 홈 디렉터리에서 파일을 만듭니다. 파일은 **user_home_t** 유형으로 레이블이 지정됩니다.

```
~]$ touch file1
```

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
```

2. **context** 옵션을 사용하여 **SELinux** 컨텍스트를 정의합니다.

```
~]$ cp --context=system_u:object_r:samba_share_t:s0 file1 file2
```

3. 컨텍스트가 없으면**file2** 는 **unconfined_u:object_r:user_home_t** 컨텍스트로 레이블이 지정됩니다.

```
~]$ ls -Z file1 file2
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
-rw-rw-r--  user1 group1 system_u:object_r:samba_share_t:s0 file2
```

절차 4.14. 기존 파일에서 파일 복사

이 절차에서는 컨텍스트를 보존하는 데 옵션을 사용하지 않는 한 기존 파일을 통해 파일을 복사할 때 기존 파일의 컨텍스트가 보존되는 것을 보여줍니다.

1. **root**로 **/etc** 디렉토리에 새 파일 **file1** 을 만듭니다. 아래 표시된 대로 파일은 **etc_t** 유형으로 레이블이 지정됩니다.

```
~]# touch /etc/file1
```

```
~]$ ls -Z /etc/file1
-rw-r--r--  root root unconfined_u:object_r:etc_t:s0   /etc/file1
```

2. 다른 파일 **file2** 를 **/tmp** 디렉토리에 만듭니다. 아래 표시된 대로 파일은 **user_tmp_t** 유형으로 레이블이 지정됩니다.

```
~]$ touch /tmp/file2
```

```
~$ ls -Z /tmp/file2
-rw-r--r--  root root unconfined_u:object_r:user_tmp_t:s0 /tmp/file2
```

3. **file1** 을 **file2** 로 덮어쓰기 **:**

```
~]# cp /tmp/file2 /etc/file1
```

4. 복사 후 다음 명령은 / **etc/file 1**을 대체한 **/tmp/file2** 의 **user_tmp_t** 유형이 아닌 **etc_t** 유형으로 레이블이 지정된 **file 1** 을 보여줍니다**.**

```
~]$ ls -Z /etc/file1
-rw-r--r--  root root unconfined_u:object_r:etc_t:s0   /etc/file1
```

중요

파일 및 디렉터리를 이동하지 않고 복사합니다**.** 이렇게 하면 올바른 **SELinux** 컨텍스트로 레이블이 지정되도록 하는 데 도움이 됩니다**.** 잘못된 **SELinux** 컨텍스트는 프로세스가 해당 파일 및 디렉터리에 액세스하지 못하도록 할 수 있습니다**.**

4.10.2. 파일 및 디렉토리 이동

파일과 디렉토리는 이동 시 현재 **SELinux** 컨텍스트를 유지합니다**.** 대부분의 경우 이 동작은 이동 중인 위치에 대해 올바르지 않습니다**.** 다음 예제에서는 사용자의 홈 디렉터리에서 **Apache HTTP** 서버에서 사용하는 **/var/www/html/** 디렉터리로 파일을 이동하는 방법을 보여줍니다**.** 파일이 이동되었으므로 올바른 **SELinux** 컨텍스트를 상속하지 않습니다**.**

절차 4.15. 파일 및 디렉토리 이동

1. 홈 디렉터리로 변경하고 파일을 만듭니다**.** 파일은 **user_home_t** 유형으로 레이블이 지정됩니다**.**

```
~]$ touch file1
```

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
```

2. **/var/www/html/** 디렉터리의 **SELinux** 컨텍스트를 보려면 다음 명령을 입력합니다**.**

```
~]$ ls -dZ /var/www/html/
drwxr-xr-x  root root system_u:object_r:httpd_sys_content_t:s0 /var/www/html/
```

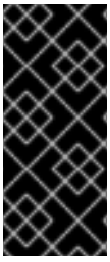
기본적으로 **/var/www/html/** 은 **httpd_sys_content_t** 유형으로 레이블이 지정됩니다. **/var/www/html/** 에 생성된 파일과 디렉토리는 이 유형을 상속받습니다. 따라서 이 유형으로 레이블이 지정됩니다.

3. **root**인 경우 **file1** 을 **/var/www/html/** 로 이동합니다. 이 파일이 이동되었으므로 현재 **user_home_t** 유형을 유지합니다.

```
~]# mv file1 /var/www/html/
```

```
~]# ls -Z /var/www/html/file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 /var/www/html/file1
```

기본적으로 **Apache HTTP** 서버는 **user_home_t** 유형으로 레이블이 지정된 파일을 읽을 수 없습니다. 웹 페이지로 구성된 모든 파일에 **user_home_t** 유형 또는 **Apache HTTP Server**에서 읽을 수 없는 다른 유형으로 레이블이 지정되면 **Mozilla Firefox** 와 같은 웹 브라우저를 사용하여 액세스하려고 할 때 권한이 거부됩니다.



중요

mv 명령을 사용하여 파일과 디렉토리를 이동하면 잘못된 **SELinux** 컨텍스트가 생성되어 **Apache HTTP** 서버 및 **Samba**와 같은 프로세스가 해당 파일 및 디렉터리에 액세스할 수 없습니다.

4.10.3. 기본 SELinux 컨텍스트 확인

matchpathcon 유틸리티를 사용하여 파일과 디렉터리의 **SELinux** 컨텍스트가 올바른지 확인합니다. 이 유틸리티는 시스템 정책을 쿼리한 다음 파일 경로와 연결된 기본 보안 컨텍스트를 제공합니다.[6] 다음 예제에서는 **matchpathcon** 을 사용하여 **/var/www/html/** 디렉토리의 파일에 올바르게 레이블이 지정되어 있는지 확인하는 방법을 보여줍니다.

절차 **4.16. matchpathcon**을 사용하여 기본 **SELinux Conxtext** 확인

1. **root** 사용자로 **/var/www/html/** 디렉터리에 **3**개의 파일**(file1, file2** 및 **file3)**을 만듭니다. 이러한 파일은 **/var/www/html/**:에서 **httpd_sys_content_t** 유형을 상속받습니다.

```
~]# touch /var/www/html/file{1,2,3}
```

```
~]# ls -Z /var/www/html/
```

```
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file1
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file2
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file3
```

2. **root**로 **file1** 유형을 **samba_share_t** 로 변경합니다. **Apache HTTP** 서버는 **samba_share_t** 유형으로 레이블이 지정된 파일 또는 디렉터리를 읽을 수 없습니다.

```
~]# chcon -t samba_share_t /var/www/html/file1
```

3. **matchpathcon -V** 옵션은 현재 **SELinux** 컨텍스트를 **SELinux** 정책의 올바른 기본 컨텍스트와 비교합니다. 다음 명령을 입력하여 **/var/www/html/** 디렉터리에 있는 모든 파일을 확인합니다.

```
~]$ matchpathcon -V /var/www/html/*
/var/www/html/file1 has context unconfined_u:object_r:samba_share_t:s0, should be
system_u:object_r:httpd_sys_content_t:s0
/var/www/html/file2 verified.
/var/www/html/file3 verified.
```

matchpathcon 명령의 다음 출력은 **file1** 이 **samba_share_t** 유형으로 레이블이 지정되어 있지만 **httpd_ sys_content_t** 유형으로 레이블이 지정되어야 한다고 설명합니다.

```
/var/www/html/file1 has context unconfined_u:object_r:samba_share_t:s0, should be
system_u:object_r:httpd_sys_content_t:s0
```

레이블 문제를 해결하고 **root**로 **file1** 에 대한 **Apache HTTP Server** 액세스를 허용하려면 **restorecon** 유틸리티를 사용합니다.

```
~]# restorecon -v /var/www/html/file1
restorecon reset /var/www/html/file1 context unconfined_u:object_r:samba_share_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
```

4.10.4. tar명령을 사용하여 파일 보관

tar 유틸리티는 기본적으로 확장 속성을 유지하지 않습니다. **SELinux** 컨텍스트는 확장된 특성에 저장되므로 파일을 보관할 때 컨텍스트가 손실될 수 있습니다. **tar --selinux** 명령을 사용하여 컨텍스트를 유지하고 아카이브에서 파일을 복원하는 아카이브를 만듭니다. **tar** 아카이브에 확장 속성이 없는 파일이 있거나 확장된 속성을 시스템 기본값과 일치시키려면 **restorecon** 유틸리티를 사용하십시오.

```
~]$ tar -xvf archive.tar | restorecon -f -
```

디렉터리에 따라 **restorecon** 을 실행하려면 **root** 사용자여야 할 수 있습니다.

다음 예제에서는 **SELinux** 컨텍스트를 유지하는 **tar** 아카이브를 생성하는 방법을 보여줍니다.

절차 4.17. tar 아카이브 만들기

1. **/var/www/html/** 디렉토리로 변경하고 **SELinux** 컨텍스트를 확인합니다.

```
~]$ cd /var/www/html/
```

```
html]$ ls -dZ /var/www/html/
drwxr-xr-x. root root system_u:object_r:httpd_sys_content_t:s0 .
```

2. 루트로 **/var/www/html/** 에 **3**개의 파일**(file1, file2** 및 **file3)**을 만듭니다. 이러한 파일은 **/var/www/html/:**에서 **httpd_sys_content_t** 유형을 상속받습니다.

```
html]# touch file{1,2,3}
```

```
html]$ ls -Z /var/www/html/
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file1
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file2
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file3
```

3. 루트로 다음 명령을 입력하여 **test.tar** 이라는 **tar** 아카이브를 만듭니다. **SELinux** 컨텍스트를 유지하려면 **--selinux** 를 사용합니다.

```
html]# tar --selinux -cf test.tar file{1,2,3}
```

4. **root**로 **test/** 라는 새 디렉터리를 생성한 다음 모든 사용자에게 전체 액세스 권한을 허용합니다.

```
~]# mkdir /test
```

```
~]# chmod 777 /test/
```

5. **test.tar** 파일을 **test/:**에 복사합니다.

```
~]$ cp /var/www/html/test.tar /test/
```

6. **test/** 디렉터리로 변경합니다. 이 디렉토리에 다음 명령을 입력하여 **tar** 아카이브를 추출합니다. **SELinux** 옵션을 다시 지정하면 **SELinux** 컨텍스트가 **default_t** 로 변경됩니다.

```
~]$ cd /test/
```

```
test]$ tar --selinux -xvf test.tar
```

7. **SELinux** 컨텍스트 보기. **httpd_sys_content_t** 유형이 **default_t** 로 변경되지 않고 **--selinux** 가 사용되지 않았습니다.

```
test]$ ls -lZ /test/
-rw-r--r--  user1 group1 unconfined_u:object_r:httpd_sys_content_t:s0 file1
-rw-r--r--  user1 group1 unconfined_u:object_r:httpd_sys_content_t:s0 file2
-rw-r--r--  user1 group1 unconfined_u:object_r:httpd_sys_content_t:s0 file3
-rw-r--r--  user1 group1 unconfined_u:object_r:default_t:s0 test.tar
```

8. **test/** 디렉터리가 더 이상 필요하지 않은 경우 **root**로 다음 명령을 입력하여 제거하고 이 디렉터리의 모든 파일을 제거합니다.

```
~]# rm -ri /test/
```

모든 확장 속성을 유지하는 **--xattrs** 옵션과 같이 **tar** 에 대한 자세한 내용은 **tar(1)** 매뉴얼 페이지를 참조하십시오.

4.10.5. 별표로 파일 보관

star 유틸리티는 기본적으로 확장 속성을 유지하지 않습니다. **SELinux** 컨텍스트는 확장된 특성에 저장되므로 파일을 보관할 때 컨텍스트가 손실될 수 있습니다. **star -xattr -H=exustar** 명령을 사용하여 컨텍스트를 유지하는 아카이브를 생성합니다. **star** 패키지는 기본적으로 설치되지 않습니다. **star** 를 설치하려면 **yum install star** 명령을 **root** 사용자로 실행합니다.

다음 예제에서는 **SELinux** 컨텍스트를 유지하는 별 아카이브를 생성하는 방법을 보여줍니다.

절차 **4.18.** 별 아카이브 생성

1. 루트로 **/var/www/html/** 에 **3**개의 파일**(file1, file2** 및 **file3)**을 만듭니다. 이러한 파일은 **/var/www/html/**:에서 **httpd_sys_content_t** 유형을 상속받습니다.

```
~]# touch /var/www/html/file{1,2,3}
```

```
~]# ls -Z /var/www/html/
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file1
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file2
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file3
```

2. **/var/www/html/** 디렉토리로 변경합니다. 이 디렉토리에 **root**로 다음 명령을 입력하여 **test.star**라는 별 아카이브를 생성합니다.

```
~]$ cd /var/www/html
```

```
html]# star -xattr -H=exustar -c -f=test.star file{1,2,3}
star: 1 blocks + 0 bytes (total of 10240 bytes = 10.00k).
```

3. **root**로 **test/** 라는 새 디렉터리를 생성한 다음 모든 사용자에게 전체 액세스 권한을 허용합니다.

```
~]# mkdir /test
```

```
~]# chmod 777 /test/
```

4. 다음 명령을 입력하여 **test.star** 파일을 **test /** 에 복사합니다.

```
~]$ cp /var/www/html/test.star /test/
```

5. **test/** 로 변경합니다. 이 디렉토리에 다음 명령을 입력하여 **star** 아카이브를 추출합니다.

```
~]$ cd /test/
```

```
test]$ star -x -f=test.star
star: 1 blocks + 0 bytes (total of 10240 bytes = 10.00k).
```

6. **SELinux** 컨텍스트 보기. **httpd_sys_content_t** 유형이 **default_t** 로 변경되지 않고 **-xattr -H=exustar** 옵션이 사용되지 않았습니다.

```
~]$ ls -lZ /test/
-rw-r--r--  user1 group1 unconfined_u:object_r:httpd_sys_content_t:s0 file1
-rw-r--r--  user1 group1 unconfined_u:object_r:httpd_sys_content_t:s0 file2
-rw-r--r--  user1 group1 unconfined_u:object_r:httpd_sys_content_t:s0 file3
-rw-r--r--  user1 group1 unconfined_u:object_r:default_t:s0 test.star
```

7. **test/** 디렉터리가 더 이상 필요하지 않은 경우 **root**로 다음 명령을 입력하여 제거하고 이 디렉터리의 모든 파일을 제거합니다.

```
~]# rm -ri /test/
```

8. **root**로 **star** 가 더 이상 필요하지 않은 경우 패키지를 제거합니다.

```
~]# yum remove star
```

별에 대한 자세한 내용은 **star(1)** 도움말 페이지를 참조하십시오.

4.11. 정보 수집 도구

아래에 열거된 유틸리티는 액세스 벡터 캐시 통계 또는 클래스, 유형 또는 부울 수와 같이 잘 포맷된 정보를 제공하는 명령줄 도구입니다.

avcstat

이 명령은 부팅 이후 액세스 벡터 캐시 통계의 짧은 출력을 제공합니다. 시간 간격(초)을 지정하여 실시간으로 통계를 볼 수 있습니다. 이는 초기 출력 이후 업데이트된 통계를 제공합니다. 사용된 통계 파일은 **/sys/fs/selinux/avc/cache_stats** 이며 **-f /path/to/file** 옵션을 사용하여 다른 캐시 파일을 지정할 수 있습니다.

```
~]# avcstat
  lookups       hits     misses     allocs   reclaims      frees
 47517410   47504630      12780      12780      12176      12275
```

seinfo

이 유틸리티는 클래스, 유형, 부울, 허용 규칙 및 기타 항목과 같은 정책 중단을 설명하는 데 유용합니다. **seinfo** 는 **policy.conf** 파일, 바이너리 정책 파일, 모듈식 정책 패키지 목록 또는 정책 목록 파일을 입력으로 사용하는 명령줄 유틸리티입니다. 이러한**info** 유틸리티를 사용하려면 **setools-console** 패키지가 설치되어 있어야 합니다.

seinfo 출력은 바이너리 파일과 소스 파일마다 달라집니다. 예를 들어 정책 소스 파일은 **{ }** 괄호를 사용하여 여러 규칙 요소를 한 줄로 그룹화합니다. 단일 속성이 하나 또는 여러 유형으로 확장되는 속성과 유사한 효과가 발생합니다. 이러한 항목이 확장되어 바이너리 정책 파일에서 더 이상 관련이 없으므로 검색 결과에 반환 값이 **0**입니다. 그러나 대괄호를 사용하여 이전 한 줄 규칙 각각에 따라 룰 수가 크게 증가합니다. 이제 여러 개의 개별 행이 있습니다.

일부 항목은 바이너리 정책에 존재하지 않습니다. 예를 들어, **neverallow** 규칙은 런타임이 아닌 정책 컴파일 중에만 확인되며, 초기 보안 식별자(**SID**)는 부팅 중에 커널에서 로드하는 정책보다 먼저 필요하므로 바이너리 정책의 일부가 아닙니다.

```
~]# seinfo

Statistics for policy file: /sys/fs/selinux/policy
Policy Version & Type: v.28 (binary, mls)

   Classes:            77    Permissions:       229
   Sensitivities:       1    Categories:       1024
   Types:            3001    Attributes:        244
   Users:               9    Roles:              13
   Booleans:          158    Cond. Expr.:       193
   Allow:          262796    Neverallow:          0
   Auditallow:         44    Dontaudit:      156710
   Type_trans:      10760    Type_change:        38
   Type_member:        44    Role allow:         20
   Role_trans:        237    Range_trans:      2546
   Constraints:        62    Validatetrans:       0
   Initial SIDs:       27    Fs_use:             22
   Genfscon:           82    Portcon:           373
   Netifcon:            0    Nodecon:             0
   Permissives:        22    Polcap:              2
```

these info 유틸리티는 도메인 속성이 있는 유형 수를 나열하여 제한된 다양한 프로세스 수를 추정할 수도 있습니다.

```
~]# seinfo -adomain -x | wc -l
550
```

일부 도메인 유형이 제한된 것은 아닙니다. 제한되지 않은 도메인 수를 보려면 **unconfined_domain** 특성을 사용합니다.

```
~]# seinfo -aunconfined_domain_type -x | wc -l
52
```

허용 도메인은 **--permissive** 옵션을 사용하여 계산할 수 있습니다.

```
~]# seinfo --permissive -x | wc -l
31
```

위의 명령에서 추가 **| wc -l** 명령을 제거하여 전체 목록을 확인합니다.

sesearch

sesearch 유틸리티를 사용하여 정책에서 특정 규칙을 검색할 수 있습니다. 정책 소스 파일 또는 바이너리 파일을 검색할 수 있습니다. 예를 들어 다음과 같습니다.

```
~]$ sesearch --role_allow -t httpd_sys_content_t
Found 20 role allow rules:
   allow system_r sysadm_r;
   allow sysadm_r system_r;
   allow sysadm_r staff_r;
   allow sysadm_r user_r;
   allow system_r git_shell_r;
   allow system_r guest_r;
   allow logadm_r system_r;
   allow system_r logadm_r;
   allow system_r nx_server_r;
   allow system_r staff_r;
   allow staff_r logadm_r;
   allow staff_r sysadm_r;
   allow staff_r unconfined_r;
   allow staff_r webadm_r;
   allow unconfined_r system_r;
   allow system_r unconfined_r;
   allow system_r user_r;
   allow webadm_r system_r;
   allow system_r webadm_r;
   allow system_r xguest_r;
```

sesearch 유틸리티는 *허용* 규칙 수를 제공할 수 있습니다.

```
~]# sesearch --allow | wc -l
262798
```

dontaudit 규칙 수입니다.

```
~]# sesearch --dontaudit | wc -l
156712
```

4.12. SELINUX 정책 모듈 우선순위 및 비활성화

/etc/selinux/ 의 **SELinux** 모듈 스토리지에서는 **SELinux** 모듈에서 우선 순위를 사용할 수 있습니다. 다음 명령을 **root**로 입력하여 다른 우선 순위의 두 모듈 디렉터리를 표시합니다.

```
~]# ls /etc/selinux/targeted/active/modules
100  400  disabled
```

semodule 유틸리티에서 사용하는 기본 우선 순위는 **400**이지만 **selinux-policy** 패키지에서 사용하는 우선 순위는 **100**이므로 우선 순위 **100**으로 설치된 대부분의 **SELinux** 모듈을 찾을 수 있습니다.

기존 모듈을 더 높은 우선 순위로 동일한 이름으로 수정된 모듈로 재정의할 수 있습니다. 동일한 이름과 다른 우선 순위가 있는 모듈이 더 많은 경우 정책이 구축될 때 우선 순위가 가장 높은 모듈만 사용됩니다.

예 4.1. SELinux 정책 모듈 우선 순위 사용

수정된 파일 컨텍스트를 사용하여 새 모듈을 준비합니다. **semodule -i** 명령을 사용하여 모듈을 설치하고 모듈의 우선 순위를 **400**으로 설정합니다. 다음 예에서는 **sandbox.pp** 을 사용합니다.

```
~]# semodule -X 400 -i sandbox.pp
~]# semodule --list-modules=full | grep sandbox
400 sandbox          pp
100 sandbox          pp
```

기본 모듈로 돌아가려면 **root**로 **semodule -r** 명령을 입력합니다.

```
~]# semodule -X 400 -r sandbox
libsemanage.semanage_direct_remove_key: sandbox module at priority 100 is now active.
```

시스템 정책 모듈 비활성화

시스템 정책 모듈을 비활성화하려면 **root**로 다음 명령을 입력합니다.

```
semodule -d MODULE_NAME
```

주의

semodule -r 명령을 사용하여 시스템 정책 모듈을 제거하면 시스템 스토리지에서 삭제되며 다시 로드할 수 없습니다. 모든 시스템 정책 모듈을 복원하기 위해 **selinux-policy-targeted** 패키지의 불필요한 재설치를 방지하려면 대신 **semodule -d** 명령을 사용합니다.

4.13. 다단계 보안(MLS)

다단계 보안 기술은 **La Padula** 필수 액세스 모델을 적용하는 보안 체계를 의미합니다. **MLS**에서 사용자와 프로세스는 *제목*이라고 *하며* 시스템의 파일, 장치 및 기타 패시브 구성 요소라고 *합니다*. 제목과 객체 모두 보안 수준으로 레이블이 지정되며, 주제의 모음 또는 객체 분류가 필요합니다. 예를 들어 각 보안 수준은 *민감도* 및 *범주*로 구성됩니다. 예를 들어 내부 릴리스 일정은 기밀 민감도로 내부 문서 범주에 기록됩니다.

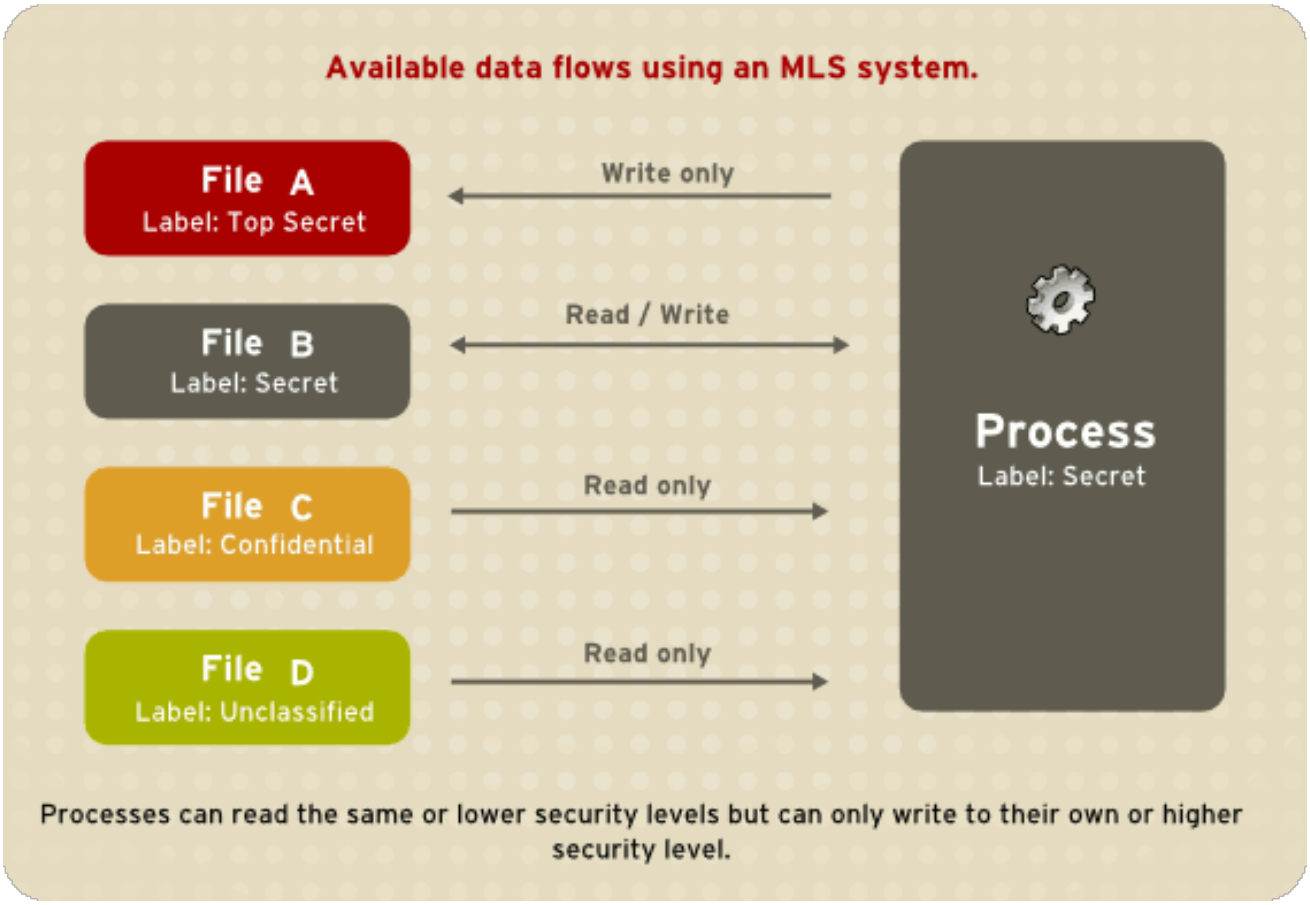
그림 4.1. "마케도니아 레벨"는 원래 미국 국방 커뮤니티에서 디자인한 것으로 나타났습니다. 위의 내부 일정 예와 관련하여 기밀 유지를 받은 사용자만 기밀 카테고리에서 문서를 볼 수 있습니다. 단, 기밀 자료만 가지고 있는 사용자는 더 높은 수준 또는 사진이 필요한 문서를 볼 수 없습니다. 낮은 수준의 문서에만 읽기 액세스 권한이 허용되며, 더 높은 수준의 문서를 사용할 수 있습니다.

그림 4.1. 마케도니아 레벨

[D]

그림 4.2. "MLS를 사용하여 허용된 데이터 흐름"는 **"Secret"** 보안 수준에서 실행되는 주체와 다른 보안 수준의 다양한 개체 간에 허용되는 모든 데이터 흐름을 보여줍니다. 간단히 요약하면 *읽기 가능하고 쓰기가 필요하지 않은 두 가지 속성을 적용할 수 있습니다.*

그림 4.2. MLS를 사용하여 허용된 데이터 흐름



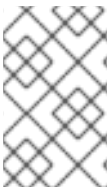
[D]

4.13.1. MLS 및 시스템 권한

MLS 액세스 규칙은 항상 기존 액세스 권한(파일 권한)과 결합됩니다. 예를 들어 **"Secret"**의 보안 수준이 있는 사용자가 **DAC(Despretionary Access Control)**를 사용하여 다른 사용자가 파일에 대한 액세스를 차단하는 경우 **"최고의 비밀"** 수준의 보안 수준이 있는 사용자의 액세스도 차단됩니다. ***DAC*** 규칙 다음에 **SELinux MLS** 정책 규칙이 확인된다는 점을 기억하는 것이 중요합니다. 보안 수준이 높은 경우 파일 시스템을 임의로 검색하는 권한을 자동으로 부여하지 않습니다.

최상위 레벨의 사용자는 다단계 시스템에 대한 관리 권한을 자동으로 획득하지 못합니다. 컴퓨터의 모든 정보에 액세스할 수 있지만 관리 권한을 갖는 것과는 다릅니다.

4.13.2. SELinux에서 MLS 활성화



참고

X Window 시스템을 실행 중인 시스템에서 **MLS** 정책을 사용하지 않는 것이 좋습니다.

다음 단계에 따라 시스템에서 **SELinux MLS** 정책을 활성화합니다.

절차 4.19. SELinux MLS 정책 활성화

1. **selinux-policy-✓s** 패키지를 설치합니다.

```
~]# yum install selinux-policy-mls
```

2. **MLS** 정책을 활성화하기 전에 파일 시스템의 각 파일의 레이블을 **MLS** 레이블로 다시 지정해야 합니다. 파일 시스템의 레이블을 다시 지정하면 제한된 도메인이 액세스가 거부될 수 있으므로 시스템이 올바르게 부팅되지 않을 수 있습니다. 이 문제가 발생하지 않도록 하려면 **/etc/selinux/config** 파일에서 **SELINUX=permissive** 를 구성합니다. 또한 **SELINUXTYPE=✓s** 를 구성하여 **MLS** 정책을 활성화합니다. 구성 파일은 다음과 같아야 합니다.

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
SELINUX=permissive
# SELINUXTYPE= can take one of these two values:
#       targeted - Targeted processes are protected,
#       mls - Multi Level Security protection.
SELINUXTYPE=mls
```

3. **SELinux**가 허용 모드에서 실행 중인지 확인합니다.

```
~]# setenforce 0
```

```
~]$ getenforce
Permissive
```

4. **fixfiles** 스크립트를 사용하여 다음 재부팅 시 파일의 레이블이 다시 지정되도록 **-F** 옵션이 포함된 **/.autorelabel** 파일을 만듭니다.

```
~]# fixfiles -F onboot
```

5. 시스템을 재부팅합니다. 다음 부팅 중에 모든 파일 시스템은 **MLS** 정책에 따라 레이블이 다시 지정됩니다. 레이블 프로세스는 적절한 **SELinux** 컨텍스트를 사용하여 모든 파일의 레이블을 지정합니다.

```
*** Warning -- SELinux mls policy relabel is required.
*** Relabeling could take a very long time, depending on file
*** system size and speed of hard drives.
***********
```

하단 행의 각 * (별표) 문자는 레이블이 지정된 1000개 파일을 나타냅니다. 위의 예에서 11개의 * 문자는 레이블이 지정된 11000개 파일을 나타냅니다. 모든 파일에 레이블을 지정하는 데 걸리는 시간은 시스템의 파일 수와 하드 디스크 드라이브의 속도에 따라 달라집니다. 최신 시스템에서 이 프로세스는 10분도 걸리지 않습니다. 레이블 지정 프로세스가 완료되면 시스템이 자동으로 재부팅됩니다.

6. 허용 모드에서는 SELinux 정책이 적용되지 않지만 강제 모드에서 실행 중인 경우 거부된 작업에 여전히 기록됩니다. 강제 모드로 변경하기 전에 root로 다음 명령을 입력하여 마지막 부팅 중에 SELinux가 작업을 거부하지 않았는지 확인합니다. 마지막 부팅 중에 SELinux에서 작업을 거부하지 않은 경우 이 명령은 출력을 반환하지 않습니다. 부팅 중 SELinux가 액세스를 거부한 경우 문제 해결 정보는 11장. *문제 해결* 을 참조하십시오.

```
~]# grep "SELinux is preventing" /var/log/messages
```

7. /var/log/messages 파일에 거부 메시지가 없거나 기존의 모든 거부를 해결한 경우 /etc/selinux/config 파일에서 SELINUX=enforcing을 구성합니다.

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of these two values:
#       targeted - Targeted processes are protected,
#       mls - Multi Level Security protection.
SELINUXTYPE=mls
```

8. 시스템을 재부팅하고 SELinux가 강제 모드에서 실행 중인지 확인합니다.

```
~]$ getenforce
Enforcing
```

MLS 정책이 활성화되어 있습니다.

```
~]# sestatus |grep mls
Policy from config file:       mls
```

4.13.3. 특정 MLS 범위를 사용하여 사용자 생성

특정 **MLS** 범위를 사용하여 새 **Linux** 사용자를 생성하려면 다음 단계를 따르십시오.

절차 4.20. 특정 MLS 범위를 사용하여 사용자 생성

1. **useradd** 명령을 사용하여 새 **Linux** 사용자를 추가하고 새 **Linux** 사용자를 기존 **SELinux** 사용자(이 경우 **staff_u**)에 매핑합니다.

```
~]# useradd -Z staff_u john
```

2. 새로 생성된 **Linux** 사용자에게 암호를 할당합니다.

```
prompt~]# passwd john
```

3. **SELinux**와 **Linux** 사용자 간의 매핑을 보려면 **root**로 다음 명령을 입력합니다. 출력은 다음과 같아야 합니다.

```
~]# semanage login -l
Login Name           SELinux User         MLS/MCS Range        Service

__default__          user_u               s0-s0                *
john                 staff_u              s0-s15:c0.c1023      *
root                 root                 s0-s15:c0.c1023      *
staff                staff_u              s0-s15:c0.c1023      *
sysadm               staff_u              s0-s15:c0.c1023      *
system_u             system_u             s0-s15:c0.c1023      *
```

4. 사용자 **john** 에 대한 특정 범위를 정의합니다.

```
~]# semanage login --modify --range s2:c100 john
```

5. **SELinux**와 **Linux** 사용자 간의 매핑을 다시 봅니다. 사용자 **john** 은 이제 특정 **MLS** 범위가 정의되었습니다.

```
~]# semanage login -l
Login Name           SELinux User         MLS/MCS Range        Service

__default__          user_u               s0-s0                *
john                 staff_u              s2:c100              *
root                 root                 s0-s15:c0.c1023      *
```

```
staff          staff_u          s0-s15:c0.c1023     *
sysadm          staff_u          s0-s15:c0.c1023     *
system_u          system_u          s0-s15:c0.c1023     *
```

6. 필요한 경우 **john**의 홈 디렉터리에서 레이블을 수정하려면 다음 명령을 입력합니다.

```
~]# chcon -R -l s2:c100 /home/john
```

4.13.4. Polyinstantiated 디렉토리 설정

/tmp 및 **/var/tmp/** 디렉토리는 일반적으로 모든 프로그램, 서비스 및 사용자의 임시 스토리지에 사용됩니다. 그러나 이러한 설정을 사용하면 이러한 디렉터리가 경쟁 조건 공격에 취약하거나 파일 이름에 따라 정보 누출이 발생할 수 있습니다. **SELinux**는 다중 *인스턴스화된 디렉터리의 형태로* 솔루션을 제공합니다. 즉, **/tmp** 및 **/var/tmp** / 를 모두 인스턴스화하여 각 사용자에 대해 개인용으로 표시되도록 합니다. 디렉터리 인스턴스화를 활성화하면 각 사용자의 **/tmp** 및 **/var/tmp** / 디렉토리가 **/tmp -inst** 및 **/var/tmp /tmp-inst** 에 자동으로 마운트됩니다.

다음 단계에 따라 디렉터리의 다중화를 활성화합니다.

절차 4.21. Polyinstantiation 디렉터리 활성화

1. **/etc/security/namespace.conf** 파일의 마지막 **3**행 주석을 제거하여 **/tmp,/var/tmp** / 및 사용자의 홈 디렉터리의 인스턴스화를 활성화합니다.

```
~]$ tail -n 3 /etc/security/namespace.conf
/tmp     /tmp-inst/           level      root,adm
/var/tmp /var/tmp/tmp-inst/    level      root,adm
$HOME    $HOME/$USER.inst/     level
```

2. **/etc/pam.d/login** 파일에서 **pam_namespace.so** 모듈이 **session**에 맞게 구성되어 있는지 확인합니다.

```
~]$ grep namespace /etc/pam.d/login
session    required     pam_namespace.so
```

3. 시스템을 재부팅합니다.

4.14. 파일 이름 전환

파일 이름 전환 기능을 사용하면 정책 작성자가 정책 전환 규칙을 작성할 때 파일 이름을 지정할 수 있

습니다. 다음을 나타내는 규칙을 작성할 수 있습니다. **A_t**라는 프로세스가 **B_t** 레이블이 지정된 디렉터리에 지정된 개체 클래스를 생성하고 지정된 오브젝트 클래스의 이름이 **objectname** 인 경우 레이블 **C_t** 가 됩니다. 이 메커니즘은 시스템의 프로세스에 대해 보다 세밀한 제어를 제공합니다.

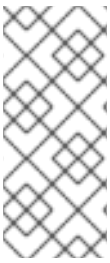
파일 이름 전환이 없으면 다음과 같은 세 가지 방법으로 개체에 레이블을 지정할 수 있습니다.

- 기본적으로 오브젝트는 상위 디렉터리에서 레이블을 상속합니다. 예를 들어 사용자가 **etc_t** 레이블이 지정된 디렉터리에 파일을 만드는 경우 파일에 **etc_t** 도 레이블이 지정됩니다. 그러나 이 방법은 다른 레이블이 있는 디렉터리 내에 여러 파일이 있는 것이 바람직한 경우 사용되지 않습니다.

- 정책 작성자는 다음을 명시하는 정책에 규칙을 작성할 수 있습니다. **A_t** 유형의 프로세스가 **B_t** 라는 레이블이 지정된 디렉터리에 지정된 오브젝트 클래스를 생성하면 오브젝트에 새 **C_t** 레이블이 부여됩니다. 단일 프로그램에서 각 오브젝트에 별도의 레이블이 필요한 동일한 디렉토리에 여러 오브젝트를 생성하는 경우 이 방법이 문제가 됩니다. 또한 생성된 오브젝트의 이름은 지정되지 않으므로 이러한 규칙은 부분적인 제어만 제공합니다.

- 특정 애플리케이션에는 이러한 애플리케이션이 특정 경로의 레이블이 무엇인지 시스템에 질문할 수 있는 **SELinux** 인식 기능이 있습니다. 그런 다음 이러한 애플리케이션은 커널을 요청하여 필수 레이블이 있는 오브젝트를 생성합니다. **SELinux** 인식 기능이 있는 애플리케이션의 예로 **rpm** 패키지 관리자, **restorecon** 유틸리티 또는 **udev** 장치 관리자가 있습니다. 그러나 **SELinux** 인식을 통해 파일 또는 디렉토리를 생성하는 모든 애플리케이션에 지시할 수 없습니다. 종종 생성 후 올바른 레이블을 사용하여 오브젝트의 레이블을 다시 지정해야 합니다. 그렇지 않으면 제한된 도메인에서 오브젝트 사용을 시도하면 **AVC** 메시지가 반환됩니다.

파일 이름 전환 기능은 잘못된 레이블 지정과 관련된 문제를 줄이고 시스템의 보안을 강화합니다. 정책 작성자는 특정 애플리케이션에서 지정된 디렉터리에 지정된 이름으로 파일만 생성할 수 있도록 올바르게 표시될 수 있습니다. 규칙은 파일 경로가 아닌 파일 이름을 고려합니다. 파일 경로의 기본 이름입니다. 파일 이름 전환은 **strcmp()** 함수에 의해 수행된 정확한 일치를 사용합니다. 정규 표현식 또는 와일드카드 문자 사용은 고려되지 않습니다.



참고

파일 경로는 커널에서 다를 수 있으며 파일 이름 전환은 레이블을 결정하는 데 경로를 사용하지 않습니다. 결과적으로 이 기능은 초기 파일 생성에만 영향을 미치며 이미 생성된 오브젝트의 잘못된 레이블을 수정하지 않습니다.

예 4.2. 파일 이름 전환으로 작성된 정책 규칙의 예

아래 예제에서는 파일 이름 전환이 있는 정책 규칙을 보여줍니다.

```
filetrans_pattern(unconfined_t, admin_home_t, ssh_home_t, dir, ".ssh")
```

이 규칙에는 **unconfined_t** 유형의 프로세스가 **admin_ home_t**라는 레이블이 지정된 디렉터리에 **~/.ssh/** 디렉터리가 생성되면 **~/. ssh/** 디렉터리에 **ssh_home_t** 레이블이 지정됩니다.

파일 이름 전환으로 작성된 정책 규칙의 유사한 예는 다음과 같습니다.

```
filetrans_pattern(staff_t, user_home_dir_t, httpd_user_content_t, dir, "public_html")
filetrans_pattern(thumb_t, user_home_dir_t, thumb_home_t, file, "missfont.log")
filetrans_pattern(kernel_t, device_t, xserver_misc_device_t, chr_file, "nvidia0")
filetrans_pattern(puppet_t, etc_t, krb5_conf_t, file, "krb5.conf")
```

참고

파일 이름 전환 기능은 주로 정책 작성자에 영향을 미치지만, 사용자는 포함된 디렉터리의 기본 레이블로 거의 항상 파일 오브젝트가 생성되지 않고 정책에 지정된 것과 다른 레이블이 있는 것을 알 수 있습니다.

4.15. PTRACE() 비활성화

ptrace() 시스템 호출을 사용하면 한 프로세스에서 다른 프로세스의 실행을 관찰하고 제어하고 메모리 및 레지스터를 변경할 수 있습니다. 이 호출은 주로 디버깅 중에 개발자가 사용합니다(예: **strace** 유틸리티 사용 시). **ptrace()** 가 필요하지 않은 경우 시스템 보안을 개선하기 위해 비활성화할 수 있습니다. 이 작업은 **unconfined _t** 도메인에서 실행 중인 프로세스를 거부하는 **deny_ptrace** 부울을 활성화하여 다른 프로세스에서 **ptrace()** 를 사용할 수 없습니다.

deny_ptrace 부울은 기본적으로 비활성화되어 있습니다. 활성화하려면 **root** 사용자로 **setsebool -P deny_ptrace**를 실행합니다.

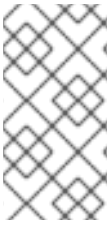
```
~]# setsebool -P deny_ptrace on
```

이 부울이 활성화되었는지 확인하려면 다음 명령을 사용합니다.

```
~]$ getsebool deny_ptrace
deny_ptrace --> on
```

이 부울을 비활성화하려면 **setsebool -P deny_ptrace off** 명령을 **root**로 실행합니다.

```
~]# setsebool -P deny_ptrace off
```



참고

setsebool -P 명령은 영구적으로 변경합니다. 재부팅 시 변경 사항이 유지되지 않도록 하려면 **-P** 옵션을 사용하지 마십시오.

이 부울은 **Red Hat Enterprise Linux**의 일부인 패키지에만 영향을 미칩니다. 결과적으로 타사 패키지는 **ptrace()** 시스템 호출을 사용할 수 있었습니다. **ptrace()** 를 사용할 수 있는 모든 도메인을 나열하려면 다음 명령을 입력합니다. **setools-console** 패키지는 **sesearch** 유틸리티를 제공하며 패키지는 기본적으로 설치되지 않습니다.

```
~]# sesearch -A -p ptrace,sys_ptrace -C | grep -v deny_ptrace | cut -d ' ' -f 5
```

4.16. 축소네일 보호

축소판 아이콘은 공격자가 **USB** 장치 또는 **CD**와 같은 이동식 미디어를 사용하여 잠긴 시스템으로 침입할 수 있습니다. 이동식 미디어를 감지하면 **Nautilus** 파일 관리자에서 축소판 드라이버 코드를 실행하여 시스템이 잠긴 경우에도 적절한 파일 브라우저에 축소판 아이콘을 표시합니다. 이 동작은 축소판 실행 파일이 취약한 경우 공격자가 암호에 들어가지 않고 잠금 화면을 우회하기 위해 축소판 드라이버 코드를 사용할 수 있기 때문에 안전하지 않습니다.

따라서 이러한 공격을 방지하기 위해 새로운 **SELinux** 정책을 사용합니다. 이 정책은 화면이 잠길 때 모든 축소판 드라이버가 잠길 수 있도록 합니다. 제한된 사용자와 제한되지 않은 사용자 모두에 대해 축소판 보호가 활성화됩니다. 이 정책은 다음 애플리케이션에 영향을 미칩니다.

- **/usr/bin/evince-thumbnailer**
- **/usr/bin/ffmpegthumbnailer**

- /usr/bin/gnome-exe-thumbnailer.sh
- /usr/bin/gnome-nds-thumbnailer
- /usr/bin/gnome-xcf-thumbnailer
- /usr/bin/gsf-office-thumbnailer
- /usr/bin/raw-thumbnailer
- /usr/bin/shotwell-video-thumbnailer
- /usr/bin/totem-video-thumbnailer
- /usr/bin/whaaw-thumbnailer
- /usr/lib/tumbler-1/tumblerd
- /usr/lib64/tumbler-1/tumblerd

---

[3] 일시적으로 기본 동작으로 되돌리려면 Linux root 사용자로 setsebool httpd_can_network_connect_db off 명령을 실행합니다. 재부팅 시 지속되는 변경 사항은 setsebool -P httpd_can_network_connect_db off 명령을 실행합니다.

[4] /etc/selinux/targeted/contexts/files/ 디렉토리에 있는 파일은 파일 및 디렉터리에 대한 컨텍스트를 정의합니다. 이 디렉터리의 파일은 restorecon 및 setfiles 유틸리티에서 읽어 파일과 디렉터리를 기본 컨텍스트로 복원합니다.

[5] Morris, James. "SELinux에서 파일 시스템 레이블 지정". 2004년 10월 1일 게시됨. 2008년 10월 14일 확인됨: http://www.linuxjournal.com/article/7426.

[6]

matchpathcon 에 대한 자세한 내용은 **matchpathcon(8)** 도움말 페이지를 참조하십시오.

5장. SEPOLICY SUITE

sepolicy 유틸리티는 설치된 **SELinux** 정책을 쿼리할 수 있는 기능 집합을 제공합니다. 이러한 기능은 **sepolgen** 또는 **setrans** 와 같은 별도의 유틸리티에서 새로 제공되었거나 이전에 제공되었습니다. 모음을 사용하면 전환 보고서, 도움말 페이지 또는 새로운 정책 모듈을 생성할 수 있으므로 사용자가 더 쉽게 액세스할 수 있고 **SELinux** 정책을 더 잘 이해할 수 있습니다.

policycoreutils-devel 패키지는 **sepolicy** 를 제공합니다. 다음 명령을 **root** 사용자로 입력하여 **sepolicy** 를 설치합니다.

```
~]# yum install policycoreutils-devel
```

sepolicy 제품군은 명령줄 매개 변수로 호출되는 다음 기능을 제공합니다.

표 **5.1. sepolicy** 기능

기능	Description
부울	부울 설명을 보려면 SELinux 정책을 쿼리합니다.
통신	SELinux 정책을 쿼리하여 도메인이 서로 통신할 수 있는지 확인합니다.
generate	SELinux 정책 모듈 템플릿 생성
GUI	SELinux 정책 용 그래픽 사용자 인터페이스
인터페이스	SELinux 정책 인터페이스 나열
manpage	SELinux man 페이지 생성
네트워크	SELinux 정책 네트워크 정보 쿼리
전환	SELinux 정책을 쿼리하고 프로세스 전환 보고서 생성

5.1. SEPOLICY PYTHON 바인딩

이전 버전의 **Red Hat Enterprise Linux**에서는 **setools** 패키지에 **se search** 및 **seinfo** 유틸리티가 포함되어 있었습니다. **sesearch** 유틸리티는 **SELinux** 정책의 규칙을 검색하는 데 사용되며, 이러한**info** 유틸리티를 사용하면 정책의 다양한 다른 구성 요소를 쿼리할 수 있습니다.

Red Hat Enterprise Linux 7에서는 **se policy** 제품군을 통해 이러한 유틸리티의 기능을 사용할 수 있도록 **se search** 및 **seinfo** 에 대한 **Python** 바인딩이 추가되었습니다. 아래 예제를 참조하십시오.

```
> python
>>> import sepolicy
>>> sepolicy.info(sepolicy.ATTRIBUTE)
Returns a dictionary of all information about SELinux Attributes
>>>sepolicy.search([sepolicy.ALLOW])
Returns a dictionary of all allow rules in the policy.
```

5.2. SELINUX 정책 모듈 생성: SEPOLICY GENERATE

이전 버전의 **Red Hat Enterprise Linux**에서는 **SELinux** 정책을 생성하는 데 **sepolgen** 또는 **selinux-polgengui** 유틸리티가 사용되었습니다. 이러한 툴이 **sepolicy** 제품군에 병합되었습니다. **Red Hat Enterprise Linux 7**에서 **sepolicy generate** 명령은 초기 **SELinux** 정책 모듈 템플릿을 생성하는 데 사용됩니다.

sepolgen 과 달리 **sepolicy generate**를 **root** 사용자로 실행할 필요가 없습니다. 이 유틸리티는 **RPM** 사양 파일도 생성합니다. 이 파일은 정책 패키지 파일(**NAME.*pp***) 및 인터페이스 파일(***NAME. if***)을 올바른 위치에 설치하는 ***RPM*** 패키지를 빌드하는 데 사용할 수 있으며, ***SELinux*** 정책을 커널에 설치하고 레이블을 수정합니다. 설정 스크립트는 ***SELinux*** 정책을 계속 설치하고 레이블을 설정합니다. 또한 ***sepolicy manpage*** 명령을 사용하여 설치된 정책을 기반으로 도움말 페이지가 생성됩니다. [7] 마지막으로 ***sepolicy*** 는 다른 시스템에 설치할 준비가 된 ***SELinux*** 정책 및 수동 페이지를 ***RPM*** 패키지로 빌드하고 컴파일합니다.

sepolicy 생성이 실행되면 다음 파일이 생성됩니다.

NAME.te - 강제 파일 입력

이 파일은 특정 도메인의 모든 유형과 규칙을 정의합니다.

NAME.if - 인터페이스 파일

이 파일은 시스템의 기본 파일 컨텍스트를 정의합니다. ***NAME.te*** 파일에 생성된 파일 유형을 가져와서 파일 경로를 유형에 연결합니다. ***restorecon*** 및 ***rpm*** 과 같은 유틸리티는 이러한 경로를 사용하여 레이블을 작성합니다.

NAME_selinux.spec - RPM 사양 파일

이 파일은 ***SELinux*** 정책을 설치하고 레이블을 설정하는 ***RPM*** 사양 파일입니다. 이 파일은 또한 정책을 설명하는 인터페이스 파일과 도움말 페이지를 설치합니다. ***sepolicy manpage -d NAME*** 명령을 사용하여 도움말 페이지를 생성할 수 있습니다.

NAME.sh - 도우미 쉘 스크립트

이 스크립트는 시스템의 레이블을 컴파일, 설치 및 수정하는 데 도움이 됩니다. 또한 설치된 정책, 컴파일 및 기타 시스템에 설치하는 데 적합한 **RPM** 패키지를 기반으로 도움말 페이지를 생성합니다.

SELinux 정책 모듈을 생성할 수 있는 경우 **sepolicy generate** 는 소스 도메인에서 대상 도메인으로 생성된 모든 경로를 출력합니다. **sepolicy generate** 에 대한 자세한 내용은 **sepolicy-generate(8)** 도움말 페이지를 참조하십시오.

5.3. 도메인 전환 이해: SEPOLICY 전환

이전에는 **setrans** 유틸리티를 사용하여 두 도메인 또는 프로세스 유형 간 전환이 가능한지 여부를 검사하고 이러한 도메인 또는 프로세스 간에 전환하는 데 사용되는 모든 중간 유형을 출력했습니다. **Red Hat Enterprise Linux 7**에서 **setrans** 은 **sepolicy** 제품군의 일부로 제공되며 이제 **sepolicy transition** 명령이 대신 사용됩니다.

sepolicy transition 명령은 **SELinux** 정책을 쿼리하고 프로세스 전환 보고서를 생성합니다. **sepolicy transition** 명령에는 소스 도메인(**- s** 옵션으로 지정)과 대상 도메인(**- t** 옵션으로 지정)이라는 두 개의 명령줄 인수가 필요합니다. 소스 도메인만 입력한 경우 **sepolicy** 전환 은 소스 도메인에서 전환할 수 있는 모든 가능한 도메인을 나열합니다. 다음 출력에는 모든 항목이 포함되어 있지 않습니다. “@” 문자는 “다음을 실행” 함을 의미합니다.

```
~]$ sepolicy transition -s httpd_t
httpd_t @ httpd_suexec_exec_t --> httpd_suexec_t
httpd_t @ mailman_cgi_exec_t --> mailman_cgi_t
httpd_t @ abrt_retrace_worker_exec_t --> abrt_retrace_worker_t
httpd_t @ dirsrvadmin_unconfined_script_exec_t --> dirsrvadmin_unconfined_script_t
httpd_t @ httpd_unconfined_script_exec_t --> httpd_unconfined_script_t
```

대상 도메인을 지정하면 **sepolicy** 전환 은 소스 도메인에서 대상 도메인으로의 모든 전환 경로에 대한 **SELinux** 정책을 검사하고 이러한 경로를 나열합니다. 다음 출력은 완료되지 않았습니다.

```
~]$ sepolicy transition -s httpd_t -t system_mail_t
httpd_t @ exim_exec_t --> system_mail_t
httpd_t @ courier_exec_t --> system_mail_t
httpd_t @ sendmail_exec_t --> system_mail_t
httpd_t ... httpd_suexec_t @ sendmail_exec_t --> system_mail_t
httpd_t ... httpd_suexec_t @ exim_exec_t --> system_mail_t
httpd_t ... httpd_suexec_t @ courier_exec_t --> system_mail_t
httpd_t ... httpd_suexec_t ... httpd_mojomojo_script_t @ sendmail_exec_t --> system_mail_t
```

sepolicy 전환에 대한 자세한 내용은 **sepolicy-transition(8)** 도움말 페이지를 참조하십시오.

5.4. 수동 페이지 생성: SEPOLICY MANPAGE

sepolicy manpage 명령은 프로세스 도메인을 문서화하는 **SELinux** 정책을 기반으로 도움말 페이지를 생성합니다. 따라서 이러한 설명서는 항상 최신 상태입니다. 자동으로 생성된 도움말 페이지의 각 이름은 프로세스 도메인 이름과 **_selinux** 접미사(예: **httpd_selinux**)로 구성됩니다.

도움말 페이지에는 제한된 도메인에 대한 **SELinux** 정책의 다양한 부분에 대한 정보를 제공하는 여러 섹션이 포함되어 있습니다.

- **Entrypoints** 섹션에는 도메인 전환 중에 실행해야 하는 모든 실행 파일이 포함되어 있습니다.
- **Process Types**(프로세스 유형) 섹션에는 대상 도메인과 동일한 접두사로 시작하는 모든 프로세스 유형이 나열됩니다.
- 부울 섹션에는 도메인과 연결된 부울이 나열됩니다.
- **Port Types**(포트 유형) 섹션에는 도메인과 동일한 접두사와 일치하는 포트 유형이 포함되어 있으며 이러한 포트 유형에 할당된 기본 포트 번호를 설명합니다.
- **Managed Files** 섹션에서는 도메인이 쓸 수 있는 유형과 이러한 유형과 관련된 기본 경로를 설명합니다.
- 파일 컨텍스트 섹션에 는 도메인과 연결된 모든 파일 유형이 포함되어 있으며 이러한 파일 유형을 시스템의 기본 경로 레이블 지정과 함께 사용하는 방법을 설명합니다.
- 공유 파일 섹션에서는 **public_content_t** 와 같은 도메인 공유 유형을 사용하는 방법을 설명합니다.

sepolicy manpage 에 대한 자세한 내용은 **sepolicy-manpage(8)** 매뉴얼 페이지를 참조하십시오.

---

[7] **sepolicy manpage** 에 대한 자세한 내용은 5.4절. "수동 페이지 생성: sepolicy manpage" 을 참조하십시오.

6장. 사용자 제한

Red Hat Enterprise Linux에서 사용자는 기본적으로 SELinux unconfined_u 사용자에게 매핑됩니다. unconfined_u 에 의해 실행되는 모든 프로세스는 unconfined_t 도메인에 있습니다. 즉, 사용자는 표준 Linux DAC 정책의 한도 내에서 시스템 전체에 액세스할 수 있습니다. 그러나 Red Hat Enterprise Linux에서는 수많은 제한된 SELinux 사용자를 사용할 수 있습니다. 즉, 사용자는 제한된 기능 집합으로 제한할 수 있습니다. 각 Linux 사용자는 SELinux 정책을 사용하여 SELinux 사용자에게 매핑되므로 Linux 사용자는 다음을 수행할 수 없는 SELinux 사용자(예:)에 지정된 제한 사항을 상속할 수 있습니다.

- X 윈도우 시스템 실행
- 네트워킹 사용
- setuid 응용 프로그램 실행 (SELinux 정책이 허용하지 않는 경우)
- 또는 su 및 sudo 명령을 실행합니다.

예를 들어 SELinux user_u 사용자가 실행하는 프로세스는 user_t 도메인에 있습니다. 이러한 프로세스는 네트워크에 연결할 수 있지만 su 또는 sudo 명령을 실행할 수 없습니다. 이렇게 하면 사용자로부터 시스템을 보호하는 데 도움이 됩니다. 제한된 사용자 및 해당 기능에 대한 자세한 내용은 3.3절. “제한된 사용자 및 제한되지 않은 사용자”, 표 3.1. “SELinux 사용자 기능” 을 참조하십시오.

6.1. LINUX 및 SELINUX 사용자 맵핑

root로 다음 명령을 입력하여 Linux 사용자와 SELinux 사용자 간의 매핑을 확인합니다.

```
~]# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

Red Hat Enterprise Linux에서 Linux 사용자는 기본적으로 SELinux __default__ login (SELinux unconfined_u 사용자에게 매핑됨)에 매핑됩니다. useradd 명령을 사용하여 Linux 사용자를 만들면 옵션이 지정되지 않은 경우 SELinux unconfined_u 사용자에게 매핑됩니다. 다음은 default-mapping을 정의합니다.

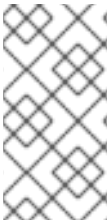
```
__default__          unconfined_u         s0-s0:c0.c1023       *
```

6.2. 새로운 LINUX 사용자 제한: USERADD

SELinux unconfined_u 사용자에 매핑된 Linux 사용자는 unconfined_t 도메인에서 실행됩니다. 이는 unconfined_u 에 매핑된 Linux 사용자로 로그인한 동안 id -Z 명령을 실행하면 표시됩니다.

```
~]$ id -Z
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Linux 사용자가 unconfined_t 도메인에서 실행되는 경우 SELinux 정책 규칙이 적용되지만 unconfined _t 도메인에서 실행되는 Linux 사용자가 거의 모든 액세스를 허용하는 정책 규칙이 있습니다. 제한되지 않은 Linux 사용자가 SELinux 정책에서 정의한 애플리케이션을 unconfined_t 도메인에서 제한된 도메인으로 전환할 수 있는 경우 제한되지 않은 Linux 사용자에게는 제한된 도메인의 제한이 있습니다. 이 경우의 보안 이점은 Linux 사용자가 제한되지 않았더라도 애플리케이션이 제한된 상태로 유지되므로 애플리케이션에서 결함을 악용하는 것은 정책에 의해 제한될 수 있다는 것입니다.



참고

이렇게 하면 사용자로부터 시스템을 보호하지 않습니다. 대신, 사용자 및 시스템은 애플리케이션의 결함으로 인해 발생할 수 있는 손상으로부터 보호되고 있습니다.

useradd 명령으로 Linux 사용자를 만드는 경우 -Z 옵션을 사용하여 매핑되는 SELinux 사용자를 지정합니다. 다음 예제에서는 새 Linux 사용자 useruuser 를 만들고 해당 사용자를 SELinux user_u 사용자에게 매핑합니다. SELinux user_u 사용자에 매핑된 Linux 사용자는 user_t 도메인에서 실행됩니다. 이 도메인에서는 SELinux 정책(예: passwd)에서 setuid 애플리케이션을 실행할 수 없으며 su 또는 sudo 명령을 실행할 수 없으므로 이러한 명령을 사용하여 root 사용자가 되지 않도록 합니다.

절차 6.1. 새 Linux 사용자를 user_u SELinux 사용자로 제한

1. **root로 SELinux user _u 사용자에 매핑되는 새 Linux 사용자 (useruuser)를 만듭니다.**

```
~]# useradd -Z user_u useruuser
```

2. **useruuser 와 user_u 간의 매핑을 보려면 root로 다음 명령을 입력합니다.**

```
~]# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service
```

```
__default__          unconfined_u        s0-s0:c0.c1023      *
root                 unconfined_u        s0-s0:c0.c1023      *
system_u             system_u            s0-s0:c0.c1023      *
useruuser            user_u              s0                  *
```

3. ***root*로 *Linux useruuser* 사용자에게 암호를 할당합니다.**

```
~]# passwd useruuser
Changing password for user useruuser.
New password: Enter a password
Retype new password: Enter the same password again
passwd: all authentication tokens updated successfully.
```

4. *현재 세션에서 로그아웃한 다음 **Linux useruuser** 사용자로 로그인합니다. 로그인하면 **pam_selinux** 모듈은 **Linux** 사용자를 **SELinux** 사용자(이 경우 **user_u**)에 매핑하고 결과 **SELinux** 컨텍스트를 설정합니다. 그런 다음 **Linux** 사용자의 쉘이 이 컨텍스트를 사용하여 시작됩니다. 다음 명령을 입력하여 **Linux** 사용자의 컨텍스트를 확인합니다.*

```
~]$ id -Z
user_u:user_r:user_t:s0
```

5. ***Linux useruuser** 의 세션에서 로그아웃한 후 계정으로 다시 로그인합니다. **Linux useruuser** 사용자를 원하지 않으려면 **root**로 홈 디렉토리와 함께 다음 명령을 입력합니다.*

```
~]# userdel -Z -r useruuser
```

6.3. 기존 LINUX 사용자 제한: SEMANAGE LOGIN

***Linux** 사용자가 **SELinux unconfined_u** 사용자(기본 동작)에 매핑되고 매핑된 **SELinux** 사용자를 변경하려는 경우 **semanage login** 명령을 사용합니다. 다음 예제에서는 **newuser** 라는 새 **Linux** 사용자를 생성한 다음 해당 **Linux** 사용자를 **SELinux user_u** 사용자에게 매핑합니다.*

*절차 **6.2. SELinux** 사용자에게 **Linux** 사용자 매핑*

1. ***root*로 새 *Linux* 사용자(*newuser*)를 만듭니다. 이 사용자는 기본 매핑을 사용하므로 **semanage login -l** 출력에 표시되지 않습니다.**

```
~]# useradd newuser
```

```
~]# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service
```

```
__default__          unconfined_u         s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

2. ***Linux newuser** 사용자를 **SELinux user_u** 사용자에게 매핑하려면 **root**로 다음 명령을 입력합니다.*

```
~]# semanage login -a -s user_u newuser
```

***a** 옵션은 새 레코드를 추가하고 **-s** 옵션은 **Linux** 사용자를 매핑할 **SELinux** 사용자를 지정합니다. 마지막 인수 **newuser** 는 지정된 **SELinux** 사용자에 매핑하려는 **Linux** 사용자입니다.*

3. ***Linux newuser** 사용자와 **user_u** 간의 매핑을 보려면 **semanage** 유틸리티를 다시 사용합니다.*

```
~]# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *
newuser              user_u               s0                   *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

4. ***root**로 **Linux newuser** 사용자에게 암호를 할당합니다.*

```
~]# passwd newuser
Changing password for user newuser.
New password: Enter a password
Retype new password: Enter the same password again
passwd: all authentication tokens updated successfully.
```

5. *현재 세션에서 로그아웃한 다음 **Linux newuser** 사용자로 로그인합니다. 새 사용자의 **SELinux** 컨텍스트를 보려면 다음 명령을 입력합니다.*

```
~]$ id -Z
user_u:user_r:user_t:s0
```

6. ***Linux newuser** 의 세션에서 로그아웃한 후 계정으로 다시 로그인합니다. **Linux newuser** 사용자를 사용하지 않으려면 **root**로 홈 디렉터리와 함께 다음 명령을 입력합니다.*

```
~]# userdel -r newuser
```

root로서 **Linux newuser** 사용자와 **user_u** 간의 매핑을 제거합니다.

```
~]# semanage login -d newuser
```

```
~]# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

6.4. 기본 맵핑 변경

Red Hat Enterprise Linux에서 **Linux** 사용자는 기본적으로 **SELinux __default__ login (SELinux unconfined_u** 사용자에게 매핑됨)에 매핑됩니다. 기본적으로 **SELinux** 사용자에게 매핑되지 않은 신규 **Linux** 사용자 및 **Linux** 사용자는 **semanage login** 명령을 사용하여 기본 매핑을 변경합니다.

예를 들어 **root**로 다음 명령을 입력하여 기본 매핑을 **unconfined_u**에서 **user_u** 로 변경합니다.

```
~]# semanage login -m -S targeted -s "user_u" -r s0 __default__
```

__default__ 로그인이 **user_u** 에 매핑되었는지 확인합니다.

```
~]# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          user_u               s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

새 **Linux** 사용자가 생성되고 **SELinux** 사용자가 지정되지 않거나 기존 **Linux** 사용자가 로그인하여 **semanage login -l** 출력의 특정 항목과 일치하지 않는 경우 **__default__** 로그인에 따라 **user_u** 에 매핑됩니다.

기본 동작으로 다시 변경하려면 **root**로 다음 명령을 입력하여 **__default__** 로그인을 **SELinux unconfined_u** 사용자에게 매핑합니다.

```
~]# semanage login -m -S targeted -s "unconfined_u" -r s0-s0:c0.c1023 __default__
```

6.5. XGUEST: 키오스크 모드

xguest 패키지는 **kiosk** 사용자 계정을 제공합니다. 이 계정은 라이브러리, 은행, 차량, 정보 키오스크, 커피 등 사람들이 사용하는 시스템을 보호하는 데 사용됩니다. **kiosk** 사용자 계정은 매우 제한적입니다. 기본적으로 사용자만 로그인하여 **Firefox** 를 사용하여 인터넷 웹사이트를 탐색할 수 있습니다. 게스트 사용자는 **xguest_u** 에 할당되어 있으며 표 3.1. "SELinux 사용자 기능" 참조하십시오. 파일 생성 또는 설정 변경과 같이 이 계정으로 로그인하는 동안 변경한 내용은 로그아웃 시 손실됩니다.

키오스크 계정을 설정하려면 다음을 수행합니다.

1. **root**로 **xguest** 패키지를 설치합니다. 필요에 따라 종속성을 설치합니다.

   ```
   ~]# yum install xguest
   ```

2. 다양한 사용자가 키오스크 계정을 사용할 수 있도록 하려면 계정은 암호로 보호되지 않으므로 **SELinux**가 강제 모드로 실행 중인 경우에만 계정을 보호할 수 있습니다. 이 계정으로 로그인하기 전에 **getenforce** 유틸리티를 사용하여 **SELinux**가 강제 모드에서 실행 중인지 확인합니다.

   ```
   ~]$ getenforce
   Enforcing
   ```

   강제 모드로 변경하는 데 대한 정보는 4.4절. "SELinux 상태 및 모드의 영구 변경" 을 참조하십시오. **SELinux**가 허용 모드이거나 비활성화된 경우 이 계정으로 로그인할 수 없습니다.

3. **GDM(GNOME Display Manager)**을 사용하면 이 계정에만 로그인할 수 있습니다. **xguest** 패키지가 설치되면 게스트 계정이 **GDM** 로그인 화면에 추가됩니다.

6.6. 사용자 애플리케이션 실행 부울

Linux 사용자가 홈 디렉터리 및 **/tmp** 디렉토리에서 애플리케이션(사용자 권한을 상속함) 및 **/tmp** 디렉토리에 있는 애플리케이션을 실행하지 않도록 하면 결함이 있거나 악성 애플리케이션이 사용자가 소유한 파일을 수정하지 않도록 할 수 있습니다.

부울은 이 동작을 변경할 수 있으며, **root**로 실행해야 하는 **setsebool** 유틸리티로 구성됩니다. **setsebool -P** 명령은 영구적으로 변경합니다. 재부팅 시 변경 사항을 유지하지 않으려면 **-P** 옵션을 사용

하지 마십시오.

guest_t

***guest_t** 도메인의 **Linux** 사용자가 홈 디렉토리 및 **/tmp** 에서 애플리케이션을 실행하지 못하도록 하려면 다음을 수행합니다.*

```
~]# setsebool -P guest_exec_content off
```

xguest_t

***xguest_t** 도메인의 **Linux** 사용자가 홈 디렉토리 및 **/tmp** 에서 애플리케이션을 실행하지 못하도록 하려면 다음을 수행합니다.*

```
~]# setsebool -P xguest_exec_content off
```

user_t

***user_t** 도메인의 **Linux** 사용자가 홈 디렉토리 및 **/tmp** 에서 애플리케이션을 실행하지 못하도록 하려면 :*

```
~]# setsebool -P user_exec_content off
```

staff_t

***staff_t** 도메인의 **Linux** 사용자가 홈 디렉토리 및 **/tmp** 에서 애플리케이션을 실행하지 못하도록 하려면 :*

```
~]# setsebool -P staff_exec_content off
```

***staff_exec_content** 부울을 및 을 활성화하여 **staff_t** 도메인의 **Linux** 사용자가 홈 디렉토리 및 **/tmp** 에서 애플리케이션을 실행할 수 있도록 허용하려면 다음을 수행하십시오.*

```
~]# setsebool -P staff_exec_content on
```

7장. SANDBOX를 사용하여 프로그램 보안 설정

샌드박스 보안 유틸리티는 시스템 관리자가 엄격하게 제한된 **SELinux** 도메인 내에서 애플리케이션을 실행할 수 있는 **SELinux** 정책 집합을 추가합니다. 새 파일을 열거나 네트워크에 대한 액세스 권한에 대한 권한 제한을 정의할 수 있습니다. 이를 통해 시스템에 손상을 초래하지 않고 신뢰할 수 없는 소프트웨어의 처리 특성을 안전하게 테스트할 수 있습니다.

7.1. SANDBOX를 사용하여 애플리케이션 실행

sandbox 유틸리티를 사용하기 전에 **policycoreutils-sandbox** 패키지를 설치해야 합니다.

```
~]# yum install policycoreutils-sandbox
```

애플리케이션을 제한하는 기본 구문은 다음과 같습니다.

```
~]$ sandbox [options] application_under_test
```

샌드박스 에서 그래픽 애플리케이션을 실행하려면 **-X** 옵션을 사용합니다. 예를 들어 다음과 같습니다.

```
~]$ sandbox -X evince
```

X는 필요한 리소스를 복사하고 사용자의 홈 디렉토리 또는 **/tmp** 디렉토리에 폐쇄된 가상 환경을 만들기 전에 애플리케이션에 대해 제한된 보조 **X** 서버(이 경우 **evince**)를 설정하도록 샌드박스 에 지시합니다.

한 세션에서 다음 세션으로 데이터를 보존하려면 다음을 수행합니다.

```
~]$ sandbox -H sandbox/home -T sandbox/tmp -X firefox
```

sandbox/home 은 **/home** 에 사용되며 **sandbox/tmp** 는 **/tmp** 에 사용됩니다. 다른 애플리케이션은 다양한 제한된 환경에 배치됩니다. 애플리케이션은 전체 화면 모드로 실행되므로 다른 기능에 액세스할 수 없습니다. 앞에서 언급한 것처럼 **sandbox_x_file_t**로 레이블이 지정된 파일을 제외하고는 파일을 열거나 생성할 수 없습니다.

*처음에는 샌드박스 내에서도 네트워크에 액세스할 수 없습니다. 액세스를 허용하려면 **sandbox_web_t** 레이블을 사용합니다. 예를 들어 **Firefox** 를 시작하려면 다음을 수행합니다.*

```
~]$ sandbox -X -t sandbox_web_t firefox
```

주의

***sandbox_net_t** 레이블을 사용하면 모든 네트워크 포트에 무제한 양방향 네트워크 액세스가 가능합니다. **sandbox_web_t** 는 웹 검색에만 필요한 포트에 대한 연결을 허용합니다.*

***sandbox_net_t** 는 필요한 경우에만 주의해서 사용해야 합니다.*

*자세한 내용은 샌드박스 **(8)** 매뉴얼 페이지와 사용 가능한 옵션의 전체 목록을 참조하십시오.*

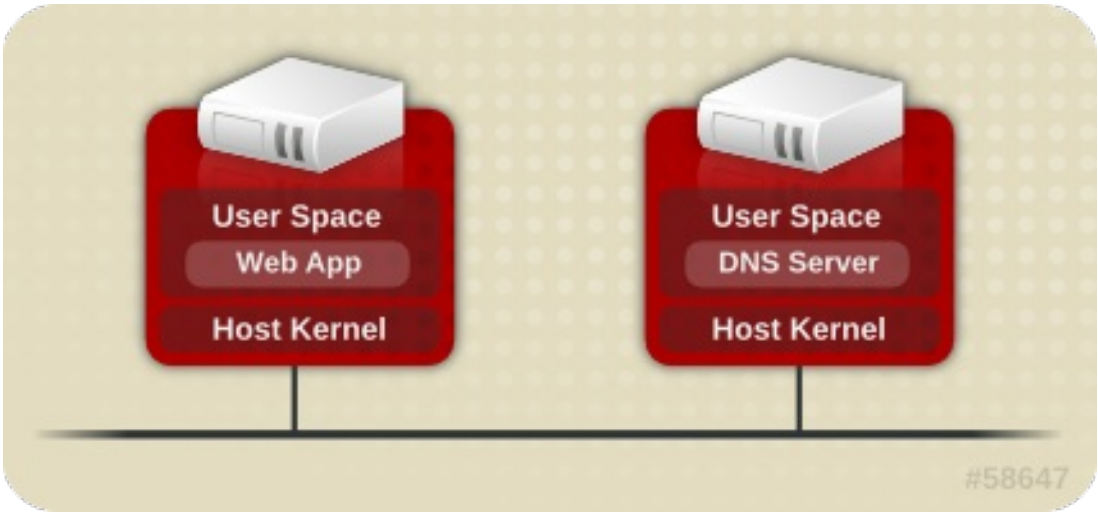
8장. SVIRT

sVirt는 **SELinux**와 가상화를 통합하는 **Red Hat Enterprise Linux**에 포함된 기술입니다. **sVirt**는 가상 시스템을 사용할 때 보안을 개선하기 위해 **MAC**(강제적 액세스 제어)를 적용합니다. 이러한 기술을 통합하는 주된 이유는 호스트 또는 다른 가상 시스템을 대상으로 하는 공격 벡터로 사용될 수 있는 하이퍼바이저의 버그에 대비하여 보안을 개선하고 시스템을 강화하는 것입니다.

이 장에서는 **sVirt**가 **Red Hat Enterprise Linux**의 가상화 기술과 통합하는 방법에 대해 설명합니다.

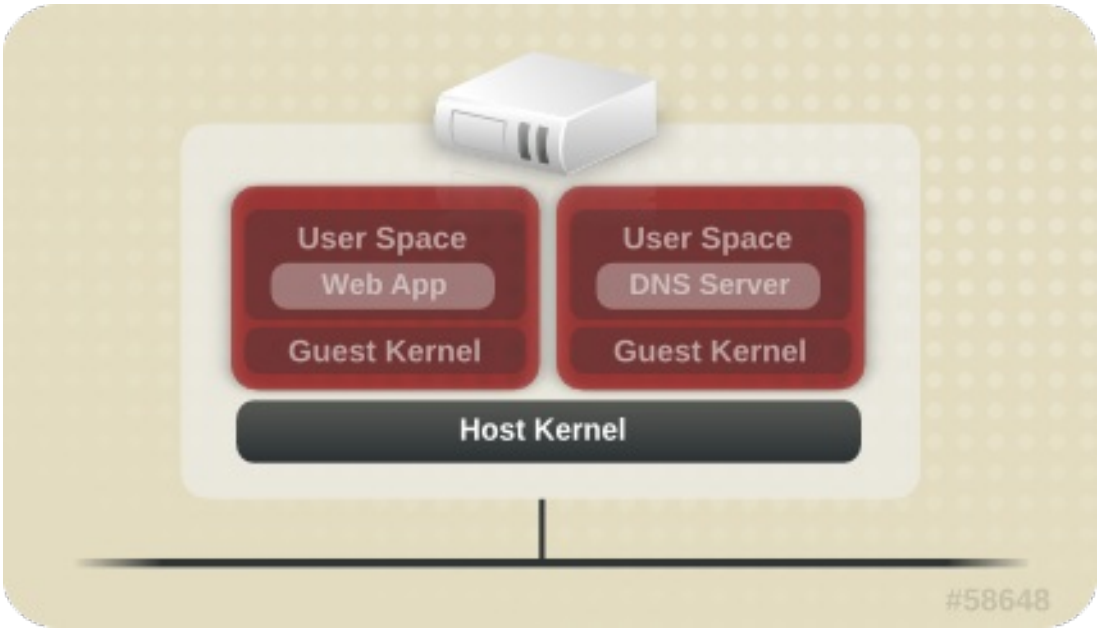
비가상화 환경

비가상화 환경에서 호스트는 물리적으로 서로 분리되며 각 호스트에는 웹 서버 또는 **DNS** 서버와 같은 서비스로 구성된 자체 포함 환경이 있습니다. 이러한 서비스는 자체 사용자 공간, 커널 및 물리적 호스트와 직접 통신하여 해당 서비스를 네트워크에 직접 제공합니다. 다음 이미지는 가상화되지 않은 환경을 나타냅니다.



가상화 환경

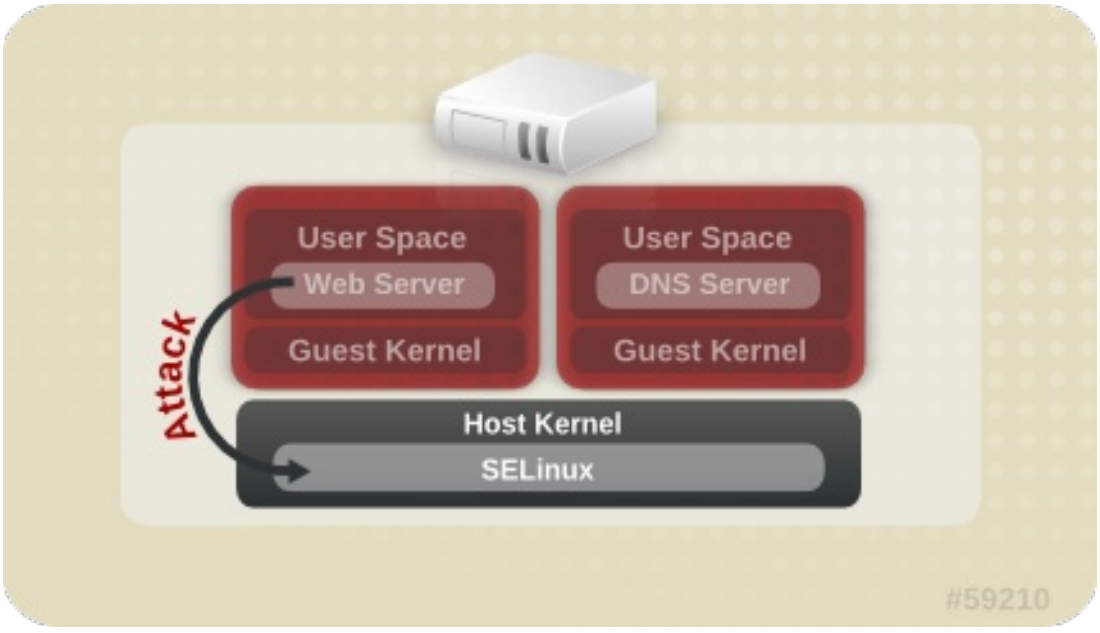
가상화 환경에서는 여러 운영 체제를 단일 호스트 커널과 물리적 호스트 내에 수용(게스트)할 수 있습니다. 다음 이미지는 가상화된 환경을 나타냅니다.



8.1. 보안 및 가상화

서비스가 가상화되지 않은 경우 시스템은 물리적으로 분리됩니다. 모든 위협은 네트워크 공격을 제외하고 일반적으로 영향을 받는 시스템에 포함됩니다. 서비스를 가상화 환경에 그룹화하면 시스템에서 추가 취약점이 발생합니다. 게스트 인스턴스에서 악용할 수 있는 하이퍼바이저에 보안 취약점이 있는 경우 이 게스트는 호스트를 공격할 뿐만 아니라 해당 호스트에서 실행 중인 다른 게스트도 수행할 수 있습니다. 이는 이론적이지 않습니다. 이미 하이퍼바이저에 공격이 있습니다. 이러한 공격은 게스트 인스턴스를 초과하여 다른 게스트를 공격에 노출할 수 있습니다.

***sVirt**는 게스트를 격리하고 악용되는 경우 추가 공격을 제한하기 위한 노력입니다. 이 내용은 가상 머신을 중단하고 다른 호스트 인스턴스로 확장할 수 없는 다음 이미지에서 보여줍니다.*



***SELinux**는 **MAC(Mandatory Access Control)** 구현에 가상화 인스턴스를 위한 플러그형 보안 프레임워크를 도입합니다. **sVirt** 프레임워크를 사용하면 게스트와 해당 리소스의 레이블을 고유하게 지정할 수*

있습니다. 레이블을 지정하면 서로 다른 게스트 간의 액세스를 거부할 수 있는 규칙을 적용할 수 있습니다.

8.2. SVIRT 레이블링

sVirt는 SELinux 보호의 다른 서비스와 마찬가지로 프로세스 기반 메커니즘과 제한 사항을 사용하여 게스트 인스턴스에 대한 추가 보안 계층을 제공합니다. 일반적으로 sVirt가 백그라운드에서 작동되고 있음을 알 수 없습니다. 이 섹션에서는 sVirt의 레이블 지정 기능에 대해 설명합니다.

다음 출력에 표시된 대로 sVirt를 사용할 때 각 VM(가상 시스템) 프로세스에 레이블이 지정되며 동적으로 생성된 수준으로 실행됩니다. 각 프로세스는 다른 레벨의 다른 VM과 격리됩니다.

```
~]# ps -eZ | grep qemu

system_u:system_r:svirt_t:s0:c87,c520 27950 ?  00:00:17 qemu-kvm
system_u:system_r:svirt_t:s0:c639,c757 27989 ? 00:00:06 qemu-system-x86
```

실제 디스크 이미지는 다음 출력에 표시된 대로 프로세스와 일치하도록 자동으로 레이블이 지정됩니다.

```
~]# ls -lZ /var/lib/libvirt/images/*

system_u:object_r:svirt_image_t:s0:c87,c520   image1
```

다음 표에서는 sVirt를 사용할 때 할당할 수 있는 다양한 라벨을 간략하게 설명합니다.

표 8.1. sVirt 레이블

유형	SELinux 문맥	Description
가상 머신 프로세스	system_u:system_r:svirt_t:MCS1	MCS1은 임의로 선택한 MCS 필드입니다. 현재 약 50만 개의 라벨이 지원됩니다.
가상 머신 이미지	system_u:object_r:svirt_image_t:MCS1	동일한 MCS 필드가 있는 *svirt_t* 레이블이 지정된 프로세스만 해당 이미지 파일 및 장치를 읽고 쓸 수 있습니다.
가상 머신 공유 읽기/쓰기 콘텐츠	system_u:object_r:svirt_image_t:s0	*svirt_t* 라는 모든 프로세스는 svirt_image_t:s0 파일 및 장치에 쓸 수 있습니다.

유형	SELinux 문맥	Description
가상 머신 이미지	system_u:object_r:virt_content_t:s0	이미지가 종료될 때 사용되는 시스템 기본 레이블입니다. 이 레이블이 있는 파일/장치를 읽을 수 있는 *svirt_t* 가상 프로세스가 없습니다.

***sVirt*를 사용할 때 정적 레이블을 수행할 수도 있습니다. 정적 레이블을 사용하면 관리자가 가상 시스템의 *MCS/MLS* 필드를 포함한 특정 레이블을 선택할 수 있습니다. 정적으로 레이블이 지정된 가상 시스템을 실행하는 관리자는 이미지 파일에서 올바른 레이블을 설정해야 합니다. 가상 시스템은 항상 해당 레이블로 시작되고 *sVirt* 시스템은 정적으로 레이블이 지정된 가상 시스템의 콘텐츠 레이블을 변경하지 않습니다. 이렇게 하면 *sVirt* 구성 요소가 *MLS* 환경에서 실행될 수 있습니다. 요구 사항에 따라 시스템에서 다양한 민감도 수준으로 여러 가상 시스템을 실행할 수도 있습니다.**

9장. 보안 LINUX 컨테이너

LXC(LinuxContainers)는 시스템에서 동일한 서비스의 여러 복사본을 동시에 실행할 수 있는 낮은 수준의 가상화 기능입니다. 전체 가상화에 비해 컨테이너는 전체 새 시스템을 부팅할 필요가 없으며 메모리를 적게 사용할 수 있으며 기본 운영 체제를 읽기 전용으로 사용할 수 있습니다. 예를 들어, **LXC**를 사용하면 시스템 데이터를 공유하는 동시에 여러 웹 서버를 동시에 실행할 수 있으며, 심지어 **root** 사용자로 실행할 수도 있습니다. 그러나 컨테이너 내에서 권한 있는 프로세스를 실행하면 다른 컨테이너에서 실행되는 컨테이너 또는 프로세스 외부에서 실행되는 다른 프로세스에 영향을 미칠 수 있습니다. 보안 **Linux** 컨테이너는 **SELinux** 컨텍스트를 사용하므로 해당 컨테이너에서 실행되는 프로세스가 서로 또는 호스트와 상호 작용하지 못하게 합니다.

Docker 애플리케이션은 **Red Hat Enterprise Linux**에서 **Linux** 컨테이너를 관리하는 주요 유틸리티입니다. 또는 **libvirt** 패키지에서 제공하는 **virsh** 명령줄 유틸리티를 사용할 수도 있습니다.

Linux 컨테이너에 대한 자세한 내용은 컨테이너 *시작하기*를 참조하십시오.

10장. SELINUX SYSTEMD 액세스 제어

Red Hat Enterprise Linux 7에서 시스템 서비스는 systemd 데몬에 의해 제어됩니다. 이전 Red Hat Enterprise Linux 릴리스에서 데몬은 다음 두 가지 방법으로 시작할 수 있습니다.

- 부팅 시 System V init 데몬이 init .rc 스크립트를 시작한 다음 이 스크립트는 필요한 데몬을 시작했습니다. 예를 들어 부팅 시 시작된 Apache 서버에는 다음 SELinux 레이블이 있습니다.

```
system_u:system_r:httpd_t:s0
```

- 관리자는 init.rc 스크립트를 수동으로 시작하여 데몬이 실행됩니다. 예를 들어, service httpd restart 명령이 Apache 서버에서 호출되면 생성되는 SELinux 레이블은 다음과 같습니다.

```
unconfined_u:system_r:httpd_t:s0
```

수동으로 시작하면 프로세스가 SELinux 레이블을 시작한 사용자 부분을 채택하여 위의 두 시나리오에서 레이블을 지정하지 않았습니다. systemd 데몬에서는 전환이 매우 다릅니다. systemd 는 init_t 유형을 사용하여 시스템에서 데몬 시작 및 중지를 위해 모든 호출을 처리하므로 데몬을 수동으로 다시 시작할 때 레이블의 사용자 부분을 재정의할 수 있습니다. 결과적으로 위의 두 시나리오의 레이블은 예상대로 system_u:system_r:httpd_t:s0 이며 SELinux 정책은 어떤 도메인을 제어할 수 있는지 제어하도록 개선할 수 있습니다.

10.1. 서비스의 SELINUX 액세스 권한

이전 버전의 Red Hat Enterprise Linux에서는 관리자가 System V Init 스크립트 레이블을 기반으로 서비스를 시작하거나 중지할 수 있는 사용자 또는 애플리케이션을 제어할 수 있었습니다. 이제 systemd 가 모든 서비스를 시작 및 중지하고, 사용자와 프로세스는 systemctl 유틸리티를 사용하여 systemd 와 통신합니다. systemd 데몬은 SELinux 정책을 참조하고 호출 프로세스의 레이블과 호출자가 관리하려고 하는 유닛 파일의 레이블을 확인한 다음, 호출자가 액세스할 수 있는지 여부를 SELinux에 요청합니다. 이 접근 방식을 통해 시스템 서비스의 시작 및 중지를 비롯한 중요한 시스템 기능에 대한 액세스 제어를 강화합니다.

예를 들어 관리자는 NetworkManager가 systemctl 을 실행하여 D-Bus 메시지를 systemd 로 보내도록 허용해야 했습니다. 이 메시지는 NetworkManager가 요청한 서비스 시작 또는 중지를 시작하거나 중지합니다. 실제로 NetworkManager는 systemctl 에서 수행할 수 있는 모든 작업을 수행할 수 있었습니다. 또한 특정 서비스를 시작하거나 중지할 수 있도록 제한된 관리자를 설정하는 것도 불가능했습니다.

이러한 문제를 해결하기 위해 systemd 는 SELinux 액세스 관리자로도 작동합니다. systemctl 을 실행하는 프로세스 또는 D-Bus 메시지를 systemd 로 보낸 프로세스의 레이블을 검색할 수 있습니다. 그런 다음 데몬은 프로세스가 구성하려는 유닛 파일의 레이블을 조회합니다. 마지막으로 SELinux 정책에서 프로세스 레이블과 유닛 파일 레이블 간의 특정 액세스를 허용하는 경우 systemd 는 커널에서 정보를 검색

*할 수 있습니다. 즉, 특정 서비스의 **systemd** 와 상호 작용해야 하는 손상된 애플리케이션은 이제 **SELinux**에서 제한할 수 있습니다. 정책 작성자는 이러한 세분화된 제어를 사용하여 관리자를 제한할 수도 있습니다. 정책 변경에는 다음과 같은 권한이 있는 **service** 라는 새 클래스가 포함됩니다.*

```
class service
{
        start
        stop
        status
        reload
        kill
        load
        enable
        disable
}
```

*예를 들어 정책 작성자는 도메인에서 서비스의 상태를 가져오거나 서비스를 시작 및 중지할 수 있지만 서비스를 활성화하거나 비활성화할 수 없습니다. **SELinux** 및 **systemd** 의 액세스 제어 작업이 일부 경우에 일치하지 않습니다. 매핑은 **SELinux** 액세스 검사를 사용하여 **systemd** 메서드 호출을 정렬하도록 정의되었습니다. 표 10.1. "SELinux 액세스 검사에서 systemd 장치 파일 메서드 호출 매핑" 은(는) 장치 파일에서 액세스 검사를 매핑하고 표 10.2. "SELinux 액세스 검사에서 systemd 일반 시스템 호출 매핑" 는 일반적으로 시스템에 대한 액세스 검사를 다룹니다. 두 테이블에서 일치하는 항목이 없으면 정의되지 않은 시스템 확인이 호출됩니다.*

표 10.1. SELinux 액세스 검사에서 systemd 장치 파일 메서드 호출 매핑

systemd 장치 파일 방법	SELinux 액세스 확인
DisableUnitFiles	비활성화
EnableUnitFiles	활성화
GetUnit	상태
GetUnitByPID	상태
GetUnitFileState	상태
강제 종료	stop
KillUnit	stop
LinkUnitFiles	활성화
ListUnits	상태
LoadUnit	상태

systemd 장치 파일 방법	SELinux 액세스 확인
MaskUnitFiles	비활성화
PresetUnitFiles	활성화
ReenableUnitFiles	활성화
다시 실행	시작
새로 고침	새로 고침
ReloadOrRestart	시작
ReloadOrRestartUnit	시작
ReloadOrTryRestart	시작
ReloadOrTryRestartUnit	시작
ReloadUnit	새로 고침
ResetFailed	stop
ResetFailedUnit	stop
재시작	시작
RestartUnit	시작
시작	시작
StartUnit	시작
StartUnitReplace	시작
중지	stop
StopUnit	stop
TryRestart	시작
TryRestartUnit	시작
UnmaskUnitFiles	활성화

표 10.2. SELinux 액세스 검사에서 systemd 일반 시스템 호출 매핑

systemd 일반 시스템 호출	SELinux 액세스 확인
ClearJobs	reboot
FlushDevices	Halted
get	상태
GetAll	상태
GetJob	상태
GetSeat	상태
GetSession	상태
GetSessionByPID	상태
GetUser	상태
Halted	Halted
인트로스펙션	상태
kexec	reboot
KillSession	Halted
KillUser	Halted
ListJobs	상태
ListSeats	상태
ListSessions	상태
ListUsers	상태
LockSession	Halted
PowerOff	Halted
재부팅	reboot
SetUserLinger	Halted
TerminateSeat	Halted

systemd 일반 시스템 호출	SELinux 액세스 확인
TerminateSession	Halted
TerminateUser	Halted

예 10.1. 시스템 서비스에 대한 SELinux 정책

sesearch 유틸리티를 사용하면 시스템 서비스에 대한 정책 규칙을 나열할 수 있습니다. 예를 들어 sesearch -A -s NetworkManager_t -c service 명령을 호출하면 다음을 반환합니다.

```
allow NetworkManager_t dnsmasq_unit_file_t : service { start stop status reload kill load } ;
allow NetworkManager_t nscd_unit_file_t : service { start stop status reload kill load } ;
allow NetworkManager_t ntpd_unit_file_t : service { start stop status reload kill load } ;
allow NetworkManager_t pppd_unit_file_t : service { start stop status reload kill load } ;
allow NetworkManager_t polipo_unit_file_t : service { start stop status reload kill load } ;
```

10.2. SELINUX 및 JOURNALD

systemd 에서 the journald 데몬(systemd-journal이라고도 함)은 로깅 데이터를 수집하고 저장하는 시스템 서비스인 syslog 유틸리티의 대안입니다. 커널에서 수신되는 로깅 정보, 시스템 서비스의 표준 및 오류 출력에서 libc syslog() 함수를 사용하는 사용자 프로세스 또는 네이티브 API를 사용하는 로깅 정보를 기반으로 구조화 및 인덱싱된 저널을 생성하고 유지관리합니다. 각 로그 메시지의 여러 메타데이터 필드를 안전한 방식으로 암시적으로 수집합니다.

systemd-journal 서비스는 SELinux와 함께 사용하여 보안을 강화할 수 있습니다. SELinux는 설계한 작업만 수행할 수 있도록 허용하여 프로세스를 제어합니다. 정책 작성자의 보안 목표에 따라 때로는 더 적은 경우도 있습니다. 예를 들어 SELinux는 손상된 ntpd 프로세스가 네트워크 시간 처리 이외의 모든 작업을 수행하지 못하게 합니다. 그러나 ntpd 프로세스는 SELinux가 손상된 프로세스가 해당 메시지를 계속 보낼 수 있도록 syslog 메시지를 전송합니다. 손상된 ntpd 는 다른 데몬과 일치하도록 syslog 메시지를 포맷하고 관리자가 오작동할 수 있거나 syslog 파일을 읽어 전체 시스템을 손상시키는 유틸리티를 제공할 수 있었습니다.

systemd-journal 데몬은 모든 로그 메시지를 확인하고 특히 SELinux 레이블을 추가합니다. 그러면 로그 메시지의 불일치가 감지되고 발생하기 전에 이 유형의 공격을 방지할 수 있습니다. journalctl 유틸리티를 사용하여 systemd 저널 로그를 쿼리할 수 있습니다. 명령줄 인수를 지정하지 않으면 이 유틸리티를 실행하면 가장 오래된 항목부터 시작하여 저널의 전체 내용이 나열됩니다. 시스템 구성 요소에 대한 로그를 포함하여 시스템에서 생성된 모든 로그를 보려면 journalctl 을 root로 실행합니다. 루트가 아닌 사용자로 실행하는 경우 현재 로그인한 사용자와 관련된 로그로만 출력이 제한됩니다.

예 10.2. journalctl을 사용하여 로그 나열

journalctl 을 사용하여 특정 **SELinux** 레이블과 관련된 모든 로그를 나열할 수 있습니다. 예를 들어 다음 명령은 **system_u:system_r:policykit_t:s0** 라벨에 로깅된 모든 로그를 나열합니다.

```
~]# journalctl _SELINUX_CONTEXT=system_u:system_r:policykit_t:s0
Oct 21 10:22:42 localhost.localdomain polkitd[647]: Started polkitd version 0.112
Oct 21 10:22:44 localhost.localdomain polkitd[647]: Loading rules from directory /etc/polkit-1/rules.d
Oct 21 10:22:44 localhost.localdomain polkitd[647]: Loading rules from directory /usr/share/polkit-1/rules.d
Oct 21 10:22:44 localhost.localdomain polkitd[647]: Finished loading, compiling and executing 5 rules
Oct 21 10:22:44 localhost.localdomain polkitd[647]: Acquired the name org.freedesktop.PolicyKit1 on the system bus Oct 21 10:23:10 localhost polkitd[647]: Registered Authentication Agent for unix-session:c1 (system bus name :1.49, object path /org/freedesktop/PolicyKit1/AuthenticationAgent, locale en_US.UTF-8) (disconnected from bus)
Oct 21 10:23:35 localhost polkitd[647]: Unregistered Authentication Agent for unix-session:c1 (system bus name :1.80 [/usr/bin/gnome-shell --mode=classic], object path /org/freedesktop/PolicyKit1/AuthenticationAgent, locale en_US.utf8)
```

journalctl 에 대한 자세한 내용은 **journalctl(1)** 매뉴얼 페이지를 참조하십시오.

11장. 문제 해결

다음 장에서는 **SELinux**가 액세스를 거부할 때 발생하는 사항, 세 가지 문제의 원인, 올바른 레이블 지정에 대한 정보를 찾는 위치, **SELinux** 거부 분석, **audit2allow** 로 사용자 지정 정책 모듈 생성에 대해 설명합니다.

11.1. 액세스가 거부되면 어떤 문제가 발생합니까

액세스 허용 또는 허용하지 않기와 같은 **SELinux** 결정은 캐시됩니다. 이 캐시를 **AVC**(액세스 벡터 캐시)라고 합니다. 거부 메시지는 **SELinux**가 액세스를 거부하면 로깅됩니다. 이러한 거부는 **"AVC** 거부"라고도 하며 실행 중인 데몬에 따라 다른 위치에 기록됩니다.

데몬: **auditd**

로그 위치: **/var/log/audit/audit.log**

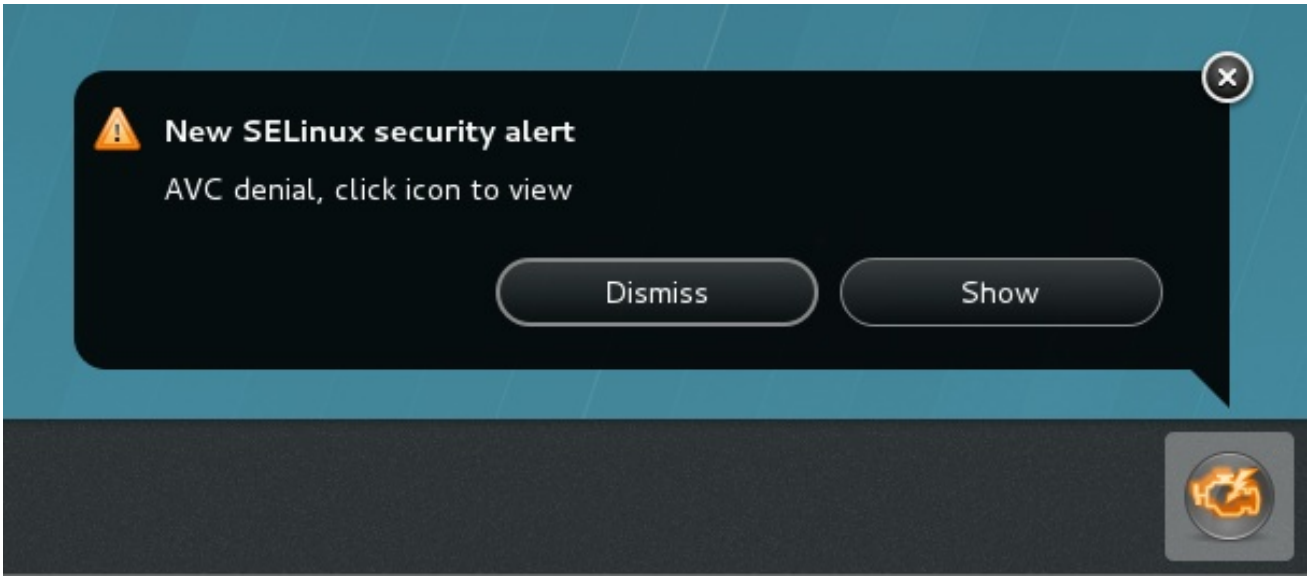
데몬: **auditd off; rsyslogd on**

로그 위치: **/var/log/messages**

데몬: **setroubleshootd, rsyslogd, auditd**

로그 위치: **/var/log/audit/audit.log.** 읽기 쉬운 거부 메시지도 **/var/log/messages**에 전송됩니다.

X Window 시스템을 실행하는 경우 **setroubleshoot** 및 **setroubleshoot -server** 패키지가 설치되어 **setroubleshootd** 및 **auditd** 데몬이 실행 중인 경우 **SELinux**에서 액세스를 거부하면 경고가 표시됩니다.



Show (표시)를 클릭하면 **SELinux**가 액세스를 거부한 이유와 액세스를 허용하는 가능한 솔루션이 표시됩니다. **X Window** 시스템을 실행하지 않는 경우 **SELinux**에서 액세스를 거부하면 명확하지 않습니다. 예를 들어, 웹사이트를 검색하는 사용자에게 다음과 유사한 오류가 발생할 수 있습니다.

```
Forbidden

You don't have permission to access file name on this server
```

이러한 경우 **DAC** 규칙(표준 **Linux** 권한)에서 액세스를 허용하는 경우 **/var/log/messages** 및 **/var/log/audit/audit.log** 에서 **"SELinux**가 차단" 및 "거부됨" 오류가 각각 있는지 확인합니다. **root** 사용자로 다음 명령을 실행하여 수행할 수 있습니다.

```
~]# grep "SELinux is preventing" /var/log/messages
```

```
~]# grep "denied" /var/log/audit/audit.log
```

11.2. 상위 세 가지 문제 원인

다음 섹션에서는 문제 레이블 지정, 서비스용 부울 및 포트 구성, **SELinux** 규칙의 세 가지 원인에 대해 설명합니다.

11.2.1. 문제 레이블 지정

SELinux를 실행하는 시스템에서는 모든 프로세스와 파일에 보안 관련 정보가 포함된 레이블이 지정됩니다. 이 정보를 **SELinux** 컨텍스트라고 합니다. 이러한 레이블이 잘못되면 액세스가 거부될 수 있습니다. 애플리케이션이 잘못 레이블된 경우 잘못된 레이블이 프로세스에 할당될 수 있습니다. 이로 인해 **SELinux**가 액세스를 거부할 수 있으며 프로세스가 레이블이 잘못 지정된 파일이 생성될 수 있습니다.

레이블 지정 문제의 일반적인 원인은 비표준 디렉터리가 서비스에 사용되는 경우입니다. 예를 들어 웹 사이트에 **/var/www/html/** 를 사용하는 대신 관리자가 **/srv/myweb/** 을 사용하려고 합니다. **Red Hat Enterprise Linux**에서 **/srv** 디렉터리에 **var_t** 유형으로 레이블이 지정됩니다. **/srv** 에 생성된 파일과 디렉터리는 이 유형을 상속합니다. 또한 최상위 디렉토리(예: **/myserver**)에서 새로 생성된 오브젝트는 **default_t** 유형으로 레이블이 지정될 수 있습니다. **SELinux**는 **Apache HTTP Server(httpd)**가 이러한 두 가지 유형 모두에 액세스하지 못하도록 합니다. 액세스를 허용하려면 **SELinux**에서 **/srv/myweb/** 의 파일에 **httpd** 에서 액세스할 수 있음을 알아야 합니다.

```
~]# semanage fcontext -a -t httpd_sys_content_t "/srv/myweb(/.*)?"
```

이 **semanage** 명령은 **/srv/myweb/** 디렉토리(및 그 아래의 모든 파일 및 디렉터리)에 대한 컨텍스트를 **SELinux** 파일 컨텍스트 구성에 추가합니다.[8]. **semanage** 유틸리티는 컨텍스트를 변경하지 않습니다. **root**로 **restorecon** 유틸리티를 실행하여 변경 사항을 적용합니다.

```
~]# restorecon -R -v /srv/myweb
```

파일 컨텍스트 구성에 컨텍스트를 추가하는 방법에 대한 자세한 내용은 4.7.2절. “영구적인 변경 사항:

semanage fcontext" 을 참조하십시오.

11.2.1.1. 올바른 문맥이란 무엇입니까?

matchpathcon 유틸리티는 파일 경로의 컨텍스트를 확인하고 해당 경로의 기본 레이블과 비교합니다. 다음 예제에서는 파일을 잘못 레이블로 지정한 디렉터리에서 matchpathcon 을 사용하는 방법을 보여줍니다.

```
~]$ matchpathcon -V /var/www/html/*
/var/www/html/index.html has context unconfined_u:object_r:user_home_t:s0, should be
system_u:object_r:httpd_sys_content_t:s0
/var/www/html/page1.html has context unconfined_u:object_r:user_home_t:s0, should be
system_u:object_r:httpd_sys_content_t:s0
```

이 예제에서 index.html 및 page1.html 파일에는 user_home_t 유형으로 레이블이 지정됩니다. 이 유형은 사용자 홈 디렉토리의 파일에 사용됩니다. mv 명령을 사용하여 홈 디렉토리의 파일을 이동하면 user_home_t 유형으로 파일에 레이블이 지정될 수 있습니다. 이 유형은 홈 디렉토리 외부에 없어야 합니다. restorecon 유틸리티를 사용하여 이러한 파일을 올바른 유형으로 복원합니다.

```
~]# restorecon -v /var/www/html/index.html
restorecon reset /var/www/html/index.html context unconfined_u:object_r:user_home_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
```

디렉터리 아래의 모든 파일의 컨텍스트를 복원하려면 -R 옵션을 사용합니다.

```
~]# restorecon -R -v /var/www/html/
restorecon reset /var/www/html/page1.html context unconfined_u:object_r:samba_share_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
restorecon reset /var/www/html/index.html context unconfined_u:object_r:samba_share_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
```

matchpathcon 의 자세한 예는 4.10.3절. "기본 SELinux 컨텍스트 확인" 을 참조하십시오.

11.2.2. 제한된 서비스 실행 방법은 무엇입니까?

서비스는 다양한 방법으로 실행할 수 있습니다. 이를 위해 서비스를 실행하는 방법을 지정해야 합니다. 이 작업은 SELinux 정책 작성에 대한 지식 없이 런타임 시 SELinux 정책 부분을 변경할 수 있는 부울을 통해 수행할 수 있습니다. 이렇게 하면 SELinux 정책을 다시 로드하거나 다시 컴파일하지 않고도 NFS 볼륨에 서비스 액세스 허용 등의 변경 사항을 허용합니다. 또한 기본이 아닌 포트 번호에서 서비스를 실행하려면 semanage 명령을 사용하여 정책 구성을 업데이트해야 합니다.

예를 들어 Apache HTTP 서버가 MariaDB와 통신할 수 있도록 하려면

httpd_can_network_connect_db 부울을 활성화합니다.

```
~]# setsebool -P httpd_can_network_connect_db on
```

특정 서비스에 대한 액세스가 거부된 경우 **getsebool** 및 **grep** 유틸리티를 사용하여 액세스를 허용하는 부울을 사용할 수 있는지 확인합니다. 예를 들어 **getsebool -a | grep ftp** 명령을 사용하여 **FTP** 관련 부울을 검색합니다.

```
~]$ getsebool -a | grep ftp
ftpd_anon_write --> off
ftpd_full_access --> off
ftpd_use_cifs --> off
ftpd_use_nfs --> off

ftpd_connect_db --> off
httpd_enable_ftp_server --> off
tftp_anon_write --> off
```

부울 목록 및 **on** 또는 **off** 여부에 대해 **getsebool -a** 명령을 실행합니다. 부울 목록의 경우 각 항목이 무엇인지, **on** 또는 **off** 상태인지에 대한 설명으로 **semanage boolean -l** 명령을 **root**로 실행합니다. 부울 나열 및 구성에 대한 자세한 내용은 *4.6절. "부울"* 을 참조하십시오.

포트 번호

정책 구성에 따라 서비스는 특정 포트 번호에서만 실행되도록 허용할 수 있습니다. 정책 변경 없이 서비스가 실행되는 포트를 변경하려고 하면 서비스가 시작되지 않을 수 있습니다. 예를 들어 **semanage port -l | grep http** 명령을 **root**로 실행하여 **http** 관련 포트를 나열합니다.

```
~]# semanage port -l | grep http
http_cache_port_t              tcp      3128, 8080, 8118
http_cache_port_t              udp      3130
http_port_t                    tcp      80, 443, 488, 8008, 8009, 8443
pegasus_http_port_t            tcp      5988
pegasus_https_port_t           tcp      5989
```

http_port_t 포트 유형은 **Apache HTTP Server**가 수신 대기할 수 있는 포트를 정의합니다. 이 경우 **TCP** 포트 **80, 443, 488, 8008, 8009, 8443**입니다. **httpd**가 포트 **9876(Listen 9 876)** 에서 수신 대기하지 만이를 반영하도록 정책이 업데이트되지 않도록 관리자가 **httpd.conf** 를 구성하는 경우 다음 명령이 실패합니다.

```
~]# systemctl start httpd.service
Job for httpd.service failed. See 'systemctl status httpd.service' and 'journalctl -xn' for details.
```

```
~]# systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
```

```
  Active: failed (Result: exit-code) since Thu 2013-08-15 09:57:05 CEST; 59s ago
  Process: 16874 ExecStop=/usr/sbin/httpd $OPTIONS -k graceful-stop (code=exited,
status=0/SUCCESS)
  Process: 16870 ExecStart=/usr/sbin/httpd $OPTIONS -DFOREGROUND (code=exited,
status=1/FAILURE)
```

다음과 유사한 **SELinux** 거부 메시지가 **/var/log/audit/audit.log**에 기록됩니다.

```
type=AVC msg=audit(1225948455.061:294): avc:  denied  { name_bind } for  pid=4997
comm="httpd" src=9876 scontext=unconfined_u:system_r:httpd_t:s0
tcontext=system_u:object_r:port_t:s0 tclass=tcp_socket
```

httpd 가 **http_port_t** 포트 유형에 대해 나열되지 않은 포트에서 수신 대기하도록 허용하려면 **semanage port** 명령을 입력하여 정책 구성에 포트를 추가합니다.[9]:

```
~]# semanage port -a -t http_port_t -p tcp 9876
```

a 옵션은 새 레코드를 추가하고, **-t** 옵션은 유형을 정의하고, **-p** 옵션은 프로토콜을 정의합니다. 마지막 인수는 추가할 포트 번호입니다.

11.2.3. 진화하는 규칙 및 깨진 애플리케이션

애플리케이션이 손상되어 **SELinux**에서 액세스를 거부할 수 있습니다. 또한 **SELinux** 규칙이 진화하고 있습니다. **SELinux**는 특정 방식으로 애플리케이션이 실행되고 있지 않을 수 있으므로 애플리케이션이 예상대로 작동하는 경우에도 액세스를 거부할 수 있습니다. 예를 들어, **PostgreSQL**의 새 버전이 릴리스되면 현재 정책이 이전에 표시되지 않은 작업을 수행하여 액세스가 허용되어야 하는 경우에도 액세스가 거부될 수 있습니다.

이러한 경우 액세스가 거부된 후 **audit2allow** 유틸리티를 사용하여 사용자 지정 정책 모듈을 생성하여 액세스를 허용합니다. **audit2allow** 사용에 대한 자세한 내용은 **11.3.8절. "액세스 허용: audit2allow"** 을 참조하십시오.

11.3. 문제 해결

다음 섹션에서는 문제를 해결하는 데 도움이 됩니다. **SELinux** 규칙보다 먼저 확인되는 **Linux** 권한을 확인하며, **SELinux**가 액세스를 거부하는 원인은 있지만 기록되는 거부는 발생하지 않습니다. 레이블 및 부울에 대한 정보가 포함된 서비스의 도움말 페이지, 전체 시스템이 아닌 한 프로세스를 허용하도록 허용 도메인, 전체 메시지를 검색 및 보는 방법, 거부 메시지 분석, **audit2allow** 로 사용자 지정 정책 모듈 생성.

11.3.1. Linux 권한

액세스가 거부되면 표준 **Linux** 권한을 확인합니다. 1장. 소개 에서 언급했듯이 대부분의 운영 체제는 **DAC**(임의적 액세스 제어) 시스템을 사용하여 액세스를 제어하므로 사용자가 소유한 파일의 권한을 제어할 수 있습니다. **SELinux** 정책 규칙은 **DAC** 규칙 후에 확인됩니다. **DAC** 규칙이 먼저 액세스를 거부하면 **SELinux** 정책 규칙이 사용되지 않습니다.

액세스가 거부되고 **SELinux** 거부가 기록되지 않은 경우 다음 명령을 사용하여 표준 **Linux** 권한을 확인합니다.

```
~]$ ls -l /var/www/html/index.html
-rw-r----- 1 root root 0 2009-05-07 11:06 index.html
```

이 예제에서 **index.html** 은 **root** 사용자 및 그룹이 소유합니다. **root** 사용자에게는 읽기 및 쓰기 권한(**-rw**)이 있으며 **root** 그룹의 멤버는 읽기 권한(**-r-**)을 갖습니다. 다른 모든 사용자에게는 액세스 권한이 없습니다(**---**). 기본적으로 이러한 권한으로 인해 **httpd** 에서 이 파일을 읽을 수 없습니다. 이 문제를 해결하려면 **chown** 명령을 사용하여 소유자와 그룹을 변경합니다. 이 명령은 **root**로 실행해야 합니다.

```
~]# chown apache:apache /var/www/html/index.html
```

이 경우 **httpd** 가 **Linux Apache** 사용자로 실행되는 기본 구성이 있다고 가정합니다. **httpd** 를 다른 사용자로 실행하는 경우 **apache:apache** 를 해당 사용자로 교체합니다.

Linux 권한 관리에 대한 정보는 **Fedora Documentation Project "Permissions"** 초안을 참조하십시오.

11.3.2. 음소거 거부의 원인

특정 상황에서 **SELinux**가 액세스를 거부하면 **AVC** 거부 메시지가 기록되지 않을 수 있습니다. 애플리케이션 및 시스템 라이브러리 기능은 작업을 수행하는 데 필요한 것보다 더 많은 액세스 권한을 조사하는 경우가 많습니다. 무해한 애플리케이션 탐색을 위해 **AVC** 거부로 감사 로그를 채우지 않고 최소 권한을 유지하기 위해 정책은 **dontaudit** 규칙을 사용하여 권한을 허용하지 않고 **AVC** 거부를 음소거할 수 있습니다. 이러한 규칙은 표준 정책에서 일반적입니다. **dontaudit**의 단점은 **SELinux**가 액세스를 거부하지만 거부 메시지가 기록되지 않으므로 문제 해결이 더 어렵습니다.

dontaudit 규칙을 일시적으로 비활성화하여 모든 거부를 로깅할 수 있도록 하려면 다음 명령을 **root**로 입력합니다.

```
~]# semodule -DB
```

D 옵션은 **dontaudit** 규칙을 비활성화합니다. **-B** 옵션은 정책을 다시 빌드합니다. **semodule -DB** 를 실행한 후 권한 문제가 발생한 애플리케이션을 실행해 보고 애플리케이션과 관련된 **SELinux** 거부가 기록

*되는지 확인하십시오. **dontaudit** 규칙에 의해 일부 거부를 무시하고 처리해야 하므로 허용되는 거부를 결정할 때 주의하십시오. 확실하지 않거나 지침을 검색할 때 **SELinux** 목록(예: **fedora-selinux-list**)의 다른 **SELinux** 사용자와 개발자에게 문의하십시오.*

*정책을 다시 빌드하고 **dontaudit** 규칙을 활성화하려면 **root**로 다음 명령을 입력합니다.*

```
~]# semodule -B
```

*이렇게 하면 정책이 원래 상태로 복원됩니다. **dontaudit** 규칙의 전체 목록을 보려면 **sesearch --dontaudit** 명령을 실행합니다. **-s** 도메인 옵션과 **grep** 명령을 사용하여 검색 범위를 좁힙니다. 예를 들어 다음과 같습니다.*

```
~]$ sesearch --dontaudit -s smbd_t | grep squid
dontaudit smbd_t squid_port_t : tcp_socket name_bind ;
dontaudit smbd_t squid_port_t : udp_socket name_bind ;
```

*분석 거부에 대한 자세한 내용은 **11.3.6절. “원시 감사 메시지”** 및 **11.3.7절. “sealert 메시지”** 을 참조하십시오.*

11.3.3. 서비스 수동 페이지

*서비스 도움말 페이지에는 특정 상황에 사용할 파일 유형과 같은 중요한 정보가 포함되어 있으며, 서비스에 있는 액세스를 변경하는 부울(예: **NFS** 볼륨에 **httpd** 액세스)이 포함되어 있습니다. 이 정보는 표준 도움말 페이지 또는 **sepolicy manpage** 유틸리티를 사용하여 모든 서비스 도메인에 대해 **SELinux** 정책에서 자동으로 생성할 수 있는 도움말 페이지에 있을 수 있습니다. 이러한 도움말 페이지의 이름은 **service-name_selinux** 형식으로 지정됩니다. 이러한 도움말 페이지는 **selinux-policy-doc** 패키지와 함께 제공됩니다.*

*예를 들어 **httpd_selinux(8)** 도움말 페이지에는 주어진 상황에 사용할 파일 유형과 부울을 사용하여 스크립트를 허용하고, 파일을 공유하고, 사용자 홈 디렉터리 내의 디렉터리에 액세스하는 등의 정보가 있습니다. 서비스를 위한 **SELinux** 정보가 포함된 기타 도움말 페이지는 다음과 같습니다.*

- ***Samba:** 예를 들어 **samba_selinux(8)** 매뉴얼 페이지는 **samba_enable_home_dirs** 부울을 활성화하면 **Samba**에서 사용자 홈 디렉터리를 공유할 수 있음을 설명합니다.*
- ***NFS:** **nfsd_selinux(8)** 도움말 페이지는 사용자가 가능한 한 방법으로 **nfsd** 프로세스를 보안할 수 있는 **SELinux nfsd** 정책을 설명합니다.*

*도움말 페이지의 정보는 **SELinux**가 액세스를 거부하지 않도록 올바른 파일 유형과 부울을 구성하는*

데 도움이 됩니다.

sepolicy manpage 에 대한 자세한 내용은 **5.4**절. "수동 페이지 생성: **sepolicy manpage**" 을 참조하십시오.

11.3.4. 허용 도메인

SELinux가 허용 모드로 실행 중이면 **SELinux**는 액세스를 거부하지 않지만 강제 모드에서 실행 중인 경우 거부된 작업에 대해 거부된 작업에 대해 거부됩니다. 이전에는 단일 도메인 허용(프로세스: 프로세스가 도메인에서 실행됨)을 수행할 수 없었습니다. 이로 인해 전체 시스템의 문제를 해결할 수 있게 되었습니다.

허용 도메인을 사용하면 관리자가 전체 시스템 허용을 수행하는 대신 단일 프로세스(도메인)가 허용 실행되도록 구성할 수 있습니다. **SELinux** 검사는 허용 도메인에 대해 계속 수행됩니다. 그러나 커널은 **SELinux**가 액세스를 거부한 상황에서 **AVC** 거부를 허용합니다.

허용 도메인에는 다음과 같은 용도가 있습니다.

- 단일 프로세스(도메인) 실행 허용(도메인)을 실행하여 전체 시스템을 허용하도록 위험을 초래하지 않고 문제를 해결하는 데 사용할 수 있습니다.
- 이를 통해 관리자는 새 애플리케이션에 대한 정책을 만들 수 있습니다. 이전에는 최소 정책을 생성한 다음 전체 시스템을 허용 모드로 전환하여 애플리케이션을 실행할 수 있지만 **SELinux** 거부는 여전히 기록되었습니다. 그러면 **audit2allow** 를 사용하여 정책을 작성할 수 있습니다. 이로 인해 전체 시스템이 위험해질 수 있습니다. 허용 도메인을 사용하면 새 정책의 도메인만 전체 시스템을 위험에 빠뜨리지 않고 허용으로 표시할 수 있습니다.

11.3.4.1. 도메인 허용 만들기

도메인 허용을 만들려면 **semanage permissive -a domain** 명령을 실행합니다. 여기서 **domain** 은 허용할 도메인입니다. 예를 들어 **httpd_t** 도메인(**Apache HTTP Server**가 실행되는 도메인)을 허용하려면 **root**로 다음 명령을 입력합니다.

```
~]# semanage permissive -a httpd_t
```

허용된 도메인 목록을 보려면 **semodule -l | grep permissive** 명령을 **root**로 실행합니다. 예를 들어 다음과 같습니다.

```
~]# semodule -l | grep permissive
permissive_httpd_t    (null)
permissivedomains     (null)
```

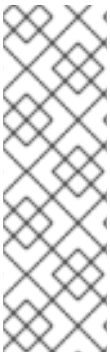
더 이상 도메인이 허용되지 않으려면 **semanage permissive -d domain** 명령을 **root**로 실행합니다. 예를 들어 다음과 같습니다.

```
~]# semanage permissive -d httpd_t
```

11.3.4.2. 허용 도메인 비활성화

permissivedomains.pp 모듈에는 시스템에 제공되는 모든 허용 도메인 선언이 포함되어 있습니다. 모든 허용 도메인을 비활성화하려면 **root**로 다음 명령을 입력합니다.

```
~]# semodule -d permissivedomains
```



참고

semodule -d 명령을 통해 정책 모듈을 비활성화하면 더 이상 **semodule -l** 명령의 출력에 표시되지 않습니다. 비활성화를 포함한 모든 정책 모듈을 보려면 **root**로 다음 명령을 입력합니다.

```
~]# semodule --list-modules=full
```

11.3.4.3. 허용 도메인에 대한 거부

SYSCALL 메시지는 허용 도메인에 따라 다릅니다. 다음은 **Apache HTTP** 서버의 **AVC** 거부 (및 관련 시스템 호출)의 예입니다.

```
type=AVC msg=audit(1226882736.442:86): avc:  denied  { getattr } for  pid=2427 comm="httpd"
path="/var/www/html/file1" dev=dm-0 ino=284133 scontext=unconfined_u:system_r:httpd_t:s0
tcontext=unconfined_u:object_r:samba_share_t:s0 tclass=file

type=SYSCALL msg=audit(1226882736.442:86): arch=40000003 syscall=196 success=no exit=-13
a0=b9a1e198 a1=bfc2921c a2=54dff4 a3=2008171 items=0 ppid=2425 pid=2427 auid=502 uid=48
gid=48 euid=48 suid=48 fsuid=48 egid=48 sgid=48 fsgid=48 tty=(none) ses=4 comm="httpd"
exe="/usr/sbin/httpd" subj=unconfined_u:system_r:httpd_t:s0 key=(null)
```

기본적으로 **httpd_t** 도메인은 허용되지 않으므로 작업이 거부되고 **SYSCALL** 메시지에는 **success=no** 가 포함됩니다. 다음은 동일한 상황에 대한 **AVC** 거부의 예입니다. 단, **semanage permissive -a httpd_t** 명령이 **httpd_t** 도메인을 허용하도록 실행되었습니다.

```
type=AVC msg=audit(1226882925.714:136): avc:  denied  { read } for  pid=2512 comm="httpd"
name="file1" dev=dm-0 ino=284133 scontext=unconfined_u:system_r:httpd_t:s0
tcontext=unconfined_u:object_r:samba_share_t:s0 tclass=file

type=SYSCALL msg=audit(1226882925.714:136): arch=40000003 syscall=5 success=yes exit=11
a0=b962a1e8 a1=8000 a2=0 a3=8000 items=0 ppid=2511 pid=2512 auid=502 uid=48 gid=48
euid=48 suid=48 fsuid=48 egid=48 sgid=48 fsgid=48 tty=(none) ses=4 comm="httpd"
exe="/usr/sbin/httpd" subj=unconfined_u:system_r:httpd_t:s0 key=(null)
```

이 경우 **AVC** 거부가 기록되었지만 **SYSCALL** 메시지의 **success=yes** 에 표시된 대로 액세스가 거부되지 않았습니다.

허용 도메인에 대한 자세한 내용은 **Dan Walsh**의 **"허용 도메인"** 블로그 항목을 참조하십시오.

11.3.5. 거부 검색 및 보기

이 섹션에서는 **setroubleshoot,setroubleshoot-server,dbus** 및 **audit** 패키지가 설치되어 있고 **auditd,rsyslogd, setroubleshootd** 데몬이 실행 중이라고 가정합니다. 이러한 데몬을 시작하는 방법에 대한 자세한 내용은 **4.2절. "사용된 로그 파일은 무엇입니까?"** 을 참조하십시오. **ausearch,aureport** 및 **sealert** 와 같은 **SELinux AVC** 메시지를 검색하고 보는 데 여러 개의 유틸리티를 사용할 수 있습니다.

ausearch

audit 패키지는 다양한 검색 기준에 따라 이벤트에 대한 감사 데몬 로그를 쿼리할 수 있는 **ausearch** 유틸리티를 제공합니다.[10] **ausearch** 유틸리티는 **/var/log/audit/audit.log** 에 액세스하므로 **root** 사용자로 실행해야 합니다.

검색 대상: 모든 거부

명령: **ausearch -m avc,user_avc,selinux_err,user_selinux_err**

검색 대상: 이에 대한 거부는 오늘

명령: **ausearch -m avc -ts** 오늘

검색 대상: 지난 **10**분 후 거부

명령: **ausearch -m avc -ts recent**

특정 서비스를 위한 **SELinux AVC** 메시지를 검색하려면 **-c** 통신 이름 옵션을 사용합니다. 여기서 통신은 실행 파일의 이름(예: **Apache HTTP Server**의 경우 **httpd, Samba**의 **smbd**)을 사용합니다.

```
~]# ausearch -m avc -c httpd
```

```
~]# ausearch -m avc -c smbd
```

각 **ausearch** 명령을 사용하면 읽기 쉽도록 **--interpret (-i)** 옵션 또는 스크립트 처리를 위해 **--raw (-r)** 옵션을 사용하는 것이 좋습니다. 추가 **ausearch** 옵션은 **ausearch(8)** 도움말 페이지를 참조하십시오.

aureport

audit 패키지는 감사 시스템 로그에 대한 요약 보고서를 생성하는 **aureport** 유틸리티를 제공합니다.[11] **aureport** 유틸리티는 **/var/log/audit/audit.log** 에 액세스하므로 **root** 사용자로 실행해야 합니다. **SELinux** 거부 메시지 목록 및 각 메시지 목록을 보려면 **aureport -a** 명령을 실행합니다. 다음은 두 개의 거부를 포함하는 출력 예입니다.

```
~]# aureport -a

AVC Report
============================================================
# date time comm subj syscall class permission obj event
============================================================
1. 05/01/2009 21:41:39 httpd unconfined_u:system_r:httpd_t:s0 195 file getattr
system_u:object_r:samba_share_t:s0 denied 2
2. 05/03/2009 22:00:25 vsftpd unconfined_u:system_r:ftpd_t:s0 5 file read
unconfined_u:object_r:cifs_t:s0 denied 4
```

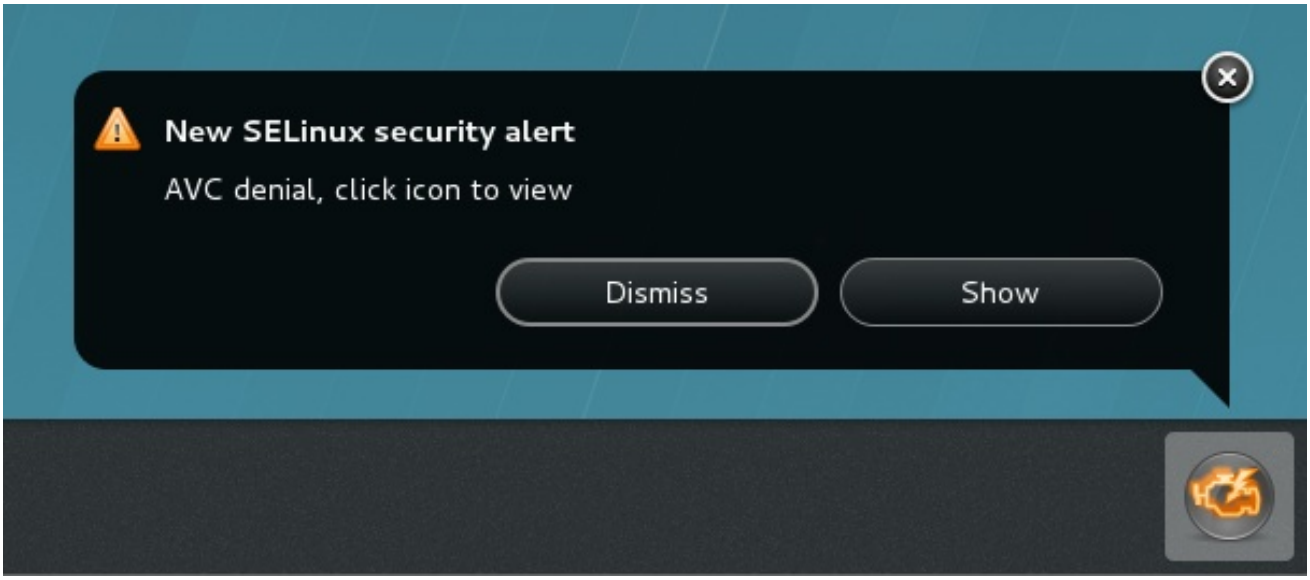
sealert

setroubleshoot-server 패키지는 **setroubleshoot-server** 에서 변환하는 거부 메시지를 읽는 **sealert** 유틸리티를 제공합니다.[12] 거부는 **/var/log/messages** 에 표시된 대로 할당된 **ID**입니다. 다음은 메시지에서 거부된 예입니다.

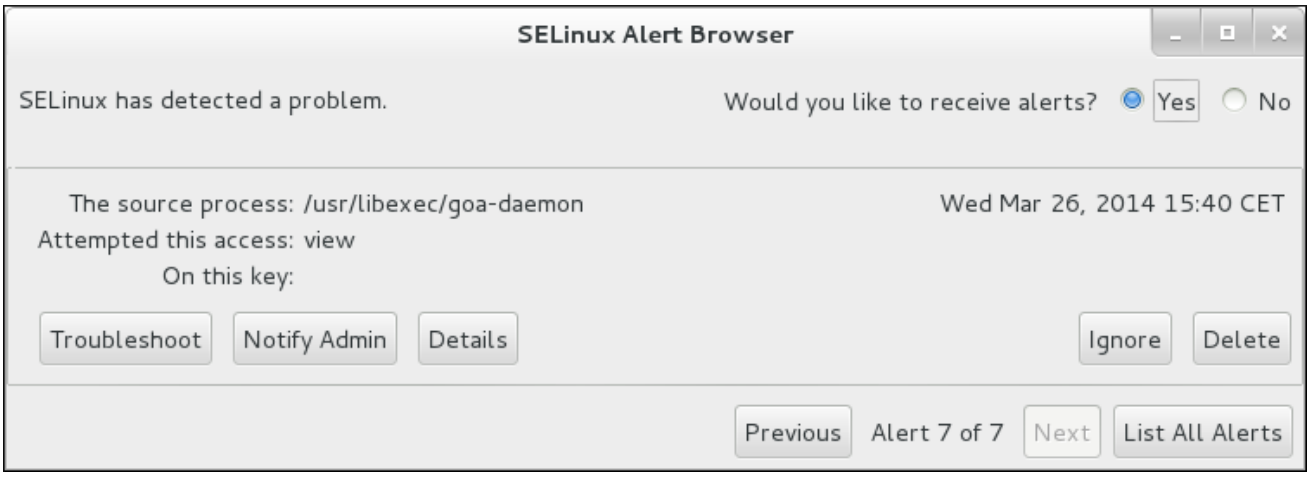
```
setroubleshoot: SELinux is preventing /usr/sbin/httpd from name_bind access on the tcp_socket. For
complete SELinux messages. run sealert -l 8c123656-5dda-4e5d-8791-9e3bd03786b7
```

이 예에서 거부 **ID**는 **8c123656-5dda-4e5d-8791-9e3bd03786b7** 입니다. **l** 옵션은 **ID**를 인수로 사용합니다. **sealert -l 8c123656-5dda-4e5d-8791-9e3bd03786b7** 명령을 실행하면 **SELinux**가 액세스를 거부한 이유와 액세스를 허용하는 가능한 솔루션을 자세히 분석할 수 있습니다.

X Window 시스템을 실행하는 경우 **setroubleshoot** 및 **setroubleshoot -server** 패키지가 설치되어 **setroubleshootd,dbus** 및 **auditd** 데몬이 실행 중인 경우 **SELinux**에서 액세스를 거부하면 경고가 표시됩니다.



Show (표시)를 클릭하면 sealert GUI가 시작되어 문제를 해결할 수 있습니다.



*또는 sealert -b 명령을 실행하여 sealert GUI를 시작합니다. 모든 거부 메시지의 상세한 분석을 보려면 sealert -l * 명령을 실행합니다.*

11.3.6. 원시 감사 메시지

원시 감사 메시지는 /var/log/audit/audit.log 에 기록됩니다. 다음은 Apache HTTP Server(httpd_t 도메인에서 실행)에서 /var/www/html/file1 파일(samba _share_t 유형으로 레이블이 지정됨)에 액세스하려고 할 때 발생한 AVC 거부 메시지(및 관련 시스템 호출) 예입니다.

```
type=AVC msg=audit(1226874073.147:96): avc:  denied  { getattr } for  pid=2465 comm="httpd"
path="/var/www/html/file1" dev=dm-0 ino=284133 scontext=unconfined_u:system_r:httpd_t:s0
tcontext=unconfined_u:object_r:samba_share_t:s0 tclass=file

type=SYSCALL msg=audit(1226874073.147:96): arch=40000003 syscall=196 success=no exit=-13
```

> *a0=b98df198 a1=bfec85dc a2=54dff4 a3=2008171 items=0 ppid=2463 pid=2465 auid=502 uid=48 gid=48 euid=48 suid=48 fsuid=48 egid=48 sgid=48 fsgid=48 tty=(none) ses=6 comm="httpd" exe="/usr/sbin/httpd" subj=unconfined_u:system_r:httpd_t:s0 key=(null)*

{ getattr }

*중괄호의 항목은 거부된 권한을 나타냅니다. **getattr** 항목은 소스 프로세스가 타겟 파일의 상태 정보를 읽으려는 것을 나타냅니다. 이 작업은 파일을 읽기 전에 수행됩니다. 이 작업은 레이블이 잘못 되어 액세스 중인 파일로 인해 거부됩니다. 일반적으로 표시되는 권한에는 **getattr,read, write** 가 포함됩니다.*

comm="httpd"

*프로세스를 시작한 실행 파일입니다. 실행 파일의 전체 경로는 시스템 호출(SYSCALL)메시지의 **exe=** 섹션에 있습니다. 이 경우 **exe="/usr/sbin/httpd"** 입니다.*

path="/var/www/html/file1"

프로세스가 액세스하려고 한 개체(대상)의 경로입니다.

scontext="unconfined_u:system_r:httpd_t:s0"

*거부된 작업을 시도한 프로세스의 **SELinux** 컨텍스트입니다. 이 경우 **httpd_t** 도메인에서 실행 중인 **Apache HTTP** 서버의 **SELinux** 컨텍스트입니다.*

tcontext="unconfined_u:object_r:samba_share_t:s0"

*프로세스가 액세스를 시도한 개체(대상)의 **SELinux** 컨텍스트입니다. 이 경우 **file1** 의 **SELinux** 컨텍스트입니다. **samba_share_t** 유형은 **httpd_t** 도메인에서 실행되는 프로세스에 액세스할 수 없습니다.*

*특정 상황에서 **tcontext** 는 프로세스가 사용자 **ID**와 같이 실행 중인 프로세스의 특성을 변경하는 시스템 서비스를 실행하려고 하는 경우와 같이 **scontext** 와 일치할 수 있습니다. 또한 프로세스가 일반 제한(예: 메모리)에서 허용하는 것보다 더 많은 리소스(예: 메모리)를 사용하려고 할 때 **tcontext** 가 **scontext** 와 일치할 수 있으므로 보안 검사를 통해 해당 프로세스가 해당 제한을 중단할 수 있는지 확인합니다.*

시스템 호출(SYSCALL)메시지에서 다음 두 가지 항목을 설정해야 합니다.

- *__success=no:(AVC)__이 강제 적용되었는지 여부를 나타냅니다. **success=no** 는 시스템 호출이 성공하지 않았음을 나타냅니다(**SELinux** 거부 액세스). **success=yes** 는 시스템 호출이 성공*

*했음을 나타냅니다. 이는 허용 도메인 또는 제한되지 않은 도메인(예: **unconfined _service_t** 및 **kernel_t**)에 대해 확인할 수 있습니다.*

- ***exe="/usr/sbin/httpd":** 프로세스를 시작한 실행 파일의 전체 경로(이 경우 **exe="/usr/sbin/httpd ".***

*잘못된 파일 유형은 **SELinux**에서 액세스를 거부하는 일반적인 원인입니다. 문제 해결을 시작하려면 소스 컨텍스트**(scontext)**를 대상 컨텍스트**(tcontext)**와 비교합니다. 프로세스**(scontext)**가 이러한 객체**(tcontext)**에 액세스해야 합니까**?** 예를 들어**, Apache HTTP Server(httpd_t)**는 별도로 구성하지 않는 한 **httpd_sys_content_t,public_content_t** 등과 같은 **httpd_selinux(8)** 도움말 페이지에 지정된 유형에만 액세스해야 합니다.*

11.3.7. sealert 메시지

*거부는 **/var/log/messages** 에 표시된 대로 할당된 **ID**입니다. 다음은 **Apache HTTP Server(httpd_t** 도메인에서 실행)에서 **/var/www/html/file1** 파일**(samba _share_t** 유형으로 레이블이 지정됨)에 액세스하려고 할 때 발생한 **AVC** 거부**(** 메시지로 로깅)입니다.*

```
hostname setroubleshoot: SELinux is preventing httpd (httpd_t) "getattr" to /var/www/html/file1
(samba_share_t). For complete SELinux messages. run sealert -l 32eee32b-21ca-4846-a22f-
0ba050206786
```

*권장되는 대로 **sealert -l 32eee32b-21ca-4846-a22f-0ba050206786** 명령을 실행하여 전체 메시지를 확인합니다. 이 명령은 로컬 시스템에서만 작동하며 **sealert GUI**와 동일한 정보를 제공합니다.*

```
~]$ sealert -l 32eee32b-21ca-4846-a22f-0ba050206786
SELinux is preventing httpd from getattr access on the file /var/www/html/file1.

*****  Plugin restorecon (92.2 confidence) suggests   ************************

If you want to fix the label.
/var/www/html/file1 default label should be httpd_sys_content_t.
Then you can run restorecon.
Do
# /sbin/restorecon -v /var/www/html/file1

*****  Plugin public_content (7.83 confidence) suggests   *********************

If you want to treat file1 as public content
Then you need to change the label on file1 to public_content_t or public_content_rw_t.
Do
# semanage fcontext -a -t public_content_t '/var/www/html/file1'
# restorecon -v '/var/www/html/file1'

*****  Plugin catchall (1.41 confidence) suggests   **************************
```

```
If you believe that httpd should be allowed getattr access on the file1 file by default.
Then you should report this as a bug.
You can generate a local policy module to allow this access.
Do
allow this access for now by executing:
# ausearch -c 'httpd' --raw | audit2allow -M my-httpd
# semodule -i my-httpd.pp


Additional Information:
Source Context                system_u:system_r:httpd_t:s0
Target Context                unconfined_u:object_r:samba_share_t:s0
Target Objects                /var/www/html/file1 [ file ]
Source                        httpd
Source Path                   httpd
Port                          <Unknown>
Host                          hostname.redhat.com
Source RPM Packages
Target RPM Packages
Policy RPM                    selinux-policy-3.13.1-166.el7.noarch
Selinux Enabled               True
Policy Type                   targeted
Enforcing Mode                Enforcing
Host Name                     hostname.redhat.com
Platform                      Linux hostname.redhat.com
                              3.10.0-693.el7.x86_64 #1 SMP Thu Jul 6 19:56:57
                              EDT 2017 x86_64 x86_64
Alert Count                   2
First Seen                    2017-07-20 02:52:11 EDT
Last Seen                     2017-07-20 02:52:11 EDT
Local ID                      32eee32b-21ca-4846-a22f-0ba050206786

Raw Audit Messages
type=AVC msg=audit(1500533531.140:295): avc:  denied  { getattr } for  pid=24934 comm="httpd"
path="/var/www/html/file1" dev="vda1" ino=31457414 scontext=system_u:system_r:httpd_t:s0
tcontext=unconfined_u:object_r:samba_share_t:s0 tclass=file


Hash: httpd,httpd_t,samba_share_t,file,getattr
```

요약

*거부된 작업에 대한 간략한 요약. 이는 **/var/log/messages** 의 거부와 동일합니다. 이 예제에서 **httpd** 프로세스는 **samba_share_t** 유형으로 레이블이 지정된파일**(file1)**에 대한 액세스가거부되었습니다.*

자세한 설명

*보다 자세한 설명. 이 예제에서 **file1** 은 **samba_share_t** 유형으로 레이블이 지정됩니다. 이 유형은 **Samba**를 사용하여 내보내려는 파일 및 디렉터리에 사용됩니다. 설명은 이러한 액세스가 필요한 경우 **Apache HTTP Server** 및 **Samba**에서 액세스할 수 있는 유형으로 유형을 변경하는 것을 제안합니다.*

액세스 허용

*액세스를 허용하는 방법에 대한 제안 사항. 이는 파일의 레이블을 다시 지정하거나 부울을 활성화하거나 로컬 정책 모듈을 만드는 것일 수 있습니다. 이 경우 제안은 **Apache HTTP** 서버와 **Samba** 모두에서 액세스할 수 있는 유형을 사용하여 파일에 레이블을 지정하는 것입니다.*

명령 수정

*액세스를 허용하고 거부를 해결하기 위해 제안된 명령. 이 예제에서는 명령을 제공하여 **Apache HTTP** 서버 및 **Samba**에서 액세스할 수 있는 **public_content_t** 로 **file1** 유형을 변경합니다.*

추가 정보

*정책 패키지 이름 및 버전(**selinux-policy-3.13.1-166.el7.noarch**)과 같은 버그 보고서에 유용한 정보이지만 거부가 발생한 이유를 해결하는 데 도움이 되지 않을 수 있습니다.*

원시 감사 메시지

*거부와 연결된 **/var/log/audit/audit.log** 의 원시 감사 메시지입니다. **AVC** 거부의 각 항목에 대한 자세한 내용은 **11.3.6절. "원시 감사 메시지"** 을 참조하십시오.*

11.3.8. 액세스 허용: audit2allow

주의

***production**의 이 섹션에서는 예제를 사용하지 마십시오. 이는 **audit2allow** 유틸리티의 사용을 시연하는 데만 사용됩니다.*

***audit2allow** 유틸리티는 거부된 작업의 로그에서 정보를 수집한 다음 **SELinux** 정책 허용 규칙을 생성합니다.[13] **11.3.7절. "sealert 메시지"** 에 따라 거부 메시지를 분석한 후 레이블이 변경되거나 부울이 액세스를 허용하지 않으면 **audit2allow** 를 사용하여 로컬 정책 모듈을 만듭니다. **SELinux**에서 액세스를 거부하면 **audit2allow** 를 실행하면 이전에 거부된 액세스를 허용하는 유형 적용 규칙이 생성됩니다.*

***SELinux** 거부가 표시되는 경우 **audit2allow** 를 사용하여 첫 번째 옵션으로 로컬 정책 모듈을 생성해서는 안 됩니다. 레이블 지정 문제가 있는 경우 문제 해결을 시작해야 합니다. 두 번째 가장 자주 발생하는 사례는 프로세스 구성을 변경했으며 **SELinux**에 대해 알려주는 것을 잊어버리는 것입니다. 자세한 내용은 **SELinux 오류의 4가지 주요 원인 백서**를 참조하십시오.*

*다음 예제에서는 **audit2allow** 를 사용하여 정책 모듈을 생성하는 방법을 보여줍니다.*

1. *거부 메시지 및 관련 시스템 호출은 **/var/log/audit/audit.log** 파일에 기록됩니다.*

```
type=AVC msg=audit(1226270358.848:238): avc:  denied  { write } for  pid=13349
comm="certwatch" name="cache" dev=dm-0 ino=218171
scontext=system_u:system_r:certwatch_t:s0 tcontext=system_u:object_r:var_t:s0 tclass=dir

type=SYSCALL msg=audit(1226270358.848:238): arch=40000003 syscall=39 success=no
exit=-13 a0=39a2bf a1=3ff a2=3a0354 a3=94703c8 items=0 ppid=13344 pid=13349
auid=4294967295 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=(none)
ses=4294967295 comm="certwatch" exe="/usr/bin/certwatch"
subj=system_u:system_r:certwatch_t:s0 key=(null)
```

*이 예에서 **certwatch** 는 **var_t** 유형으로 레이블이 지정된 디렉터리에 대한 쓰기 액세스 권한을 거부했습니다. 거부 메시지를 **11.3.7절. "sealert 메시지"** 에 따라 분석합니다. 레이블이 변경되지 않거나 부울 액세스를 허용하지 않은 경우 **audit2allow** 를 사용하여 로컬 정책 모듈을 생성합니다.*

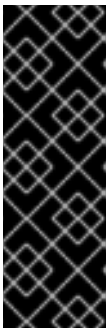
2. *다음 명령을 입력하여 액세스가 거부된 이유에 대한 사람이 읽을 수 있는 설명을 생성합니다. **audit2allow** 유틸리티는 **/var/log/audit/audit.log** 를 읽으므로 **root** 사용자로 실행해야 합니다.*

```
~]# audit2allow -w -a
type=AVC msg=audit(1226270358.848:238): avc:  denied  { write } for  pid=13349
comm="certwatch" name="cache" dev=dm-0 ino=218171
scontext=system_u:system_r:certwatch_t:s0 tcontext=system_u:object_r:var_t:s0 tclass=dir
 Was caused by:
  Missing type enforcement (TE) allow rule.

 You can use audit2allow to generate a loadable module to allow this access.
```

***a** 명령줄 옵션을 사용하면 모든 감사 로그를 읽습니다. **w** 옵션은 사람이 읽을 수 있는 설명을 생성합니다. 표시된 대로 유형 적용 규칙이 누락되어 액세스가 거부되었습니다.*

3. *거부된 액세스를 허용하는 유형 적용 규칙을 보려면 다음 명령을 입력합니다.*

```
~]# audit2allow -a


#============= certwatch_t ==============
allow certwatch_t var_t:dir write;
```



중요

유형 적용 규칙이 누락된 경우 일반적으로 **SELinux** 정책의 버그로 인해 발생하며 **Red Hat Bugzilla** 에서 보고해야 합니다. **Red Hat Enterprise Linux**의 경우 **Red Hat Enterprise Linux** 제품에 대한 버그를 생성하고 **selinux-policy** 구성 요소를 선택합니다. 이러한 버그 보고서에 **audit2allow -w -a** 및 **audit2allow -a** 명령의 출력을 포함합니다.

4. **audit2allow -a** 에서 표시하는 규칙을 사용하려면 **root**로 다음 명령을 입력하여 사용자 지정 모듈을 생성합니다. **M** 옵션은 현재 작업 디렉터리에 **-M** 으로 지정된 이름으로 **Type Enforcement** 파일(**.te**) 을 생성합니다.

```
~]# audit2allow -a -M mycertwatch
****************** IMPORTANT ***********************
To make this policy package active, execute:

semodule -i mycertwatch.pp
```

5. 또한 **audit2allow** 는 정책 패키지(**.pp**)에 **Type Enforcement** 규칙을 컴파일합니다.

```
~]# ls
mycertwatch.pp  mycertwatch.te
```

모듈을 설치하려면 **root**로 다음 명령을 입력합니다.

```
~]# semodule -i mycertwatch.pp
```

중요

audit2allow 로 생성된 모듈은 필요한 것보다 더 많은 액세스를 허용할 수 있습니다. **audit2allow** 를 사용하여 생성된 정책을 검토를 위해 업스트림 **SELinux** 목록에 게시하는 것이 좋습니다. 정책에 버그가 있다고 생각되면 **Red Hat Bugzilla** 에 버그를 생성하십시오.

여러 프로세스의 거부 메시지가 여러 개 있지만 단일 프로세스에 대한 사용자 지정 정책만 생성하려면 **grep** 유틸리티를 사용하여 **audit2allow** 에 대한 입력을 좁힙니다. 다음 예제에서는 **grep**을 사용하여 **audit 2allow** 를 통해 **certwatch** 와 관련된 거부 메시지만 보내는 방법을 보여줍니다.

```
~]# grep certwatch /var/log/audit/audit.log | audit2allow -R -M mycertwatch2
****************** IMPORTANT ***********************
To make this policy package active, execute:
```

```
semodule -i mycertwatch2.pp
```

---

[8]

*/etc/selinux/targeted/contexts/files/ 에 있는 파일은 파일 및 디렉터리에 대한 컨텍스트를 정의합니다. 이 디렉터리의 파일은 **restorecon** 및 **setfiles** 유틸리티에서 읽어 파일과 디렉터리를 기본 컨텍스트로 복원합니다.*

[9]

***semanage port -a** 명령은 **/etc/selinux/targeted/modules/active/ports.local** 파일에 항목을 추가합니다. 기본적으로 이 파일은 **root**만 볼 수 있습니다.*

[10]

***ausearch** 에 대한 자세한 내용은 **ausearch(8)** 도움말 페이지를 참조하십시오.*

[11]

***aureport** 에 대한 자세한 내용은 **aureport(8)** 도움말 페이지를 참조하십시오.*

[12]

***sealert** 에 대한 자세한 내용은 **sealert(8)** 도움말 페이지를 참조하십시오.*

[13]

***audit2allow** 에 대한 자세한 내용은 **audit2allow(1)** 도움말 페이지를 참조하십시오.*

12장. 추가 정보

12.1. 기여자

- Dominick Grift - 기술 편집자
- Murray McAllister - Red Hat 제품 보안
- James Morris - 기술 편집자
- Ericregister - 기술 편집자
- Scott Radvan - Red Hat 고객 콘텐츠 서비스
- Daniel Walsh - Red Hat 보안 엔지니어링

12.2. 기타 리소스

Fedora

- 메인 페이지: http://fedoraproject.org/wiki/SELinux.
- 문제 해결: http://fedoraproject.org/wiki/SELinux/Troubleshooting.
- Fedora SELinux FAQ: https://fedoraproject.org/wiki/SELinux_FAQ.

국가안보국(NSA)

NSA는 원래 SELinux 개발자였습니다. NSA의 NSA의 NSA 연구진들은 Linux 커널의 주요 하위 시스템에서 유연한 필수 액세스 제어를 설계 및 구현하고 Flask 아키텍처(보안 서버 및 액세스 벡터 캐시)에서

제공하는 새로운 운영 체제 구성 요소를 구현했습니다.

- 주요 SELinux 웹사이트: https://www.nsa.gov/what-we-do/research/selinux/.
- SELinux 메일링 목록: https://www.nsa.gov/what-we-do/research/selinux/mailing-list.shtml.
- SELinux 설명서: https://www.nsa.gov/what-we-do/research/selinux/documentation/index.shtml.
- SELinux 배경 정보: https://www.nsa.gov/what-we-do/research/selinux/related-work/.

Tresys 기술

Tresys Technology 는 다음 중 업스트림입니다.

- SELinux 사용자 공간 라이브러리 및 툴.
- SELinux 참조 정책.

SELinux GitHub 리포지토리

- SELinux 프로젝트: https://github.com/SELinuxProject
- Tresys 기술: https://github.com/TresysTechnology/

SELinux 프로젝트 Wiki

- 메인 페이지: http://selinuxproject.org/page/Main_Page.
- 설명서, 메일링 리스트, 웹사이트 및 툴에 대한 링크를 포함한 사용자 리소스: http://selinuxproject.org/page/User_Resources.

SELinux 노트북 - 기초 - 4번째 버전

http://freecomputerbooks.com/books/The_SELinux_Notebook-4th_Edition.pdf

디지털Ocean: CentOS 7의 SELinux 소개

https://www.digitalocean.com/community/tutorial_series/an-introduction-to-selinux-on-centos-7

IRC

Freenode 에서 :

- *#SELinux*
- *#fedora-selinux*

II 부. 제한된 서비스 관리

이 부분에서는 실제 작업에 중점을 두고 다양한 서비스를 설정하고 구성하는 방법에 대한 정보를 제공합니다. 각 서비스에 대해 사양을 사용하는 가장 일반적인 유형과 부울이 나열됩니다. 또한 이러한 서비스를 구성하는 실제 사례와 **SELinux**가 운영을 보완하는 방법에 대한 데모도 포함되어 있습니다.

SELinux가 강제 모드인 경우 **Red Hat Enterprise Linux**에서 사용되는 기본 정책은 대상 정책입니다. 대상이 되는 프로세스는 제한된 도메인과 제한되지 않은 도메인에서 대상으로 실행되지 않는 프로세스입니다. 대상 정책 및 제한 및 제한되지 않은 프로세스에 대한 자세한 내용은 *3장. 대상 지정 정책* 을 참조하십시오.

13장. APACHE HTTP 서버

Apache HTTP 서버는 현재 **HTTP** 표준과 함께 오픈 소스 **HTTP** 서버를 제공합니다.[14]

Red Hat Enterprise Linux에서 **httpd** 패키지는 **Apache HTTP** 서버를 제공합니다. 다음 명령을 입력하여 **httpd** 패키지가 설치되어 있는지 확인합니다.

```
~]$ rpm -q httpd
package httpd is not installed
```

설치되지 않고 **Apache HTTP** 서버를 사용하려면 **root** 사용자로 **yum** 유틸리티를 사용하여 설치합니다.

```
~]# yum install httpd
```

13.1. APACHE HTTP 서버 및 SELINUX

SELinux가 활성화되면 기본적으로 **Apache HTTP Server(httpd)**가 제한된 실행됩니다. 제한된 프로세스는 자체 도메인에서 실행되며 다른 제한된 프로세스와 분리됩니다. **SELinux** 정책 구성에 따라 공격자가 제한된 프로세스가 손상되면 공격자가 리소스에 대한 액세스와 가능한 손상을 제한합니다. 다음 예제에서는 자체 도메인에서 실행되는 **httpd** 프로세스를 보여줍니다. 이 예제에서는 **httpd,setroubleshoot,setroubleshoot-server** 및 **policycoreutils-python** 패키지가 설치되어 있다고 가정합니다.

1. **getenforce** 명령을 실행하여 **SELinux**가 강제 모드로 실행 중인지 확인합니다.

   ```
   ~]$ getenforce
   Enforcing
   ```

   명령은 **SELinux**가 강제 모드에서 실행 중일 때 **Enforcing** (강제)을 반환합니다.

2. **root**로 다음 명령을 입력하여 **httpd** 를 시작합니다.

   ```
   ~]# systemctl start httpd.service
   ```

서비스가 실행 중인지 확인합니다. 출력에는 아래 정보가 포함되어야 합니다(시간 스탬프만 다릅니다).

```
~]# systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: active (running) since Mon 2013-08-05 14:00:55 CEST; 8s ago
```

3. **httpd** 프로세스를 보려면 다음 명령을 실행합니다.

```
~]$ ps -eZ | grep httpd
system_u:system_r:httpd_t:s0    19780 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    19781 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    19782 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    19783 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    19784 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    19785 ?        00:00:00 httpd
```

httpd 프로세스와 연결된 **SELinux** 컨텍스트는 **system_u:system_r:httpd_t:s0**입니다. 컨텍스트의 마지막 부분인 **httpd_t** 는 유형입니다. 유형은 프로세스 및 파일의 유형에 대한 도메인을 정의합니다. 이 경우 **httpd** 프로세스는 **httpd _t** 도메인에서 실행됩니다.

SELinux 정책은 제한된 도메인(예: **httpd_t**)에서 실행되는 프로세스가 파일, 기타 프로세스 및 일반적으로 시스템과 상호 작용하는 방법을 정의합니다. **httpd** 액세스를 허용하려면 파일에 올바르게 레이블이 지정되어야 합니다. 예를 들어 **httpd** 는 **httpd_sys_content_t** 유형으로 레이블이 지정된 파일을 읽을 수 있지만 **Linux(DAC)** 권한이 쓰기 액세스를 허용하는 경우에도 쓸 수 없습니다. 스크립트 네트워크 액세스 허용, **NFS** 및 **CIFS** 볼륨에 대한 **httpd** 액세스 허용, **CGI(Common Gateway Interface)** 스크립트를 실행할 수 있는 **httpd** 등의 특정 동작을 허용하려면 부울을 활성화해야 합니다.

httpd 가 **TCP** 포트 **80, 443, 488, 8008, 8009** 또는 **8443** 이외의 포트에서 수신 대기하도록 **/etc/httpd/conf/httpd.conf** 파일을 구성하는 경우 **semanage port** 명령을 사용하여 **SELinux** 정책 구성에 새 포트 번호를 추가해야 합니다. 다음 예제에서는 **httpd** 의 **SELinux** 정책 구성에 아직 정의되어 있지 않은 포트에서 수신 대기하도록 **httpd** 를 구성하는 방법을 보여주므로 **httpd** 가 시작되지 않습니다. 이 예제에서는 **httpd** 가 정책에 아직 정의되지 않은 비표준 포트에서 수신 대기하도록 **SELinux** 시스템을 구성하는 방법도 보여줍니다. 이 예제에서는 **httpd** 패키지가 설치되어 있다고 가정합니다. 예제에서 **root** 사용자로 각 명령을 실행합니다.

1. 다음 명령을 입력하여 **httpd** 가 실행 중이 아닌지 확인합니다.

```
~]# systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
        Active: inactive (dead)
```

출력이 다르면 프로세스를 중지합니다.

```
~]# systemctl stop httpd.service
```

2. **semanage** 유틸리티를 사용하여 **SELinux**에서 **httpd** 가 수신 대기할 수 있는 포트를 확인합니다.

```
~]# semanage port -l | grep -w http_port_t
http_port_t                    tcp      80, 443, 488, 8008, 8009, 8443
```

3. **/etc/httpd/conf/httpd.conf** 파일을 **root**로 편집합니다. **httpd** 의 **SELinux** 정책 구성에 구성되지 않은 포트를 나열하도록 **Listen** 옵션을 구성합니다. 이 예제에서는 **httpd** 가 포트 **12345**에서 수신 대기하도록 구성되어 있습니다.

```
# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses (0.0.0.0)
#
#Listen 12.34.56.78:80
Listen 127.0.0.1:12345
```

4. **httpd** 를 시작하려면 다음 명령을 입력합니다.

```
~]# systemctl start httpd.service
Job for httpd.service failed. See 'systemctl status httpd.service' and 'journalctl -xn' for details.
```

다음과 유사한 **SELinux** 거부 메시지가 기록됩니다.

```
setroubleshoot: SELinux is preventing the httpd (httpd_t) from binding to port 12345. For
complete SELinux messages. run sealert -l f18bca99-db64-4c16-9719-1db89f0d8c77
```

5. **SELinux**가 이 예제에서 사용된 대로 **httpd** 가 포트 **12345**에서 수신 대기하도록 하려면 다음 명령이 필요합니다.

```
~]# semanage port -a -t http_port_t -p tcp 12345
```

6. **httpd** 를 다시 시작하고 새 포트에서 수신하도록 합니다.

```
~]# systemctl start httpd.service
```

7. 이제 **httpd** 가 비표준 포트(이 예에서는 **TCP 12345**)에서 수신 대기하도록 **SELinux**가 구성되었으므로 이 포트에서 **httpd** 가 성공적으로 시작됩니다.

8. **httpd** 가 수신 대기하고 **TCP** 포트 **12345**에서 통신하고 있음을 증명하려면 지정된 포트에 대한 **telnet** 연결을 열고 다음과 같이 **HTTP GET** 명령을 실행합니다.

```
~]# telnet localhost 12345
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
GET / HTTP/1.0

HTTP/1.1 200 OK
Date: Wed, 02 Dec 2009 14:36:34 GMT
Server: Apache/2.2.13 (Red Hat)
Accept-Ranges: bytes
Content-Length: 3985
Content-Type: text/html; charset=UTF-8
[...continues...]
```

13.2. 유형

고급 프로세스 격리를 제공하기 위해 **SELinux** 대상 정책에 사용되는 기본 권한 제어 방법은 **Type Enforcement**입니다. 모든 파일과 프로세스는 유형으로 레이블이 지정됩니다. **type**은 프로세스의 **SELinux** 도메인과 파일의 **SELinux** 유형을 정의합니다. **SELinux** 정책 규칙은 유형에 액세스하는 도메인이든 다른 도메인에 액세스하는 도메인이든 관계없이 유형이 서로 액세스하는 방법을 정의합니다. 액세스는 허용하는 특정 **SELinux** 정책 규칙이 있는 경우에만 허용됩니다.

다음 예제에서는 **/var/www/html/** 디렉터리에 새 파일을 만들고, 해당 상위 디렉터리(**/var/www/html/**)에서 **httpd_sys_content_t** 유형을 상속하는 파일을 보여줍니다.

1. **/var/www/html/** 의 **SELinux** 컨텍스트를 보려면 다음 명령을 입력합니다.

```
~]$ ls -dZ /var/www/html
drwxr-xr-x  root root system_u:object_r:httpd_sys_content_t:s0 /var/www/html
```

/var/www/html/ 이 **httpd_sys_content_t** 유형으로 레이블이 지정되었음을 표시합니다.

2. **touch** 유틸리티를 루트로 사용하여 새 파일을 생성합니다.

```
~]# touch /var/www/html/file1
```

3. 다음 명령을 입력하여 **SELinux** 컨텍스트를 확인합니다.

```
~]$ ls -Z /var/www/html/file1
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 /var/www/html/file1
```

ls -Z 명령은 **httpd_sys_content_t** 유형으로 레이블이 지정된 **file1** 을 표시합니다. **SELinux**를 사용하면 **httpd** 에서 이 유형으로 레이블이 지정된 파일을 읽을 수 있지만 **Linux** 권한이 쓰기 액세스를 허용하는 경우에도 쓸 수는 없습니다. **SELinux** 정책은 **httpd**가 실행되는 **httpd_t** 도메인에서 실행 중인 프로세스 유형을 읽고 쓸 수 있는 유형을 정의합니다. 따라서 프로세스가 다른 프로세스에서 사용할 파일에 액세스하는 것을 방지할 수 있습니다.

예를 들어 **httpd**는 **httpd _sys_content_t** 유형(**Apache HTTP Server**용으로 의도됨)으로 레이블이 지정된 파일에 액세스할 수 있지만 기본적으로 **samba_share_t** 유형(**Samba**의 경우 의도)으로 레이블이 지정된 파일에 액세스할 수 없습니다. 또한 사용자 홈 디렉터리의 파일에는 **user_home_t** 유형:으로 레이블이 지정됩니다. 기본적으로 **httpd** 는 사용자 홈 디렉터리의 파일을 읽거나 쓸 수 없습니다.

다음은 **httpd** 와 함께 사용되는 몇 가지 유형을 나열합니다. 다양한 유형을 사용하여 유연한 액세스를 구성할 수 있습니다.

httpd_sys_content_t

정적 웹 콘텐츠에 이 유형을 사용합니다(예: 정적 웹사이트에서 사용하는 **.html** 파일). 이 유형으로 레이블이 지정된 파일은 **httpd** 및 **httpd** 에서 실행되는 스크립트에 액세스할 수 있습니다 (읽기 전용). 기본적으로 이 유형으로 레이블이 지정된 파일과 디렉터리는 **httpd** 또는 기타 프로세스에 의해 작성하거나 수정할 수 없습니다. 기본적으로 에 생성되거나 **/var/www/html/** 디렉터리에 생성된 파일은 **httpd_sys_content_t** 유형으로 레이블이 지정됩니다.

httpd_sys_script_exec_t

httpd 를 실행할 스크립트에 이 유형을 사용합니다. 이 유형은 일반적으로 **/var/www/cgi-bin/** 디렉터리의 **CGI(Common Gateway Interface)** 스크립트에 사용됩니다. 기본적으로 **SELinux** 정책은 **httpd** 가 **CGI** 스크립트를 실행하지 못하게 합니다. 이를 허용하려면 **httpd_sys_script_exec_t** 유형으로 스크립트 레이블을 지정하고 **httpd_enable_cgi** 부울을 활성화합니다. **httpd_sys_script_exec_t**로 레이블이 지정된 스크립트는 **httpd_ sys_script_t** 도메인에서 **httpd _sys_script_t** 를 실행합니다. **httpd_sys_script_t** 도메인은 **postgresql_t, mysqld_t** 등의 다른 시스템 도메인에 액세스할 수 있습니다.

httpd_sys_rw_content_t

이 유형으로 레이블이 지정된 파일은 **httpd_sys_script_exec_t** 유형으로 레이블이 지정된 스크립트로 에 쓸 수 있지만 다른 유형으로 레이블이 지정된 스크립트에서는 수정할 수 없습니다.

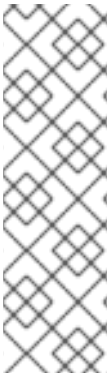
httpd_sys_rw_content_t 유형을 사용하여 **httpd _ sys_script_exec_t** 유형으로 레이블이 지정된 스크립트에서 읽고 쓸 파일에 레이블을 지정해야 합니다.

httpd_sys_ra_content_t

이 유형으로 레이블이 지정된 파일은 **httpd_sys_script_exec_t** 유형으로 레이블이 지정된 스크립트로 에 추가할 수 있지만 다른 유형으로 레이블이 지정된 스크립트에서는 수정할 수 없습니다. **httpd_sys_ra_content_t** 유형을 사용하여 **httpd _ sys_script_exec_t** 유형으로 레이블이 지정된 스크립트에서 읽고 에 추가할 파일에 레이블을 지정해야 합니다.

httpd_unconfined_script_exec_t

이 유형으로 레이블이 지정된 스크립트는 **SELinux** 보호 없이 실행됩니다. 다른 모든 옵션을 모두 고갈시킨 후 복잡한 스크립트에만 이 유형을 사용합니다. **httpd** 또는 전체 시스템에 대해 **SELinux** 보호를 비활성화하는 대신 이 유형을 사용하는 것이 좋습니다.



참고

httpd에 사용 가능한 유형을 추가하려면 다음 명령을 입력합니다.

```
~]$ grep httpd /etc/selinux/targeted/contexts/files/file_contexts
```

절차 **13.1. SELinux** 컨텍스트 변경

chcon 명령을 사용하여 파일 및 디렉터리의 유형을 변경할 수 있습니다. **chcon** 을 사용한 변경 사항은 파일 시스템의 레이블을 다시 지정하거나 **restorecon** 명령 후에도 유지되지 않습니다. **SELinux** 정책은 사용자가 지정된 파일에 대한 **SELinux** 컨텍스트를 수정할 수 있는지 여부를 제어합니다. 다음 예제에서는 **httpd**에서 사용할 새 디렉토리와 **index.html** 파일을 생성하고 **httpd** 가 액세스할 수 있도록 해당 파일과 디렉터리에 레이블을 지정하는 방법을 보여줍니다.

1. **mkdir** 유틸리티를 **root**로 사용하여 **httpd** 에서 사용할 파일을 저장할 최상위 디렉토리 구조를 만듭니다.

```
~]# mkdir -p /my/website
```

2. **file-context** 구성의 패턴과 일치하지 않는 파일과 디렉터리에 **default_t** 유형으로 레이블이 지정될 수 있습니다. 이 유형은 제한된 서비스에 액세스할 수 없습니다.

```
~]$ ls -dZ /my
drwxr-xr-x  root root unconfined_u:object_r:default_t:s0 /my
```

3. **root**로 다음 명령을 입력하여 **my**/ 디렉토리 및 하위 디렉터리의 유형을 **httpd** 에 액세스할 수 있는 유형으로 변경합니다. 이제 **/my/website**/ 에 생성된 파일은 **default _t** 유형이 아닌 **httpd_sys_content _t** 유형을 상속하므로 **httpd**에서 액세스할 수 있습니다.

```
~]# chcon -R -t httpd_sys_content_t /my/
~]# touch /my/website/index.html
~]# ls -Z /my/website/index.html
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 /my/website/index.html
```

chcon 에 대한 자세한 내용은 **4.7.1절. "임시 변경 사항: chcon"** 을 참조하십시오.

semanage fcontext 명령(정책**core utils -python** 패키지에서 제공)을 사용하여 재레이블 및 **restorecon** 명령 후에도 지속되는 레이블을 변경합니다. 이 명령은 파일 컨텍스트 구성에 변경 사항을 추가합니다. 그런 다음 **file-context** 구성을 읽는 **restorecon** 을 실행하여 레이블 변경 사항을 적용합니다. 다음 예제에서는 **httpd**에서 사용할 새 디렉토리와 **index.html** 파일을 생성하고 **httpd** 가 액세스할 수 있도록 해당 디렉터리와 파일의 레이블을 영구적으로 변경하는 방법을 보여줍니다.

1. **mkdir** 유틸리티를 **root**로 사용하여 **httpd** 에서 사용할 파일을 저장할 최상위 디렉토리 구조를 만듭니다.

```
~]# mkdir -p /my/website
```

2. **root**로 다음 명령을 입력하여 레이블 변경 사항을 파일 컨텍스트 구성에 추가합니다.

```
~]# semanage fcontext -a -t httpd_sys_content_t "/my(/.*)?"
```

"/my(/.*)?" 표현식은 레이블 변경 사항이 **my**/ 디렉토리 및 그 아래의 모든 파일과 디렉터리에 적용됨을 의미합니다.

3. **touch** 유틸리티를 **root**로 사용하여 새 파일을 생성합니다.

```
~]# touch /my/website/index.html
```

4. 다음 명령을 **root**로 입력하여 레이블 변경 사항을 적용합니다(**restorecon** 은 **file-context** 구성 읽기, **2** 단계에서 **semanage** 명령으로 수정됨):

```
~]# restorecon -R -v /my/
restorecon reset /my context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
restorecon reset /my/website context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
restorecon reset /my/website/index.html context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
```

semanage에 대한 자세한 내용은 4.7.2절. “영구적인 변경 사항: semanage fcontext” 을 참조하십시오.

13.3. 부울

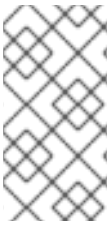
SELinux는 서비스를 실행하는 데 필요한 최소 액세스 수준을 기반으로 합니다. 서비스는 다양한 방법으로 실행할 수 있으므로 서비스 실행 방법을 지정해야 합니다. 이 작업은 **SELinux** 정책 작성에 대한 지식 없이 런타임 시 **SELinux** 정책 부분을 변경할 수 있는 부울을 사용하여 수행할 수 있습니다. 이렇게 하면 **SELinux** 정책을 다시 로드하거나 다시 컴파일하지 않고도 **NFS** 볼륨에 서비스 액세스 허용 등의 변경 사항을 허용합니다.

부울의 상태를 수정하려면 **setsebool** 명령을 사용합니다. 예를 들어 **httpd_anon_write** 부울을 활성화하려면 **root** 사용자로 다음 명령을 입력합니다.

```
~]# setsebool -P httpd_anon_write on
```

동일한 예제를 사용하여 부울을 비활성화하려면 다음과 같이 명령에서 를 **off** 로 변경하면 됩니다.

```
~]# setsebool -P httpd_anon_write off
```



참고

재부팅 시 **setsebool** 변경 사항이 지속되지 않도록 하려면 **-P** 옵션을 사용하지 마십시오.

다음은 **httpd** 가 실행 중인 방식을 지원하는 일반적인 부울에 대한 설명입니다.

httpd_anon_write

이 부울을 비활성화하면 **httpd** 에서 **public_content_rw_t** 유형으로 레이블이 지정된 파일에만 읽기 액세스 권한을 가질 수 있습니다. 이 부울을 활성화하면 **httpd** 가 공용 파일 전송 서비스의 파일이

*포함된 공용 디렉터리와 같은 **public_content_rw_t** 유형으로 레이블이 지정된 파일에 쓸 수 있습니다.*

httpd_mod_auth_ntlm_winbind

*이 부울을 활성화하면 **httpd** 에서 **mod_auth_ntlm_winbind** 모듈을 사용하여 **NTLM** 및 **Winbind** 인증 메커니즘에 액세스할 수 있습니다.*

httpd_mod_auth_pam

*이 부울을 활성화하면 **httpd** 에서 **mod_auth_pam** 모듈을 사용하여 **PAM** 인증 메커니즘에 액세스할 수 있습니다.*

httpd_sys_script_anon_write

*이 부울은 공개 파일 전송 서비스에 사용된 대로 **public_content_rw_t** 유형으로 레이블이 지정된 파일에 대한 쓰기 액세스가 허용되는지 여부를 정의합니다.*

httpd_builtin_scripting

*이 부울은 **httpd** 스크립팅에 대한 액세스를 정의합니다. 이 부울을 활성화하려면 **PHP** 콘텐츠에 필요합니다.*

httpd_can_network_connect

*이 부울을 비활성화하면 **HTTP** 스크립트 및 모듈이 네트워크 또는 원격 포트에 대한 연결을 시작하지 못하게 합니다. 이 액세스를 허용하려면 이 부울을 활성화합니다.*

httpd_can_network_connect_db

*이 부울을 비활성화하면 **HTTP** 스크립트 및 모듈이 데이터베이스 서버에 대한 연결을 시작하지 못하게 합니다. 이 액세스를 허용하려면 이 부울을 활성화합니다.*

httpd_can_network_relay

***httpd** 를 정방향 또는 역방향 프록시로 사용하는 경우 이 부울을 활성화합니다.*

httpd_can_sendmail

*이 부울을 비활성화하면 **HTTP** 모듈이 메일을 보내지 못하게 합니다. 이는 **httpd** 에서 취약점을 발견해야 하는 스팸 공격을 방지할 수 있습니다. **HTTP** 모듈이 메일을 보낼 수 있도록 이 부울을 활성화합니다.*

httpd_dbus_avahi

*비활성화된 경우 이 부울은**D-Bus** 를 통해 **avahi** 서비스에 대한 **httpd** 액세스를 거부합니다. 이 액세스를 허용하려면 이 부울을 활성화합니다.*

httpd_enable_cgi

*이 부울을 비활성화하면 **httpd** 가 **CGI** 스크립트를 실행하지 못하게 합니다. **httpd** 가 **CGI** 스크립트를 실행할 수 있도록 이 부울을 활성화합니다(**CGI** 스크립트에 **httpd_sys_script_exec_t** 유형으로 레이블이 지정되어야 함).*

httpd_enable_ftp_server

*이 부울을 활성화하면 **httpd** 가 **FTP** 포트에서 수신 대기하고 **FTP** 서버 역할을 합니다.*

httpd_enable_homedirs

*이 부울을 비활성화하면 **httpd** 가 사용자 홈 디렉터리에 액세스할 수 없습니다. 이 부울을 활성화하여 사용자 홈 디렉토리에 대한 **httpd** 액세스를 허용합니다(예: **/home/*/** 의 콘텐츠).*

httpd_execmem

*이 부울을 사용하면 **httpd** 에서 실행 가능하고 쓰기 가능한 메모리 주소가 필요한 프로그램을 실행할 수 있습니다. 이 부울을 활성화하면 버퍼 오버플로에 대한 보호가 감소하지만 특정 모듈 및 애플리케이션(예: **Java** 및 킥 애플리케이션)에 대한 보호 권한이 필요하므로 보안 관점에서는 사용하지 않는 것이 좋습니다.*

httpd_ssi_exec

*이 부울은 웹 페이지의 서버 측에 **SSI**(서버 측) 요소를 포함할 수 있는지 여부를 정의합니다.*

httpd_tty_comm

*이 부울은 **httpd** 가 제어 터미널에 액세스할 수 있는지 여부를 정의합니다. 일반적으로 이 액세스는 필요하지 않지만 **SSL** 인증서 파일을 구성하는 등의 경우 터미널 액세스가 암호 프롬프트를 표시하고 처리해야 합니다.*

httpd_unified

*이 부울을 활성화하면 **httpd_t** 가 모든 **httpd** 유형(**sys_content_t** 실행, 읽기 또는 쓰기)에 대한 전체 액세스를 허용합니다. 비활성화된 경우 읽기 전용, 쓰기 가능 또는 실행 가능한 웹 콘텐츠 간에 분리가 수행됩니다. 이 부울을 비활성화하면 추가적인 보안 수준이 보장되지만, 각각 보유해야 하는 파일*

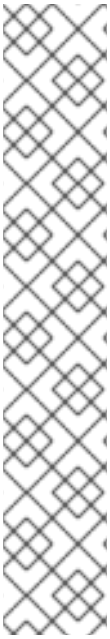
액세스에 따라 스크립트 및 기타 웹 콘텐츠에 개별적으로 레이블을 지정하는 관리 오버헤드가 추가됩니다.

httpd_use_cifs

Samba를 사용하여 마운트된 파일 시스템과 같은 **cifs_t** 유형으로 레이블이 지정된 **CIFS** 볼륨의 파일에 **httpd** 액세스를 허용하도록 이 부울을 활성화합니다.

httpd_use_nfs

NFS를 사용하여 마운트된 파일 시스템과 같이 **nfs_t** 유형으로 레이블이 지정된 **NFS** 볼륨의 파일에 **httpd** 액세스를 허용하도록 이 부울을 활성화합니다.



참고

SELinux 정책의 지속적인 개발로 인해 위의 목록에 항상 서비스와 관련된 부울이 모두 포함되지 않을 수 있습니다. 나열하려면 다음 명령을 입력합니다.

```
~]$ getsebool -a | grep service_name
```

특정 부울에 대한 설명을 보려면 다음 명령을 입력합니다.

```
~]$ sepolicy booleans -b boolean_name
```

이 명령이 작동하려면 **sepolicy** 유틸리티를 제공하는 추가 **policycoreutils-devel** 패키지가 필요합니다.

13.4. 설정 예

다음 예제에서는 **SELinux**가 **Apache HTTP** 서버를 보완하는 방법과 **Apache HTTP Server**의 전체 기능을 유지 관리하는 방법을 보여주는 실제 데모를 제공합니다.

13.4.1. 정적 사이트 실행

정적 웹 사이트를 만들려면 해당 웹 사이트의 **.html** 파일에 **httpd_sys_content_t** 유형의 레이블을 지정합니다. 기본적으로 **Apache HTTP** 서버는 **httpd_sys_content_t** 유형으로 레이블이 지정된 파일에 쓸 수 없습니다. 다음 예제에서는 읽기 전용 웹 사이트에 대한 파일을 저장할 새 디렉터리를 생성합니다.

1. **mkdir** 유틸리티를 **root**로 사용하여 최상위 디렉토리를 생성합니다.

```
~]# mkdir /mywebsite
```

2. ***root*로 */mywebsite/index.html* 파일을 만듭니다. */mywebsite/index.html*에 다음 콘텐츠를 복사하여 붙여넣습니다.**

```
<html>
<h2>index.html from /mywebsite/</h2>
</html>
```

3. ***Apache HTTP Server*가 */mywebsite/* 에 대한 액세스 권한뿐 아니라 그 아래의 파일 및 하위 디렉터리에만 액세스할 수 있도록 하려면 *httpd_sys_content_t* 유형의 디렉터리에 레이블을 지정합니다. *root*로 다음 명령을 입력하여 레이블 변경 사항을 파일 컨텍스트 구성에 추가합니다.**

```
~]# semanage fcontext -a -t httpd_sys_content_t "/mywebsite(/.*)?"
```

4. ***restorecon* 유틸리티를 루트로 사용하여 레이블을 변경합니다.**

```
~]# restorecon -R -v /mywebsite
restorecon reset /mywebsite context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
restorecon reset /mywebsite/index.html context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
```

5. ***이 예에서는 */etc/httpd/conf/httpd.conf* 파일을 *root*로 편집합니다. 기존 *DocumentRoot* 옵션을 주석 처리합니다. *DocumentRoot "/mywebsite"* 옵션을 추가합니다. 편집 후 이러한 옵션은 다음과 같이 표시됩니다.**

```
#DocumentRoot "/var/www/html"
DocumentRoot "/mywebsite"
```

6. ***root*로 다음 명령을 입력하여 *Apache HTTP Server*의 상태를 확인합니다. 서버가 중지되면 시작합니다.**

```
~]# systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: inactive (dead)
```

```
~]# systemctl start httpd.service
```

***서버가 실행 중인 경우 다음 명령을 *root*로 실행하여 서비스를 다시 시작합니다(*httpd .conf*

의 변경 사항도적용됨).

```
~]# systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: active (running) since Wed 2014-02-05 13:16:46 CET; 2s ago
```

```
~]# systemctl restart httpd.service
```

7. *웹 브라우저를 사용하여 **http://localhost/index.html** 로 이동합니다. 다음은 다음과 같습니다.*

```
index.html from /mywebsite/
```

13.4.2. NFS 및 CIFS 볼륨 공유

*기본적으로 클라이언트쪽의 **NFS** 마운트는 **NFS** 볼륨의 정책에서 정의한 기본 컨텍스트로 레이블이 지정됩니다. 일반적인 정책에서 이 기본 컨텍스트는 **nfs_t** 유형을 사용합니다. 또한 기본적으로 클라이언트측에 마운트된 **Samba** 공유는 정책에서 정의한 기본 컨텍스트로 레이블이 지정됩니다. 일반적인 정책에서 이 기본 컨텍스트는 **cifs_t** 유형을 사용합니다.*

*정책 구성에 따라 서비스는 **nfs_t** 또는 **cifs_t** 유형으로 레이블이 지정된 파일을 읽을 수 없습니다. 이렇게 하면 이러한 유형으로 레이블이 지정된 파일 시스템이 마운트되지 않고 다른 서비스에서 읽거나 내보내지 못하게 할 수 있습니다. 부울을 활성화하거나 비활성화하여 **nfs_t** 및 **cifs_t** 유형에 액세스할 수 있는 서비스를 제어할 수 있습니다.*

httpd_use_nfs** 부울을 활성화하여 **httpd** 가 **NFS** 볼륨에 액세스하고 공유할 수 있도록 허용합니다 **(nfs _t** 유형으로 레이블됨**).

```
~]# setsebool -P httpd_use_nfs on
```

httpd_use_cifs** 부울을 활성화하여 **httpd** 가 **CIFS** 볼륨에 액세스하고 공유할 수 있도록 허용합니다 **(cifs _t** 유형으로 레이블이 지정됨**).

```
~]# setsebool -P httpd_use_cifs on
```

참고

*재부팅 시 **setsebool** 변경 사항이 지속되지 않도록 하려면 **-P** 옵션을 사용하지 마십시오.*

13.4.3. 서비스 간 파일 공유

type Enforcement를 사용하면 프로세스가 다른 프로세스에서 사용할 파일에 액세스하는 것을 방지할 수 있습니다. 예를 들어 Samba는 기본적으로 Apache HTTP 서버에서 사용하기 위한 httpd_sys_content_t 유형으로 레이블이 지정된 파일을 읽을 수 없습니다. 필요한 파일에 public_content_ t 또는 public_content_rw_t 유형으로 레이블이 지정된 경우 Apache HTTP Server, FTP, rsync 및 Samba 간에 파일을 공유할 수 있습니다.

다음 예제에서는 디렉터리와 파일을 만들고 Apache HTTP 서버, FTP, rsync 및 Samba를 통해 해당 디렉토리와 파일을 공유(읽기 전용)할 수 있습니다.

1. mkdir 유틸리티를 root로 사용하여 여러 서비스 간에 파일을 공유할 새 최상위 디렉토리를 생성합니다.

```
~]# mkdir /shares
```

2. file-context 구성의 패턴과 일치하지 않는 파일과 디렉터리에 default_t 유형으로 레이블이 지정될 수 있습니다. 이 유형은 제한된 서비스에 액세스할 수 없습니다.

```
~]$ ls -dZ /shares
drwxr-xr-x  root root unconfined_u:object_r:default_t:s0 /shares
```

3. root로 /shares/index.html 파일을 만듭니다. 다음 콘텐츠를 복사하여 /shares/index.html 에 붙여넣습니다.

```
<html>
<body>
<p>Hello</p>
</body>
</html>
```

4. public_content_t 유형으로 /shares/ 레이블을 지정하면 Apache HTTP 서버, FTP, rsync 및 Samba의 읽기 전용 액세스만 허용됩니다. root로 다음 명령을 입력하여 레이블 변경 사항을 파일 컨텍스트 구성에 추가합니다.

```
~]# semanage fcontext -a -t public_content_t "/shares(/.*)?"
```

5. restorecon 유틸리티를 root로 사용하여 라벨 변경 사항을 적용합니다.

```
~]# restorecon -R -v /shares/
```

```
restorecon reset /shares context unconfined_u:object_r:default_t:s0-
>system_u:object_r:public_content_t:s0
restorecon reset /shares/index.html context unconfined_u:object_r:default_t:s0-
>system_u:object_r:public_content_t:s0
```

Samba를 통해 **/shares/** 를 공유하려면 다음을 수행합니다.

1. **samba, samba -common** 및 **samba-client** 패키지가 설치되어 있는지 확인합니다(버전 번호가 다를 수 있음).

```
~]$ rpm -q samba samba-common samba-client
samba-3.4.0-0.41.el6.3.i686
samba-common-3.4.0-0.41.el6.3.i686
samba-client-3.4.0-0.41.el6.3.i686
```

이러한 패키지가 설치되어 있지 않으면 **root**로 다음 명령을 실행하여 설치합니다.

```
~]# yum install package-name
```

2. **/etc/samba/smb.conf** 파일을 **root**로 편집합니다. 다음 항목을 이 파일의 맨 아래에 추가하여 **Samba**를 통해 **/shares/** 디렉토리를 공유합니다.

```
[shares]
comment = Documents for Apache HTTP Server, FTP, rsync, and Samba
path = /shares
public = yes
writable = no
```

3. **Samba** 파일 시스템을 마운트하려면 **Samba** 계정이 필요합니다. **root**로 다음 명령을 입력하여 **Samba** 계정을 만듭니다. 여기서 **username** 은 기존 **Linux** 사용자입니다. 예를 들어 **smbpasswd -a testuser** 는 **Linux testuser** 사용자에 대한 **Samba** 계정을 만듭니다.

```
~]# smbpasswd -a testuser
New SMB password: Enter a password
Retype new SMB password: Enter the same password again
Added user testuser.
```

위의 명령을 실행하면 시스템에 없는 계정의 사용자 이름을 지정하면 **'username'!** 오류에 대한 **Unix** 계정을 찾을 수 없습니다.

4. **Samba** 서비스를 시작합니다.

```
~]# systemctl start smb.service
```

5. 다음 명령을 입력하여 사용 가능한 공유를 나열합니다. 여기서 **username** 은 **3**단계에 추가된 **Samba** 계정입니다. 암호를 입력하라는 메시지가 표시되면 **3**단계에서 **Samba** 계정에 할당된 암호를 입력합니다(버전 번호는 다를 수 있음).

```
~]$ smbclient -U username -L localhost
Enter username's password:
Domain=[HOSTNAME] OS=[Unix] Server=[Samba 3.4.0-0.41.el6]

Sharename       Type      Comment
---------       ----      -------
shares          Disk      Documents for Apache HTTP Server, FTP, rsync, and Samba
IPC$            IPC       IPC Service (Samba Server Version 3.4.0-0.41.el6)
username        Disk      Home Directories
Domain=[HOSTNAME] OS=[Unix] Server=[Samba 3.4.0-0.41.el6]

Server               Comment
---------            -------

Workgroup            Master
---------            -------
```

6. **mkdir** 유틸리티를 사용하여 새 디렉토리를 만듭니다. 이 디렉토리는 **Samba** 공유를 마운트하는 데 사용됩니다.

```
~]# mkdir /test/
```

7. 다음 명령을 루트로 입력하여 **Samba** 공유를 **/test/** 에 마운트하고 **username** 을 **3**단계의 사용자 이름으로 바꿉니다.

```
~]# mount //localhost/shares /test/ -o user=username
```

3단계에 구성된 사용자 이름의 암호를 입력합니다.

8. **Samba**를 통해 공유되는 파일의 내용을 확인합니다.

```
~]$ cat /test/index.html
<html>
<body>
```

```
<p>Hello</p>
</body>
</html>
```

Apache HTTP 서버를 통해 /shares/ 를 공유하려면 다음을 수행합니다.

1. **httpd 패키지가 설치되었는지 확인합니다 (버전 번호는 다를 수 있음).**

```
~]$ rpm -q httpd
httpd-2.2.11-6.i386
```

이 패키지가 설치되지 않은 경우 yum 유틸리티를 root로 사용하여 설치합니다.

```
~]# yum install httpd
```

2. **/var/www/html/ 디렉터리로 변경합니다. root로 다음 명령을 입력하여 /shares/ 디렉터리에 링크(이름이 지정된 공유)를 생성합니다.**

```
html]# ln -s /shares/ shares
```

3. **Apache HTTP 서버를 시작합니다.**

```
~]# systemctl start httpd.service
```

4. **웹 브라우저를 사용하여 http://localhost/shares 로 이동합니다. /shares/index.html 파일이 표시됩니다.**

기본적으로 Apache HTTP 서버는 index.html 파일이 있는 경우 해당 파일을 읽습니다. /shares/ 에 index.html 이 없고 대신 file1, file 2 및 file 3 이 있는 경우 http://localhost/shares 에 액세스할 때 디렉토리 목록이 발생했습니다.

1. **index.html 파일을 제거합니다.**

```
~]# rm -i /shares/index.html
```

2. ***touch** 유틸리티를 루트로 사용하여 **/shares/** 에 세 개의 파일을 생성합니다.*

```
~]# touch /shares/file{1,2,3}
~]# ls -Z /shares/
-rw-r--r--  root root system_u:object_r:public_content_t:s0 file1
-rw-r--r--  root root unconfined_u:object_r:public_content_t:s0 file2
-rw-r--r--  root root unconfined_u:object_r:public_content_t:s0 file3
```

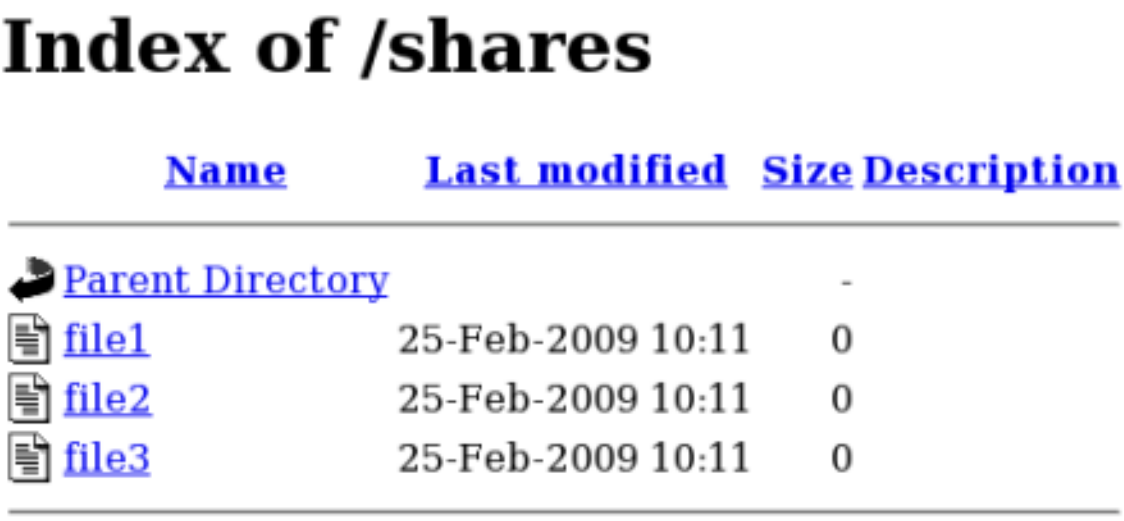
3. ***root**로 다음 명령을 입력하여 **Apache HTTP Server**의 상태를 확인합니다.*

```
~]# systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: inactive (dead)
```

서버가 중지되면 시작합니다.

```
~]# systemctl start httpd.service
```

4. *웹 브라우저를 사용하여 **http://localhost/shares** 로 이동합니다. 디렉터리 목록이 표시됩니다.*



13.4.4. 포트 번호 변경

*정책 구성에 따라 서비스는 특정 포트 번호에서만 실행되도록 허용할 수 있습니다. 정책 변경 없이 서비스가 실행되는 포트를 변경하려고 하면 서비스가 시작되지 않을 수 있습니다. **semanage** 유틸리티를 **root** 사용자로 사용하여 **SELinux**에서 **httpd** 가 수신 대기할 수 있는 포트를 나열합니다.*

```
~]# semanage port -l | grep -w http_port_t
http_port_t                    tcp      80, 443, 488, 8008, 8009, 8443
```

기본적으로 **SELinux**를 사용하면 **httpd** 가 **TCP** 포트 **80, 443, 488, 8008, 8009** 또는 **8443**에서 수신 대기할 수 있습니다. **httpd** 가 **http_port_t** 에 대해 나열되지 않은 포트에서 수신 대기하도록 **/etc/httpd/conf/httpd.conf** 가 구성된 경우**httpd** 가 시작되지 않습니다.

TCP 포트 **80, 443, 488, 8008, 8009** 또는 **8443** 이외의 포트에서 실행되도록 **httpd** 를 구성하려면 다음을 수행합니다.

1. **Listen** 옵션에 **httpd** 에 대한 **SELinux** 정책에 구성되지 않은 포트가 나열되도록 **/etc/httpd/conf/httpd.conf** 파일을 루트로 편집합니다. 다음 예제에서는 **10.0.0.1 IP** 주소 및 **TCP** 포트 **12345**에서 수신 대기하도록 **httpd** 를 구성합니다.

```
# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses (0.0.0.0)
#
#Listen 12.34.56.78:80
Listen 10.0.0.1:12345
```

2. 다음 명령을 **root** 사용자로 입력하여 포트를 **SELinux** 정책 구성에 추가합니다.

```
~]# semanage port -a -t http_port_t -p tcp 12345
```

3. 포트가 추가되었는지 확인합니다.

```
~]# semanage port -l | grep -w http_port_t
http_port_t                    tcp      12345, 80, 443, 488, 8008, 8009, 8443
```

포트 **12345**에서 더 이상 **httpd** 를 실행하지 않으면 **semanage** 유틸리티를 **root**로 사용하여 정책 구성에서 포트를 제거합니다.

```
~]# semanage port -d -t http_port_t -p tcp 12345
```

---

[14]
자세한 내용은 *시스템 관리자 가이드* 의 **Apache HTTP Sever** 섹션을 참조하십시오.

14장. SAMBA

Samba는**SMB**(서버 메시지 블록) 및**CIFS**(**Common Internet File System**) 프로토콜의 오픈 소스 구현으로, 다양한 운영 체제에서 클라이언트 간에 파일 및 인쇄 서비스를 제공합니다.[15]

Red Hat Enterprise Linux에서 **samba** 패키지는 **Samba** 서버를 제공합니다. 다음 명령을 입력하여 **samba** 패키지가 설치되어 있는지 확인합니다.

```
~]$ rpm -q samba
package samba is not installed
```

설치되어 있지 않고 **Samba**를 사용하려면 **yum** 유틸리티를 **root** 사용자로 사용하여 설치합니다.

```
~]# yum install samba
```

14.1. SAMBA 및 SELINUX

SELinux가 활성화되면 기본적으로 **Samba** 서버(**smbd**)가 제한된 실행됩니다. 제한된 서비스는 자체 도메인에서 실행되며 다른 제한된 서비스와 분리됩니다. 다음 예제에서는 자체 도메인에서 실행되는 **smbd** 프로세스를 보여줍니다. 이 예제에서는 **samba** 패키지가 설치되어 있다고 가정합니다.

1. **getenforce** 명령을 실행하여 **SELinux**가 강제 모드로 실행 중인지 확인합니다.

   ```
   ~]$ getenforce
   Enforcing
   ```

   명령은 **SELinux**가 강제 모드에서 실행 중일 때 **Enforcing** (강제)을 반환합니다.

2. **root**로 다음 명령을 입력하여 **smbd** 를 시작합니다.

   ```
   ~]# systemctl start smb.service
   ```

   서비스가 실행 중인지 확인합니다. 출력에는 아래 정보가 포함되어야 합니다(시간 스탬프만 다릅니다).

   ```
   ~]# systemctl status smb.service
   smb.service - Samba SMB Daemon
   ```

```
Loaded: loaded (/usr/lib/systemd/system/smb.service; disabled)
Active: active (running) since Mon 2013-08-05 12:17:26 CEST; 2h 22min ago
```

3. **smbd** 프로세스를 보려면 다음 명령을 실행합니다.

```
~]$ ps -eZ | grep smb
system_u:system_r:smbd_t:s0      9653 ?        00:00:00 smbd
system_u:system_r:smbd_t:s0      9654?         00:00:00 smbd
```

smbd 프로세스와 연결된 **SELinux** 컨텍스트는 **system_u:system_r:smbd_t:s0**입니다. 컨텍스트의 마지막 부분인 **smbd_t** 는 유형입니다. 유형은 프로세스 및 파일의 유형에 대한 도메인을 정의합니다. 이 경우 **smbd** 프로세스는 **smbd_t** 도메인에서 실행됩니다.

smbd 가 파일에 액세스하고 공유할 수 있도록 파일에 올바르게 레이블이 지정되어야 합니다. 예를 들어 **smbd** 는 **samba_share_t** 유형으로 레이블이 지정된 파일에 읽고 쓸 수 있지만 기본적으로 **Apache HTTP** 서버에서 사용하기 위한 **httpd_sys_content_t** 유형으로 레이블이 지정된 파일에 액세스할 수 없습니다. 홈 디렉터리 및 **NFS** 볼륨을 **Samba**를 통해 내보내고 **Samba**가 도메인 컨트롤러 역할을 허용하는 등의 특정 동작을 허용하려면 부울을 활성화해야 합니다.

14.2. 유형

고급 프로세스 격리를 제공하기 위해 **SELinux** 대상 정책에 사용되는 기본 권한 제어 방법은 **Type Enforcement**입니다. 모든 파일과 프로세스는 유형으로 레이블이 지정됩니다. **type**은 프로세스의 **SELinux** 도메인과 파일의 **SELinux** 유형을 정의합니다. **SELinux** 정책 규칙은 유형에 액세스하는 도메인이든 다른 도메인에 액세스하는 도메인이든 관계없이 유형이 서로 액세스하는 방법을 정의합니다. 액세스는 허용하는 특정 **SELinux** 정책 규칙이 있는 경우에만 허용됩니다.

samba_share_t 유형으로 파일에 레이블을 지정하여 **Samba**가 공유할 수 있도록 합니다. 생성한 파일만 레이블을 지정하고 **samba_share_t** 유형의 시스템 파일의 레이블을 다시 지정하지 마십시오. 이러한 파일과 디렉토리를 공유하기 위해 부울을 활성화할 수 있습니다. **SELinux**를 사용하면 **/etc/samba/smb.conf** 파일 및 **Linux** 권한이 적절하게 설정되는 한 **Samba**가 **samba_share_t** 유형으로 레이블이 지정된 파일에 쓸 수 있습니다.

samba_etc_t 유형은 **smb.conf** 와 같은 **/etc/samba/** 디렉터리의 특정 파일에서 사용됩니다. **samba_etc_t** 유형으로 파일에 수동으로 레이블을 지정하지 마십시오. 이 디렉터리의 파일에 올바르게 레이블이 지정되지 않은 경우 **restorecon -R -v /etc/samba** 명령을 루트 사용자로 입력하여 해당 파일을 기본 컨텍스트로 복원합니다. **/etc/samba/smb.conf** 에 **samba_etc_t** 유형으로 레이블이 지정되지 않은 경우 **Samba** 서비스를 시작하면 실패하고 **SELinux** 거부 메시지가 기록될 수 있습니다. 다음은 **/etc/samba/smb.conf** 가 **httpd_sys_content_t** 유형으로 레이블이 지정된 경우 거부 메시지의 예입니다.

```
setroubleshoot: SELinux is preventing smbd (smbd_t) "read" to ./smb.conf (httpd_sys_content_t). For
complete SELinux messages. run sealert -l deb33473-1069-482b-bb50-e4cd05ab18af
```

-

14.3. 부울

SELinux는 서비스를 실행하는 데 필요한 최소 액세스 수준을 기반으로 합니다. 서비스는 다양한 방법으로 실행할 수 있으므로 서비스 실행 방법을 지정해야 합니다. 다음 부울을 사용하여 **SELinux**를 설정합니다.

smbd_anon_write

이 부울을 활성화하면 **smbd** 가 특별한 액세스 제한이 없는 공통 파일을 위해 예약된 영역과 같은 공용 디렉토리에 쓸 수 있습니다.

samba_create_home_dirs

이 부울을 활성화하면 **Samba**가 새로운 홈 디렉토리를 독립적으로 만들 수 있습니다. 대체로 이 작업은 **PAM**과 같은 메커니즘에 의해 수행됩니다.

samba_domain_controller

이 부울을 사용하면 **Samba**가 도메인 컨트롤러 역할을 할 뿐만 아니라 **useradd,groupadd, passwd** 와 같은 관련 명령을 실행할 수 있는 권한을 부여할 수 있습니다.

samba_enable_home_dirs

이 부울을 활성화하면 **Samba**가 사용자의 홈 디렉토리를 공유할 수 있습니다.

samba_export_all_ro

파일 또는 디렉터리를 내보내 읽기 전용 권한을 허용합니다. 이렇게 하면 **samba_share_t** 유형으로 레이블이 지정되지 않은 파일과 디렉터리를 **Samba**를 통해 공유할 수 있습니다. **samba_export_all_ro** 부울이 활성화되어 있지만 **samba_export_all_rw** 부울이 비활성화되면 **/etc/samba/smb.conf** 에 쓰기 액세스가 구성되어 있어도 **Samba** 공유에 대한 쓰기 액세스 권한이 거부됩니다.

samba_export_all_rw

파일 또는 디렉터리를 내보내 읽기 및 쓰기 권한을 허용합니다. 이렇게 하면 **samba_share_t** 유형으로 레이블이 지정되지 않은 파일과 디렉터리를 **Samba**를 통해 내보낼 수 있습니다. 쓰기 액세스를 허용하도록 **/etc/samba/smb.conf** 및 **Linux** 권한의 권한을 구성해야 합니다.

samba_run_unconfined

이 부울을 활성화하면 **Samba**가 **/var/lib/samba/scripts/** 디렉토리에서 제한되지 않은 스크립트

를 실행할 수 있습니다.

samba_share_fusefs

***Samba**가 **fusefs** 파일 시스템을 공유하려면 이 부울을 활성화해야 합니다.*

samba_share_nfs

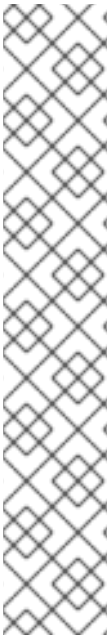
*이 부울을 비활성화하면 **smbd** 가 **Samba**를 통해 **NFS** 공유에 대한 전체 액세스 권한을 가질 수 없습니다. 이 부울을 활성화하면 **Samba**가 **NFS** 볼륨을 공유할 수 있습니다.*

use_samba_home_dirs

***Samba** 홈 디렉토리에 원격 서버를 사용하도록 이 부울을 활성화합니다.*

virt_use_samba

***CIFS** 파일에 대한 가상 머신 액세스를 허용합니다.*



참고

***SELinux** 정책의 지속적인 개발로 인해 위의 목록에 항상 서비스와 관련된 부울이 모두 포함되지 않을 수 있습니다. 나열하려면 다음 명령을 입력합니다.*

```
~]$ getsebool -a | grep service_name
```

특정 부울에 대한 설명을 보려면 다음 명령을 입력합니다.

```
~]$ sepolicy booleans -b boolean_name
```

*이 명령이 작동하려면 **sepolicy** 유틸리티를 제공하는 추가 **policycoreutils-devel** 패키지가 필요합니다.*

14.4. 설정 예

*다음 예제에서는 **SELinux**가 **Samba** 서버를 보완하는 방법과 **Samba** 서버의 전체 기능을 관리하는 방법을 보여주는 실제 데모를 제공합니다.*

14.4.1. 생성한 디렉토리를 공유

*다음 예제에서는 새 디렉터리를 만들고 **Samba**를 통해 해당 디렉터리를 공유합니다.*

1. ***samba, samba -common, samba-client** 패키지가 설치되어 있는지 확인합니다.*

```
~]$ rpm -q samba samba-common samba-client
package samba is not installed
package samba-common is not installed
package samba-client is not installed
```

*이러한 패키지가 설치되지 않은 경우 **yum** 유틸리티를 **root**로 사용하여 설치합니다.*

```
~]# yum install package-name
```

2. ***root**로 **mkdir** 유틸리티를 사용하여 **Samba**를 통해 파일을 공유할 새 최상위 디렉토리를 만듭니다.*

```
~]# mkdir /myshare
```

3. ***touch** 유틸리티 루트를 사용하여 빈 파일을 만듭니다. 이 파일은 나중에 **Samba** 공유가 올바르게 마운트되었는지 확인하는 데 사용됩니다.*

```
~]# touch /myshare/file1
```

4. ***SELinux**를 사용하면 **/etc/samba/smb.conf** 파일 및 **Linux** 권한이 적절하게 설정되는 한 **samba_share_t** 유형으로 레이블이 지정된 파일에 **Samba**가 읽고 쓸 수 있습니다. **root**로 다음 명령을 입력하여 레이블 변경 사항을 파일 컨텍스트 구성에 추가합니다.*

```
~]# semanage fcontext -a -t samba_share_t "/myshare(/.*)?"
```

5. ***restorecon** 유틸리티를 **root**로 사용하여 라벨 변경 사항을 적용합니다.*

```
~]# restorecon -R -v /myshare
restorecon reset /myshare context unconfined_u:object_r:default_t:s0-
>system_u:object_r:samba_share_t:s0
restorecon reset /myshare/file1 context unconfined_u:object_r:default_t:s0-
>system_u:object_r:samba_share_t:s0
```

6. */etc/samba/smb.conf* 를 루트로 편집합니다. 이 파일의 맨 아래에 다음을 추가하여 **Samba** 를 통해 **/myshare/** 디렉토리를 공유합니다.

```
[myshare]
comment = My share
path = /myshare
public = yes
writable = no
```

7. **Samba** 파일 시스템을 마운트하려면 **Samba** 계정이 필요합니다. **root**로 다음 명령을 입력하여 **Samba** 계정을 만듭니다. 여기서 **username** 은 기존 **Linux** 사용자입니다. 예를 들어 **smbpasswd -a testuser** 는 **Linux testuser** 사용자에 대한 **Samba** 계정을 만듭니다.

```
~]# smbpasswd -a testuser
New SMB password: Enter a password
Retype new SMB password: Enter the same password again
Added user testuser.
```

위의 명령을 입력하면 시스템에 없는 계정의 사용자 이름을 지정하면 **'username'!** 오류에 대한 **Unix** 계정을 찾을 수 없습니다.

8. **Samba** 서비스를 시작합니다.

```
~]# systemctl start smb.service
```

9. 다음 명령을 입력하여 사용 가능한 공유를 나열합니다. 여기서 **username** 은 **7**단계에 추가된 **Samba** 계정입니다. 암호를 입력하라는 메시지가 표시되면 **7**단계에서 **Samba** 계정에 할당된 암호를 입력합니다(버전 번호는 다를 수 있음).

```
~]$ smbclient -U username -L localhost
Enter username's password:
Domain=[HOSTNAME] OS=[Unix] Server=[Samba 3.4.0-0.41.el6]

Sharename       Type      Comment
---------       ----      -------
myshare         Disk      My share
IPC$            IPC       IPC Service (Samba Server Version 3.4.0-0.41.el6)
username        Disk      Home Directories
Domain=[HOSTNAME] OS=[Unix] Server=[Samba 3.4.0-0.41.el6]

Server               Comment
---------            -------
```

```
Workgroup            Master
---------            -------
```

10. **mkdir** 유틸리티를 **root**로 사용하여 새 디렉터리를 만듭니다. 이 디렉터리는 **myshare Samba** 공유를 마운트하는 데 사용됩니다.

```
~]# mkdir /test/
```

11. **root**로 다음 명령을 입력하여 **myshare Samba** 공유를 **/test/** 에 마운트하고 **username** 을 **7** 단계의 사용자 이름으로 바꿉니다.

```
~]# mount //localhost/myshare /test/ -o user=username
```

7단계에 구성된 사용자 이름의 암호를 입력합니다.

12. 다음 명령을 입력하여 **3**단계에서 만든 **file1** 파일을 확인합니다.

```
~]$ ls /test/
file1
```

14.4.2. 웹 사이트 공유

예를 들어 **/var/www/html/** 디렉터리에서 웹사이트를 공유하려는 경우 **samba_share_t** 유형으로 파일 레이블을 지정할 수 없습니다. 이러한 경우 **samba_export_all_ro** 부울을 사용하여 파일 또는 디렉터리(현재 레이블이 없는 경우)를 공유하고 읽기 전용 권한 또는 **samba_export_all_rw** 부울을 사용하여 모든 파일 또는 디렉터리(현재 레이블이 없는 경우)를 공유하고 읽기 및 쓰기 권한을 허용합니다.

다음 예제에서는 **/var/www/html/** 에 웹 사이트에 대한 파일을 만든 다음 **Samba**를 통해 해당 파일을 공유하여 읽기 및 쓰기 권한을 허용합니다. 이 예제에서는 **httpd,samba,samba-common,samba-client** 및 **wget** 패키지가 설치되어 있다고 가정합니다.

1. **root** 사용자로 **/var/www/html/file1.html** 파일을 만듭니다. 다음 콘텐츠를 복사하여 이 파일에 붙여넣습니다.

```
<html>
<h2>File being shared through the Apache HTTP Server and Samba.</h2>
</html>
```

2. 다음 명령을 입력하여 **file1.html**의 **SELinux** 컨텍스트를 확인합니다.

```
~]$ ls -Z /var/www/html/file1.html
-rw-r--r--. root root unconfined_u:object_r:httpd_sys_content_t:s0 /var/www/html/file1.html
```

파일은 **httpd_sys_content_t** 로 레이블이 지정됩니다. 기본적으로 **Apache HTTP** 서버는 이 유형에 액세스할 수 있지만 **Samba**는 이 유형에 액세스할 수 없습니다.

3. **Apache HTTP** 서버를 시작합니다.

```
~]# systemctl start httpd.service
```

4. 사용자가 에 대한 쓰기 액세스 권한이 있는 디렉터리로 변경하고 다음 명령을 입력합니다. 기본 구성을 변경하지 않는 한 이 명령은 성공합니다.

```
~]$ wget http://localhost/file1.html
Resolving localhost... 127.0.0.1
Connecting to localhost|127.0.0.1|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 84 [text/html]
Saving to: `file1.html.1'

100%[======================>] 84          --.-K/s   in 0s

`file1.html.1' saved [84/84]
```

5. **/etc/samba/smb.conf** 를 루트로 편집합니다. 이 파일의 맨 아래에 다음을 추가하여 **Samba** 를 통해 **/var/www/html/** 디렉토리를 공유합니다.

```
[website]
comment = Sharing a website
path = /var/www/html/
public = no
writable = no
```

6. **/var/www/html/** 디렉터리에는 **httpd_sys_content_t** 유형으로 레이블이 지정됩니다. 기본적으로 **Samba**는 **Linux** 권한이 허용하더라도 이 유형으로 레이블이 지정된 파일과 디렉터리에 액세스할 수 없습니다. **Samba** 액세스를 허용하려면 **samba_export_all_ro** 부울을 활성화합니다.

```
~]# setsebool -P samba_export_all_ro on
```

*재부팅 시 변경 사항이 유지되지 않도록 하려면 **-P** 옵션을 사용하지 마십시오.*
***samba_export_all_ro** 부울을 활성화하면 **Samba**가 모든 유형에 액세스할 수 있습니다.*

7. ***Samba** 서비스를 시작합니다.*

```
~]# systemctl start smb.service
```

---

[15]
*자세한 내용은 시스템 관리자 가이드의 **Samba** 섹션을 참조하십시오.*

15장. 파일 전송 프로토콜

FTP(파일 전송 프로토콜)는 오늘날 인터넷에서 발견된 가장 오래되고 가장 일반적으로 사용되는 프로토콜 중 하나입니다. 그 목적은 사용자가 원격 호스트에 직접 로그인하거나 원격 시스템을 사용하는 방법에 대한 지식이 없어도 네트워크에서 컴퓨터 호스트 간에 파일을 안정적으로 전송하는 것입니다. 이를 통해 사용자는 간단한 표준 명령 세트를 사용하여 원격 시스템의 파일에 액세스할 수 있습니다.

vsftpd(Very Secure FTP Daemon)는 처음부터 빠르고 안정적이며, 가장 중요한 것은 안전하도록 설계되었습니다. **vsftpd** 가 **Red Hat Enterprise Linux**와 함께 배포된 유일한 독립 실행형 **FTP**인 이유 때문에 많은 수의 연결을 효율적이고 안전하게 처리할 수 있습니다.

Red Hat Enterprise Linux에서 **vsftpd** 패키지는 **Very Secure FTP** 데몬을 제공합니다. 다음 명령을 입력하여 **vsftpd** 가 설치되어 있는지 확인합니다.

```
~]$ rpm -q vsftpd
package vsftpd is not installed
```

FTP 서버와 **vsftpd** 패키지가 설치되지 않도록 하려면 **root** 사용자로 **yum** 유틸리티를 사용하여 설치합니다.

```
~]# yum install vsftpd
```

15.1. 유형

고급 프로세스 격리를 제공하기 위해 **SELinux** 대상 정책에 사용되는 기본 권한 제어 방법은 **Type Enforcement**입니다. 모든 파일과 프로세스는 유형으로 레이블이 지정됩니다. **type**은 프로세스의 **SELinux** 도메인과 파일의 **SELinux** 유형을 정의합니다. **SELinux** 정책 규칙은 유형에 액세스하는 도메인이든 다른 도메인에 액세스하는 도메인이든 관계없이 유형이 서로 액세스하는 방법을 정의합니다. 액세스는 허용하는 특정 **SELinux** 정책 규칙이 있는 경우에만 허용됩니다.

기본적으로 익명 사용자는 **FTP**를 사용하여 로그인할 때 **/var/ftp/** 디렉토리의 파일에 대한 읽기 액세스 권한을 갖습니다. 이 디렉토리에는 **public_content_t** 유형으로 레이블이 지정되어 쓰기 액세스가 **/etc/vsftpd/vsftpd.conf** 에 구성되어 있는 경우에도 읽기 액세스만 허용합니다. **public_content_t** 유형은 **Apache HTTP Server, Samba, NFS** 등의 다른 서비스에 액세스할 수 있습니다.

다음 유형 중 하나를 사용하여 **FTP**를 통해 파일을 공유합니다.

public_content_t

public_content_t 유형으로 만든 파일과 디렉터리에 레이블을 지정하여 **vsftpd** 를 통해 읽기 전

용으로 공유합니다. **Apache HTTP Server, Samba, NFS** 등의 기타 서비스도 이 유형으로 레이블이 지정된 파일에 액세스할 수 있습니다. **Linux** 권한이 쓰기 액세스를 허용하는 경우에도 **public_content_t** 유형으로 레이블이 지정된 파일은 에 쓸 수 없습니다. 쓰기 액세스 권한이 필요한 경우 **public_content_rw_t** 유형을 사용합니다.

public_content_rw_t

public_content_rw_t 유형으로 만든 파일과 디렉터리에 레이블을 지정하여 **vsftpd** 를 통해 읽기 및 쓰기 권한으로 공유합니다. **Apache HTTP Server, Samba, NFS** 등의 기타 서비스도 이 유형으로 레이블이 지정된 파일에 액세스할 수 있습니다. 각 서비스에 대한 부울은 이 유형으로 레이블이 지정된 파일에 쓰기 전에 활성화해야 합니다.

15.2. 부울

SELinux는 서비스를 실행하는 데 필요한 최소 액세스 수준을 기반으로 합니다. 서비스는 다양한 방법으로 실행할 수 있으므로 서비스 실행 방법을 지정해야 합니다. 다음 부울을 사용하여 **SELinux**를 설정합니다.

ftpd_anon_write

비활성화된 경우 이 부울은 **vsftpd** 가 **public_content_rw_t** 유형으로 레이블이 지정된 파일 및 디렉터리에 쓰는 것을 금지합니다. 사용자가 **FTP**를 사용하여 파일을 업로드할 수 있도록 이 부울을 활성화합니다. 파일이 업로드되는 디렉터리는 **public_content_rw_t** 유형 및 **Linux** 권한으로 레이블 지정해야 합니다.

ftpd_full_access

이 부울이 활성화되면 **DAC(Linux)** 권한만 사용하여 액세스를 제어하며 인증된 사용자는 **public_content_ t** 또는 **public_content_rw_t** 유형으로 레이블이 지정되지 않은 파일에 읽고 쓸 수 있습니다.

ftpd_use_cifs

이 부울을 활성화하면 **vsftpd** 가 **cifs_t** 유형으로 레이블이 지정된 파일과 디렉터리에 액세스할 수 있으므로 이 부울을 활성화하면 **Samba**를 통해 마운트된 파일 시스템을 공유할 수 있습니다 .

ftpd_use_nfs

이 부울을 활성화하면 **vsftpd** 가 **nfs_t** 유형으로 레이블이 지정된 파일과 디렉터리에 액세스할 수 있으므로 이 부울을 사용하면 **NFS**를 사용하여 마운트된 파일 시스템을 공유할 수 있습니다 .

ftpd_connect_db

FTP 데몬이 데이터베이스에 대한 연결을 시작할 수 있도록 허용합니다.

httpd_enable_ftp_server

httpd 데몬이 **FTP** 포트에서 수신 대기하고 **FTP** 서버 역할을 하도록 허용합니다.

tftp_anon_write

이 부울을 활성화하면 특수한 액세스 제한이 없는 공통 파일용으로 예약된 영역과 같은 공용 디렉터리에 **TFTP** 액세스가 허용됩니다.

중요

Red Hat Enterprise Linux 7.7에서는 **ftp_home_dir** 부울을 제공하지 않습니다. 자세한 내용은 **Red Hat Enterprise Linux 7.3** 릴리스 노트 문서를 참조하십시오.

참고

SELinux 정책의 지속적인 개발로 인해 위의 목록에 항상 서비스와 관련된 부울이 모두 포함되지 않을 수 있습니다. 나열하려면 다음 명령을 입력합니다.

```
~]$ getsebool -a | grep service_name
```

특정 부울에 대한 설명을 보려면 다음 명령을 입력합니다.

```
~]$ sepolicy booleans -b boolean_name
```

이 명령이 작동하려면 **sepolicy** 유틸리티를 제공하는 추가 **policycoreutils-devel** 패키지가 필요합니다.

16장. 네트워크 파일 시스템

NFS(네트워크 파일 시스템)를 사용하면 원격 호스트가 네트워크를 통해 파일 시스템을 마운트하고 로컬로 마운트된 파일 시스템과 상호 작용할 수 있습니다. 이를 통해 시스템 관리자는 네트워크의 중앙 집중식 서버에 리소스를 통합할 수 있습니다.[16]

Red Hat Enterprise Linux에서는 전체 NFS 지원에 nfs-utils 패키지가 필요합니다. 다음 명령을 입력하여 nfs-utils 가 설치되었는지 확인합니다.

```
~]$ rpm -q nfs-utils
package nfs-utils is not installed
```

설치되지 않고 NFS를 사용하려면 yum 유틸리티를 root로 사용하여 설치합니다.

```
~]# yum install nfs-utils
```

16.1. NFS 및 SELINUX

SELinux를 실행하는 경우 NFS 데몬은 제한되지 않은 kernel_t 도메인 유형으로 레이블이 지정된 nfsd 프로세스를 제외하고 기본적으로 제한됩니다. SELinux 정책을 사용하면 기본적으로 NFS가 파일을 공유할 수 있습니다. 또한 클라이언트와 서버 간에 SELinux 레이블을 전달하여 NFS 볼륨에 액세스하는 제한된 도메인의 보안 제어를 강화할 수 있습니다. 예를 들어, 홈 디렉터리가 NFS 볼륨에 설정된 경우 볼륨의 다른 디렉터리가 아닌 홈 디렉터리에만 액세스할 수 있는 제한된 도메인을 지정할 수 있습니다. 마찬가지로 Secure Virtualization과 같은 애플리케이션에서는 NFS 볼륨에 이미지 파일의 레이블을 설정하여 가상 머신의 분리 수준을 높일 수 있습니다.

레이블이 지정된 NFS에 대한 지원은 기본적으로 비활성화되어 있습니다. 활성화하려면 16.4.1절. “SELinux 레이블이 지정된 NFS 지원 활성화” 을 참조하십시오.

16.2. 유형

고급 프로세스 격리를 제공하기 위해 SELinux 대상 정책에 사용되는 기본 권한 제어 방법은 Type Enforcement입니다. 모든 파일과 프로세스는 유형으로 레이블이 지정됩니다. type은 프로세스의 SELinux 도메인과 파일의 SELinux 유형을 정의합니다. SELinux 정책 규칙은 유형에 액세스하는 도메인이든 다른 도메인에 액세스하는 도메인이든 관계없이 유형이 서로 액세스하는 방법을 정의합니다. 액세스는 허용하는 특정 SELinux 정책 규칙이 있는 경우에만 허용됩니다.

기본적으로 클라이언트쪽에 마운트된 NFS 볼륨은 NFS의 정책에서 정의한 기본 컨텍스트로 레이블이 지정됩니다. 일반적인 정책에서 이 기본 컨텍스트는 nfs_t 유형을 사용합니다. root 사용자는 mount -

context 옵션을 사용하여 기본 유형을 재정의할 수 있습니다. 다음 유형은 NFS와 함께 사용됩니다. 다양한 유형을 사용하여 유연한 액세스를 구성할 수 있습니다.

var_lib_nfs_t

이 유형은 /var/lib/nfs/ 디렉터리에 복사되거나 생성된 기존 및 새 파일에 사용됩니다. 이 유형은 정상적인 작업에서 변경할 필요가 없습니다. 기본 설정의 변경 사항을 복원하려면 root 사용자로 restorecon -R -v /var/lib/nfs 명령을 실행합니다.

nfsd_exec_t

NFS와 관련된 다른 시스템 실행 파일 및 라이브러리이므로 /usr/sbin/rpc.nfsd 파일에는 nfsd_exec_t 로 레이블이 지정됩니다. 사용자는 이 유형의 파일에 레이블을 지정하지 않아야 합니다. nfsd_exec_t 는 nfsd_t 로 전환됩니다.

16.3. 부울

SELinux는 서비스를 실행하는 데 필요한 최소 액세스 수준을 기반으로 합니다. 서비스는 다양한 방법으로 실행할 수 있으므로 서비스 실행 방법을 지정해야 합니다. 다음 부울을 사용하여 SELinux를 설정합니다.

ftpd_use_nfs

이 부울을 활성화하면 ftpd 데몬이 NFS 볼륨에 액세스할 수 있습니다.

cobbler_use_nfs

이 부울을 활성화하면 cobble rd 데몬이 NFS 볼륨에 액세스할 수 있습니다.

git_system_use_nfs

이 부울을 활성화하면 Git 시스템 데몬이 NFS 볼륨의 시스템 공유 리포지토리를 읽을 수 있습니다.

httpd_use_nfs

이 부울을 활성화하면 httpd 데몬이 NFS 볼륨에 저장된 파일에 액세스할 수 있습니다.

samba_share_nfs

이 부울을 활성화하면 smbd 데몬이 NFS 볼륨을 공유할 수 있습니다. 이 부울을 비활성화하면 smbd 가 Samba를 사용하여 NFS 공유에 대한 전체 액세스 권한을 가질 수 없습니다.

sanlock_use_nfs

*이 부울을 활성화하면 **sanlock** 데몬이 **NFS** 볼륨을 관리할 수 있습니다.*

sge_use_nfs

*이 부울을 활성화하면 **sge** 스케줄러가 **NFS** 볼륨에 액세스할 수 있습니다.*

use_nfs_home_dirs

*이 부울을 활성화하면 **NFS** 홈 디렉토리에 대한 지원이 추가됩니다.*

virt_use_nfs

*이 부울을 사용하면 가상 게스트가 **NFS** 볼륨의 파일을 관리할 수 있습니다.*

xen_use_nfs

*이 부울을 활성화하면 **Xen** 이 **NFS** 볼륨의 파일을 관리할 수 있습니다.*

git_cgi_use_nfs

*이 부울을 활성화하면**CGI(Git Common Gateway Interface)**가 **NFS** 볼륨에 액세스할 수 있습니다.*

참고

__SELinux__ 정책의 지속적인 개발로 인해 위의 목록에 항상 서비스와 관련된 부울이 모두 포함되지 않을 수 있습니다. 나열하려면 다음 명령을 입력합니다.

```
~]$ getsebool -a | grep service_name
```

특정 부울에 대한 설명을 보려면 다음 명령을 입력합니다.

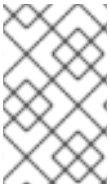
```
~]$ sepolicy booleans -b boolean_name
```

*이 명령이 작동하려면 **sepolicy** 유틸리티를 제공하는 추가 **policycoreutils-devel** 패키지가 필요합니다.*

16.4. 설정 예

16.4.1. SELinux 레이블이 지정된 NFS 지원 활성화

다음 예제에서는 **SELinux** 레이블이 지정된 **NFS** 지원을 활성화하는 방법을 보여줍니다. 이 예제에서는 **nfs-utils** 패키지가 설치되어 있고, **SELinux** 대상 정책이 사용되며 **SELinux**가 강제 모드로 실행 중이라고 가정합니다.



참고

1-3단계는 **NFS** 서버 **nfs-srv** 에서 수행해야 합니다.

1. **NFS** 서버가 실행 중인 경우 중지하십시오.

```
[nfs-srv]# systemctl stop nfs
```

서버가 중지되었는지 확인합니다.

```
[nfs-srv]# systemctl status nfs
nfs-server.service - NFS Server
   Loaded: loaded (/usr/lib/systemd/system/nfs-server.service; disabled)
   Active: inactive (dead)
```

2. **/etc/sysconfig/nfs** 파일을 편집하여 **RPCNFSDARGS** 플래그를 **"-V 4.2"** 로 설정하십시오.

```
# Optional arguments passed to rpc.nfsd. See rpc.nfsd(8)
RPCNFSDARGS="-V 4.2"
```

3. 서버를 다시 시작하고 실행 중인지 확인합니다. 출력에는 아래의 정보가 포함되어 있으며 타임스탬프만 다릅니다.

```
[nfs-srv]# systemctl start nfs
```

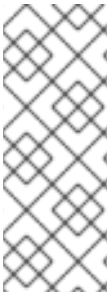
```
[nfs-srv]# systemctl status nfs
nfs-server.service - NFS Server
   Loaded: loaded (/usr/lib/systemd/system/nfs-server.service; disabled)
   Active: active (exited) since Wed 2013-08-28 14:07:11 CEST; 4s ago
```

4. *클라이언트쪽에서 **NFS** 서버를 마운트합니다.*

```
[nfs-client]# mount -o v4.2 server:mntpoint localmountpoint
```

5. *이제 모든 **SELinux** 레이블이 서버에서 클라이언트로 성공적으로 전달됩니다.*

```
[nfs-srv]$ ls -Z file
-rw-rw-r--. user user unconfined_u:object_r:svirt_image_t:s0 file
[nfs-client]$ ls -Z file
-rw-rw-r--. user user unconfined_u:object_r:svirt_image_t:s0 file
```



참고

*홈 디렉토리 또는 기타 콘텐츠에 대해 레이블이 지정된 **NFS** 지원을 활성화하면 해당 콘텐츠에 **EXT** 파일 시스템에 있던 것과 동일한 레이블이 지정됩니다. 또한 다양한 버전의 **NFS**로 시스템을 마운트하거나 레이블이 지정된 **NFS**를 지원하지 않는 서버를 마운트하려고 하면 오류가 반환될 수 있습니다.*

---

[16]
*자세한 내용은 스토리지 관리 가이드 의 **NFS**(네트워크 파일 시스템) 장을 참조하십시오.*

17장. 영국 인터넷 이름 도메인

BIND는 명명된 데몬을 사용하여 이름 확인 서비스를 수행합니다. **BIND**를 사용하면 숫자 주소가 아니라 이름별로 컴퓨터 리소스 및 서비스를 찾을 수 있습니다.

Red Hat Enterprise Linux에서 **bind** 패키지는 **DNS** 서버를 제공합니다. 다음 명령을 입력하여 **bind** 패키지가 설치되어 있는지 확인합니다.

```
~]$ rpm -q bind
package bind is not installed
```

설치되지 않은 경우 **yum** 유틸리티를 **root** 사용자로 사용하여 설치합니다.

```
~]# yum install bind
```

17.1. BIND 및 SELINUX

/var/ named/slaves/, /var/ named/ dynamic/ 및 **/var/named /data/** 디렉터리에 대한 기본 권한을 사용하면 영역 전송 및 동적 **DNS** 업데이트를 사용하여 영역 파일을 업데이트할 수 있습니다. **/var/named/** 의 파일은 마스터 영역 파일에 사용되는 **named_zone_t** 유형으로 레이블이 지정됩니다.

슬레이브 서버의 경우 슬레이브 영역을 **/ var/named/slaves/**에 배치하도록 **/etc/named.conf** 파일을 구성합니다. 다음은 **testdomain .com**의 영역 파일을 **/var/named/slaves/** 에 저장하는 슬레이브 **DNS** 서버의 도메인 항목의 예입니다.

```
zone "testdomain.com" {
   type slave;
   masters { IP-address; };
   file "/var/named/slaves/db.testdomain.com";
        };
```

영역 파일에 **named_zone_t** 레이블이 지정된 경우 영역 전송 및 동적 **DNS**가 영역 파일을 업데이트할 수 있도록 **named_write_master_zones** 부울을 활성화해야 합니다. 또한 명명된 사용자 또는 그룹이 읽기, 쓰기, 실행 액세스를 허용하도록 상위 디렉터리의 모드를 변경해야 합니다.

/var/named/ 의 영역 파일에 **named_cache_t** 유형으로 레이블이 지정되면 파일 시스템의 레이블을 다시 지정하거나 **restorecon -R /var/** 를 실행하면 해당 유형이 **named_zone_t** 로 변경됩니다.

17.2. 유형

*고급 프로세스 격리를 제공하기 위해 **SELinux** 대상 정책에 사용되는 기본 권한 제어 방법은 **Type Enforcement**입니다. 모든 파일과 프로세스는 유형으로 레이블이 지정됩니다. **type**은 프로세스의 **SELinux** 도메인과 파일의 **SELinux** 유형을 정의합니다. **SELinux** 정책 규칙은 유형에 액세스하는 도메인이든 다른 도메인에 액세스하는 도메인이든 관계없이 유형이 서로 액세스하는 방법을 정의합니다. 액세스는 허용하는 특정 **SELinux** 정책 규칙이 있는 경우에만 허용됩니다.*

*다음 유형은 **BIND**와 함께 사용됩니다. 다양한 유형을 사용하여 유연한 액세스를 구성할 수 있습니다.*

named_zone_t

*마스터 영역 파일에 사용됩니다. 다른 서비스는 이 유형의 파일을 수정할 수 없습니다. 명명된 데몬은 **named_write_master_zones** 부울이 활성화된 경우에만 이 유형의 파일을 수정할 수 있습니다.*

named_cache_t

*기본적으로 **named** 는 추가 부울이 설정되지 않고 이 유형으로 레이블이 지정된 파일에 쓸 수 있습니다. **/var/ named/slaves/**, **/var/ named/dynamic/** 및 **/var/ named/data/** 디렉터리에 복사하거나 만든 파일은 **named_cache_t** 유형으로 자동으로 레이블이 지정됩니다.*

named_var_run_t

/var/run/ bind/, **/var/run /named/** 및 **/var/ run/unbound/** 디렉터리에 복사하거나 생성된 파일은 **named_var_run_t** 유형으로 자동으로 레이블이 지정됩니다.*

named_conf_t

*일반적으로 **/etc** 디렉토리에 저장된 **BIND** 관련 구성 파일은 자동으로 **named_conf_t** 유형으로 레이블이 지정됩니다.*

named_exec_t

*일반적으로 **/usr/sbin/** 디렉토리에 저장된 **BIND** 관련 실행 파일은 **named_exec_t** 유형으로 자동으로 레이블이 지정됩니다.*

named_log_t

*일반적으로 **/var/log/** 디렉토리에 저장된 **BIND** 관련 로그 파일은 자동으로 **named_log_t** 유형으로 레이블이 지정됩니다.*

named_unit_file_t

/usr/lib/systemd/system/ 디렉토리에 있는 실행 가능한 **BIND** 관련 파일은 **named_unit_file_t**

유형으로 자동으로 레이블이 지정됩니다.

17.3. 부울

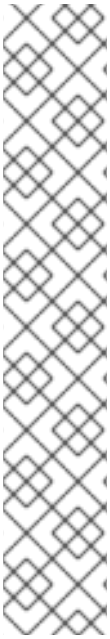
SELinux는 서비스를 실행하는 데 필요한 최소 액세스 수준을 기반으로 합니다. 서비스는 다양한 방법으로 실행할 수 있으므로 서비스 실행 방법을 지정해야 합니다. 다음 부울을 사용하여 **SELinux**를 설정합니다.

named_write_master_zones

비활성화된 경우 이 부울은 **named**가 **named _zone_t** 유형으로 레이블이 지정된 영역 파일 또는 디렉터리에 쓰는 것을 금지합니다. 데몬은 일반적으로 영역 파일에 쓸 필요가 없지만 이 부울이 필요한 경우 또는 보조 서버가 영역 파일에 작성해야 하는 경우 이 부울을 활성화하여 이 작업을 허용합니다.

named_tcp_bind_http_port

이 부울을 활성화하면 **BIND**에서 **Apache** 포트를 바인딩할 수 있습니다.



참고

SELinux 정책의 지속적인 개발로 인해 위의 목록에 항상 서비스와 관련된 부울이 모두 포함되지 않을 수 있습니다. 나열하려면 다음 명령을 입력합니다.

```
~]$ getsebool -a | grep service_name
```

특정 부울에 대한 설명을 보려면 다음 명령을 입력합니다.

```
~]$ sepolicy booleans -b boolean_name
```

이 명령이 작동하려면 **sepolicy** 유틸리티를 제공하는 추가 **policycoreutils-devel** 패키지가 필요합니다.

17.4. 설정 예

17.4.1. 동적 DNS

BIND를 사용하면 호스트가 **DNS** 및 영역 파일에서 레코드를 동적으로 업데이트할 수 있습니다. 이는 호스트 컴퓨터의 **IP** 주소가 자주 변경되고 **DNS** 레코드에 실시간 수정이 필요한 경우에 사용됩니다.

*동적 **DNS**로 업데이트할 영역 파일에 **/var/named/dynamic/** 디렉터리를 사용합니다. 에 생성되거나 이 디렉토리에 복사된 파일은 **named** 에서 쓸 수 있는 **Linux** 권한을 상속합니다. 이러한 파일은 **named_cache_t** 유형으로 레이블이 지정되므로 **SELinux**는 **named** 에서 해당 파일에 쓸 수 있도록 허용합니다.*

*/**var/named/dynamic**/ 의 영역 파일에 **named_zone_t** 유형으로 레이블이 지정된 경우 병합하기 전에 먼저 저널에 업데이트를 작성해야 하므로 동적 **DNS** 업데이트가 특정 기간 동안 성공하지 못할 수 있습니다. 저널을 병합하려고 할 때 영역 파일에 **named_zone_t** 유형으로 레이블이 지정되면 다음과 같은 오류가 기록됩니다.*

```
named[PID]: dumping master file: rename: /var/named/dynamic/zone-name: permission denied
```

*또한 다음 **SELinux** 거부 메시지가 기록됩니다.*

```
setroubleshoot: SELinux is preventing named (named_t) "unlink" to zone-name (named_zone_t)
```

*이 레이블 문제를 해결하려면 **root**로 **restorecon** 유틸리티를 사용합니다.*

```
~]# restorecon -R -v /var/named/dynamic
```

18장. 동시 버전 관리 시스템

CVS(Concurrent Versioning System)는 무료 개정 제어 시스템입니다. 이 명령은 일반적으로 여러 사용자가 액세스하는 중앙 파일 집합에 대한 수정 사항을 모니터링하고 추적하는 데 사용됩니다. 프로그래머는 일반적으로 소스 코드 리포지토리를 관리하는 데 사용되며 오픈 소스 개발자가 널리 사용합니다.

Red Hat Enterprise Linux에서 **cvs** 패키지는 **CVS**를 제공합니다. 다음 명령을 입력하여 **cvs** 패키지가 설치되어 있는지 확인합니다.

```
~]$ rpm -q cvs
package cvs is not installed
```

설치되어 있지 않고 **CVS**를 사용하려면 **yum** 유틸리티를 **root**로 사용하여 설치합니다.

```
~]# yum install cvs
```

18.1. CVS 및 SELINUX

cvs 데몬은 **cvs_t** 유형으로 레이블이 지정됩니다. 기본적으로 **Red Hat Enterprise Linux**에서 **CVS**는 특정 디렉토리를 읽고 쓸 수 있습니다. **cvs_data_t** 레이블은 **cvs** 에 대해 읽기 및 쓰기 액세스 권한이 있는 영역을 정의합니다. **SELinux**에서 **CVS**를 사용하는 경우 클라이언트가 **CVS** 데이터에 예약된 영역에 대한 전체 액세스 권한을 보유하려면 올바른 레이블을 할당해야 합니다.

18.2. 유형

고급 프로세스 격리를 제공하기 위해 **SELinux** 대상 정책에 사용되는 기본 권한 제어 방법은 **Type Enforcement**입니다. 모든 파일과 프로세스는 유형으로 레이블이 지정됩니다. **type**은 프로세스의 **SELinux** 도메인과 파일의 **SELinux** 유형을 정의합니다. **SELinux** 정책 규칙은 유형에 액세스하는 도메인이든 다른 도메인에 액세스하는 도메인이든 관계없이 유형이 서로 액세스하는 방법을 정의합니다. 액세스는 허용하는 특정 **SELinux** 정책 규칙이 있는 경우에만 허용됩니다.

CVS와 함께 다음 유형을 사용합니다. 다양한 유형을 사용하여 유연한 액세스를 구성할 수 있습니다.

cvs_data_t

이 유형은 **CVS** 리포지토리의 데이터에 사용됩니다. **CVS**는 이러한 유형이 있는 경우에만 데이터에 대한 전체 액세스 권한을 얻을 수 있습니다.

cvs_exec_t

이 유형은 **/usr/bin/cvs** 바이너리에 사용됩니다.

18.3. 부울

SELinux는 서비스를 실행하는 데 필요한 최소 액세스 수준을 기반으로 합니다. 서비스는 다양한 방법으로 실행할 수 있으므로 서비스 실행 방법을 지정해야 합니다. 다음 부울을 사용하여 **SELinux**를 설정합니다.

cvs_read_shadow

이 부울을 사용하면 **cvs** 데몬이 사용자 인증을 위해 **/etc/shadow** 파일에 액세스할 수 있습니다.

참고

SELinux 정책의 지속적인 개발로 인해 위의 목록에 항상 서비스와 관련된 부울이 모두 포함되지 않을 수 있습니다. 나열하려면 다음 명령을 입력합니다.

```
~]$ getsebool -a | grep service_name
```

특정 부울에 대한 설명을 보려면 다음 명령을 입력합니다.

```
~]$ sepolicy booleans -b boolean_name
```

이 명령이 작동하려면 **sepolicy** 유틸리티를 제공하는 추가 **policycoreutils-devel** 패키지가 필요합니다.

18.4. 설정 예

18.4.1. CVS 설정

이 예제에서는 간단한 **CVS** 설정과 원격 액세스를 허용하는 **SELinux** 구성을 설명합니다. 이 예에서는 두 개의 호스트 이름이 **cvs-srv** 이며 **IP** 주소가 **192.168.1.1** 이고 호스트 이름이 **cvs-client** 인 클라이언트와 **IP** 주소가 **192.168.1.100** 인 **CVS** 서버가 사용됩니다. 두 호스트 모두 동일한 서브넷(**192.168.1.0/24**)에 있습니다. 이 예제는 **cvs** 및 **xinetd** 패키지가 설치되어 있고, **SELinux** 대상 정책이 사용되고 **SELinux**가 강제 모드로 실행 중이라고 가정하는 유일한 예입니다.

이 예에서는 전체 **DAC** 권한을 가진 경우에도 **SELinux**가 파일 레이블을 기반으로 정책 규칙을 시행할 수 있으며 **CVS**에서 액세스하기 위해 특별히 레이블이 지정된 특정 영역에만 액세스할 수 있음을 보여

줍니다.

참고

1~9단계는 CVS 서버 cvs-srv 에서 수행되어야 합니다.

1. *이 예에서는 **cvs** 및 **xinetd** 패키지가 필요합니다. 패키지가 설치되었는지 확인합니다.*

```
[cvs-srv]$ rpm -q cvs xinetd
package cvs is not installed
package xinetd is not installed
```

*설치되지 않은 경우 **yum** 유틸리티를 **root**로 사용하여 설치합니다.*

```
[cvs-srv]# yum install cvs xinetd
```

2. *다음 명령을 **root**로 입력하여 **CVS** 라는 그룹을 만듭니다.*

```
[cvs-srv]# groupadd CVS
```

*이 작업은 **system-config-users** 유틸리티를 사용하여 수행할 수도 있습니다.*

3. *사용자 이름이 **cvsuser** 인 사용자를 만들고 이 사용자를 **CVS** 그룹의 멤버로 설정합니다. 이 작업은 **system-config-users** 를 사용하여 수행할 수 있습니다.*

4. */**etc**/**services** 파일을 편집하고 **CVS** 서버에 다음과 유사한 항목이 있는지 확인합니다.*

```
cvspserver 2401/tcp   # CVS client/server operations
cvspserver 2401/udp   # CVS client/server operations
```

5. *파일 시스템의 루트 영역에 **CVS** 리포지토리를 만듭니다. **SELinux**를 사용하는 경우 다른 하위 디렉터리에 영향을 주지 않고 재귀 레이블을 제공할 수 있도록 루트 파일 시스템에 리포지토리가 있는 것이 가장 좋습니다. 예를 들어 **root**로 리포지토리를 저장할 **/cvs/** 디렉토리를 생성합니다.*

```
[root@cvs-srv]# mkdir /cvs
```

6. 모든 사용자에게 **/cvs/** 디렉토리에 대한 전체 권한을 부여합니다.

```
[root@cvs-srv]# chmod -R 777 /cvs
```

주의

이는 예제일 뿐이며 이러한 권한은 프로덕션 시스템에서 사용해서는 안 됩니다.

7. **/etc/xinetd.d/cvs** 파일을 편집하고 **CVS** 섹션의 주석 처리되지 않고 **/cvs/** 디렉터리를 사용하도록 구성되어 있는지 확인합니다. 파일은 다음과 유사해야 합니다.

```
service cvspserver
{
 disable = no
 port   = 2401
 socket_type  = stream
 protocol  = tcp
 wait   = no
 user   = root
 passenv   = PATH
 server   = /usr/bin/cvs
 env   = HOME=/cvs
 server_args  = -f --allow-root=/cvs pserver
# bind   = 127.0.0.1
```

8. **xinetd** 데몬을 시작합니다.

```
[cvs-srv]# systemctl start xinetd.service
```

9. **system-config-firewall** 유틸리티를 사용하여 포트 **2401**에서 **TCP**를 통한 인바운드 연결을 허용하는 규칙을 추가합니다.

10. 클라이언트 측에서 **cvsuser** 사용자로 다음 명령을 입력합니다.

```
[cvsuser@cvs-client]$ cvs -d /cvs init
```

11.

*이때 **CVS**는 구성되었지만 **SELinux**는 여전히 로그인 및 파일 액세스를 거부합니다. 이를 시연하려면 **cvs-client** 에서 **$CVSROOT** 변수를 설정하고 원격으로 로그인하십시오. 다음 단계는 **cvs-client** 에서 수행해야 합니다.*

```
[cvsuser@cvs-client]$ export CVSROOT=:pserver:cvsuser@192.168.1.1:/cvs
[cvsuser@cvs-client]$
[cvsuser@cvs-client]$ cvs login
Logging in to :pserver:cvsuser@192.168.1.1:2401/cvs
CVS password: ********
cvs [login aborted]: unrecognized auth response from 192.168.100.1: cvs pserver: cannot
open /cvs/CVSROOT/config: Permission denied
```

*SELinux가 액세스를 차단했습니다. SELinux가 이러한 액세스를 허용하려면 **cvs-srv** 에서 다음 단계를 수행해야 합니다.*

12.

*/**cvs**/ 디렉토리의 기존 및 새 데이터에 재귀적으로 레이블을 지정하여 **cvs_data_t** 유형을 지정하도록 /**cvs**/ 디렉터리의 컨텍스트를 **root**로 변경합니다.*

```
[root@cvs-srv]# semanage fcontext -a -t cvs_data_t '/cvs(/.*)?'
[root@cvs-srv]# restorecon -R -v /cvs
```

13.

*클라이언트 **cvs-client** 는 이제 이 리포지토리의 모든 **CVS** 리소스에 로그인하고 액세스할 수 있어야 합니다.*

```
[cvsuser@cvs-client]$ export CVSROOT=:pserver:cvsuser@192.168.1.1:/cvs
[cvsuser@cvs-client]$
[cvsuser@cvs-client]$ cvs login
Logging in to :pserver:cvsuser@192.168.1.1:2401/cvs
CVS password: ********
[cvsuser@cvs-client]$
```

19장. SQUIID 캐싱 프록시

Squid는 **FTP, Gopher** 및 **HTTP** 데이터 개체를 지원하는 웹 클라이언트를 위한 고성능 프록시 캐싱 서버입니다. 자주 요청되는 웹 페이지를 캐싱하고 다시 사용하여 대역폭을 줄이고 응답 시간을 개선합니다.[17]

Red Hat Enterprise Linux에서 **squid** 패키지는 **Squid** 캐싱 프록시를 제공합니다. 다음 명령을 입력하여 **squid** 패키지가 설치되어 있는지 확인합니다.

```
~]$ rpm -q squid
package squid is not installed
```

설치되지 않고 **squid**를 사용하려면 **yum** 유틸리티를 **root**로 사용하여 설치합니다.

```
~]# yum install squid
```

19.1. SQUIID 캐싱 프록시 및 SELINUX

SELinux가 활성화되면 **Squid**는 기본적으로 제한된 상태로 실행됩니다. 제한된 프로세스는 자체 도메인에서 실행되며 다른 제한된 프로세스와 분리됩니다. **SELinux** 정책 구성에 따라 공격자가 제한된 프로세스가 손상되면 공격자가 리소스에 대한 액세스와 가능한 손상을 제한합니다. 다음 예제에서는 자체 도메인에서 실행되는 **Squid** 프로세스를 보여줍니다. 이 예제에서는 **squid** 패키지가 설치되어 있다고 가정합니다.

1. **getenforce** 명령을 실행하여 **SELinux**가 강제 모드로 실행 중인지 확인합니다.

   ```
   ~]$ getenforce
   Enforcing
   ```

   명령은 **SELinux**가 강제 모드에서 실행 중일 때 **Enforcing** (강제)을 반환합니다.

2. **root** 사용자로 다음 명령을 입력하여 **squid** 데몬을 시작합니다.

   ```
   ~]# systemctl start squid.service
   ```

   서비스가 실행 중인지 확인합니다. 출력에는 아래 정보가 포함되어야 합니다(시간 스탬프만 다릅니다).

```
~]# systemctl status squid.service
squid.service - Squid caching proxy
   Loaded: loaded (/usr/lib/systemd/system/squid.service; disabled)
   Active: active (running) since Mon 2013-08-05 14:45:53 CEST; 2s ago
```

3. 다음 명령을 입력하여 **squid** 프로세스를 확인합니다.

```
~]$ ps -eZ | grep squid
system_u:system_r:squid_t:s0    27018 ?        00:00:00 squid
system_u:system_r:squid_t:s0    27020 ?        00:00:00 log_file_daemon
```

squid 프로세스와 연결된 **SELinux** 컨텍스트는 **system_u:system_r:squid_t:s0**입니다. 컨텍스트의 마지막 부분인 **squid_t** 는 유형입니다. 유형은 프로세스 및 파일의 유형에 대한 도메인을 정의합니다. 이 경우 **Squid** 프로세스는 **squid_t** 도메인에서 실행됩니다.

SELinux 정책은 제한된 도메인(예: **squid_t**)에서 실행되는 프로세스를 파일, 기타 프로세스 및 일반적인 시스템과 상호 작용하는 방법을 정의합니다. **squid** 액세스를 허용하려면 파일에 올바르게 레이블이 지정되어야 합니다.

squid가 기본 **TCP** 포트 **3128, 3401** 또는 **4827** 이외의 포트에서 수신 대기 하도록 **/etc/squid/squid.conf** 파일을 구성하는 경우 **semanage port** 명령을 사용하여 **SELinux** 정책 구성에 필요한 포트 번호를 추가해야 합니다. 다음 예제에서는 **SELinux** 정책 구성에 처음에 정의되지 않은 포트에서 수신 대기하도록 **squid** 를 구성하는 방법을 보여주므로 서버가 시작되지 않습니다. 이 예제에서는 데몬이 정책에 아직 정의되지 않은 비표준 포트에서 수신 대기하도록 **SELinux** 시스템을 구성하는 방법도 보여줍니다. 이 예에서는 **squid** 패키지가 설치되어 있다고 가정합니다. 예제에서 **root** 사용자로 각 명령을 실행합니다.

1. **squid** 데몬이 실행 중이 아닌지 확인합니다.

```
~]# systemctl status squid.service
squid.service - Squid caching proxy
   Loaded: loaded (/usr/lib/systemd/system/squid.service; disabled)
   Active: inactive (dead)
```

출력이 다르면 프로세스를 중지합니다.

```
~]# systemctl stop squid.service
```

2. **SELinux**가 **squid**가 수신 대기 할 수 있는 포트를 보려면 다음 명령을 입력합니다.

```
~]# semanage port -l | grep -w -i squid_port_t
squid_port_t                tcp      3401, 4827
squid_port_t                udp      3401, 4827
```

3. */etc/squid/squid.conf* 를 루트로 편집합니다. **squid**에 대해 **SELinux** 정책 구성에 구성되지 않은 포트를 나열하도록 **http_port** 옵션을 구성합니다 . 이 예에서는 데몬이 포트 **10000**에서 수신 대기하도록 구성되어 있습니다.

```
# Squid normally listens to port 3128
http_port 10000
```

4. **setsebool** 명령을 실행하여 **squid_connect_any** 부울이 **off**로 설정되어 있는지 확인합니다. 이렇게 하면 **squid** 가 특정 포트에서만 작동할 수 있습니다.

```
~]# setsebool -P squid_connect_any 0
```

5. **squid** 데몬을 시작합니다.

```
~]# systemctl start squid.service
Job for squid.service failed. See 'systemctl status squid.service' and 'journalctl -xn' for details.
```

다음과 유사한 **SELinux** 거부 메시지가 기록됩니다.

```
localhost setroubleshoot: SELinux is preventing the squid (squid_t) from binding to port
10000. For complete SELinux messages. run sealert -l 97136444-4497-4fff-a7a7-
c4d8442db982
```

6. **SELinux**가 포트 **10000**에서 수신 대기 하도록 하려면 다음 명령이 필요합니다.

```
~]# semanage port -a -t squid_port_t -p tcp 10000
```

7. 다시 **squid** 를 시작하고 새 포트에서 수신 대기하도록 합니다.

```
~]# systemctl start squid.service
```

8. **Squid**가 비표준 포트(이 예에서는 **TCP 10000**)에서 수신 대기하도록 **SELinux**가 구성되었으므로 이 포트에서 성공적으로 시작됩니다.

19.2. 유형

고급 프로세스 격리를 제공하기 위해 SELinux 대상 정책에 사용되는 기본 권한 제어 방법은 Type Enforcement입니다. 모든 파일과 프로세스는 유형으로 레이블이 지정됩니다. type은 프로세스의 SELinux 도메인과 파일의 SELinux 유형을 정의합니다. SELinux 정책 규칙은 유형에 액세스하는 도메인이든 다른 도메인에 액세스하는 도메인이든 관계없이 유형이 서로 액세스하는 방법을 정의합니다. 액세스는 허용하는 특정 SELinux 정책 규칙이 있는 경우에만 허용됩니다.

다음 유형은 Squid와 함께 사용됩니다. 다양한 유형을 사용하여 유연한 액세스를 구성할 수 있습니다.

httpd_squid_script_exec_t

이 유형은 Squid 및 해당 구성에 대한 다양한 통계를 제공하는 cachemgr.cgi 와 같은 유틸리티에 사용됩니다.

squid_cache_t

/etc/squid/squid.conf 의 cache_dir 지시문에 정의된 대로 이 유형을 Squid에서 캐시하는 데이터에 사용합니다. 기본적으로 에 생성되거나 /var/cache/squid/ 및 /var/s pool/squid/ 디렉터리에 생성된 파일에 squid_cache_t 유형의 레이블이 지정됩니다. /var/squid Guard / 디렉터리에 생성된 squid 용 squidGuard URL 리디렉션기 플러그인의 파일도 squid_cache_t 유형으로 레이블이 지정됩니다. Squid는 캐시된 데이터에 대해 이 유형으로 레이블이 지정된 파일과 디렉토리만 사용할 수 있습니다.

squid_conf_t

이 유형은 Squid에서 구성에 사용하는 디렉터리 및 파일에 사용됩니다. 기존 파일 또는 /etc/squid/ 및 /usr/share/s quid/ 디렉토리에 생성되거나 생성된 파일은 오류 메시지 및 아이콘을 포함하여 이 유형으로 레이블이 지정됩니다.

squid_exec_t

이 유형은 squid 바이너리 /usr/sbin/squid에 사용됩니다.

squid_log_t

이 유형은 로그에 사용됩니다. 기존 파일 또는 /var/log/squid / 또는 /var/log/squid Guard/ 에 복사된 파일은 이 유형으로 레이블이 지정되어야 합니다.

squid_initrc_exec_t

이 유형은 /etc/rc.d/init.d/squid 에 있는 squid 를 시작하는 데 필요한 초기화 파일에 사용됩니다.

squid_var_run_t

이 유형은 **/var/run/** 디렉토리의 파일에서, 특히 **Squid**가 실행될 때 생성되는 **/var/run/squid.pid** 라는 프로세스 **ID(PID)**에서 사용합니다.

19.3. 부울

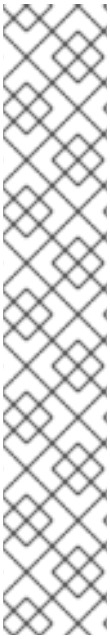
SELinux는 서비스를 실행하는 데 필요한 최소 액세스 수준을 기반으로 합니다. 서비스는 다양한 방법으로 실행할 수 있으므로 서비스 실행 방법을 지정해야 합니다. 다음 부울을 사용하여 **SELinux**를 설정합니다.

squid_connect_any

이 부울을 사용하면 **Squid**가 모든 포트의 원격 호스트에 대한 연결을 시작할 수 있습니다.

squid_use_tproxy

이 부울을 사용하면 **Squid**가 투명한 프록시로 실행될 수 있습니다.



참고

SELinux 정책의 지속적인 개발로 인해 위의 목록에 항상 서비스와 관련된 부울이 모두 포함되지 않을 수 있습니다. 나열하려면 다음 명령을 입력합니다.

```
~]$ getsebool -a | grep service_name
```

특정 부울에 대한 설명을 보려면 다음 명령을 입력합니다.

```
~]$ sepolicy booleans -b boolean_name
```

이 명령이 작동하려면 **sepolicy** 유틸리티를 제공하는 추가 **policycoreutils-devel** 패키지가 필요합니다.

19.4. 설정 예

19.4.1. 비표준 포트에 Squiid 연결

다음 예제에서는 **SELinux**가 위의 부울을 강제 적용하여 **Squid**를 보완하고 기본적으로 특정 포트에 대한 액세스만 허용하는 방법을 보여주는 실제 데모를 제공합니다. 그러면 이 예제에서 부울을 변경하고

해당 액세스가 허용됨을 보여 줍니다.

이 예제는 **SELinux**가 **Squid**의 간단한 설정에 어떤 영향을 줄 수 있는지를 보여줍니다. **Squid**에 대한 포괄적인 설명서는 이 문서의 범위를 벗어납니다. 자세한 내용은 공식 **Squid** 설명서 를 참조하십시오. 이 예제에서는 **Squid** 호스트에 인터넷 액세스라는 두 개의 네트워크 인터페이스가 있고, 방화벽이 **Squid**가 수신 대기하는 기본 **TCP** 포트를 사용하는 내부 인터페이스에서 액세스를 허용하도록 구성되었다고 가정합니다(**TCP 3128**).

1. **squid** 가 설치되었는지 확인합니다.

```
~]$ rpm -q squid
package squid is not installed
```

패키지가 설치되지 않은 경우 **yum** 유틸리티를 **root**로 사용하여 설치합니다.

```
~]# yum install squid
```

2. 기본 구성 파일 **/etc/squid/squid.conf** 를 편집하고 **cache_dir** 지시문이 주석 처리되지 않고 다음과 유사한지 확인합니다.

```
cache_dir ufs /var/spool/squid 100 16 256
```

이 행은 이 예에서 사용할 **cache_dir** 지시문의 기본 설정을 지정합니다. 이 지시문은 **Squid** 스토리지 형식(**ufs**),캐시가 상주하는시스템의디렉토리(각각**16** 및 **256**)로 구성됩니다.

3. 동일한 구성 파일에서 **http_access allow localnet** 지시문이 주석 처리되지 않았는지 확인합니다. 이를 통해 **Red Hat Enterprise Linux**에 **Squid**의 기본 설치에서 자동으로 구성된 **localnet ACL**의 트래픽을 허용합니다. 이 간단한 예는 기존 **RFC1918** 네트워크의 클라이언트 시스템이 프록시를 통해 액세스할 수 있도록 합니다.

4. 동일한 구성 파일에서 **visible_hostname** 지시문이 주석 처리되지 않고 시스템의 호스트 이름으로 구성되어 있는지 확인합니다. 값은 호스트의 **FQDN**(정규화된 도메인 이름)이어야 합니다.

```
visible_hostname squid.example.com
```

5. **root**로 다음 명령을 입력하여 **squid** 데몬을 시작합니다. **squid**가 처음 시작 되었으므로 이 명령은 **cache _dir** 지시문에서 위에 지정된 대로 캐시 디렉터리를 초기화한 다음 데몬을 시작합

니다.

```
~]# systemctl start squid.service
```

squid 가 성공적으로 시작되었는지 확인합니다. 출력에는 아래 정보가 포함되어 있으며 타임스탬프만 다릅니다.

```
~]# systemctl status squid.service
squid.service - Squid caching proxy
   Loaded: loaded (/usr/lib/systemd/system/squid.service; disabled)
   Active: active (running) since Thu 2014-02-06 15:00:24 CET; 6s ago
```

6.

squid _var_run_t 값에 표시된 대로 **squid** 프로세스 **ID(PID)**가 제한된 서비스로 시작되었는지 확인합니다.

```
~]# ls -lZ /var/run/squid.pid
-rw-r--r--. root squid unconfined_u:object_r:squid_var_run_t:s0 /var/run/squid.pid
```

7.

이때 이전에 구성된 **localnet ACL**에 연결된 클라이언트 시스템이 이 호스트의 내부 인터페이스를 프록시로 사용할 수 있습니다. 이 설정은 모든 일반 웹 브라우저 또는 시스템 전체의 설정에서 구성할 수 있습니다. 이제 **Squid**가 대상 시스템의 기본 포트**(TCP 3128)**에서 수신 대기하지만 대상 시스템은 공통 포트를 통해 인터넷의 다른 서비스에 대한 나가는 연결만 허용합니다. 이는 **SELinux** 자체에서 정의한 정책입니다. **SELinux**는 다음 단계에 표시된 대로 비표준 포트에 대한 액세스를 거부합니다.

8.

클라이언트가 **TCP** 포트 **10000**에서 수신 대기하는 웹 사이트와 같은 **Squid** 프록시를 통해 비표준 포트를 사용하여 요청하면 다음과 유사한 거부가 기록됩니다.

```
SELinux is preventing the squid daemon from connecting to network port 10000
```

9.

이러한 액세스를 허용하려면 기본적으로 비활성화되어 있으므로 **squid_connect_any** 부울을 수정해야 합니다.

```
~]# setsebool -P squid_connect_any on
```

참고

재부팅 시 **setsebool** 변경 사항이 지속되지 않도록 하려면 **-P** 옵션을 사용하지 마십시오.

10. ***Squid**가 이제 클라이언트를 대신하여 모든 포트에 대한 연결을 시작할 수 있으므로 클라이언트는 인터넷에서 비표준 포트에 액세스할 수 있습니다.*

---

[17]
*자세한 내용은 **Squid Caching Proxy** 프로젝트 페이지를 참조하십시오.*

20장. MARIADB(MYSQL 대체)

MariaDB 데이터베이스는 **MariaDB** 서버 데몬(**mysqld**)과 많은 클라이언트 프로그램 및 라이브러리로 구성된 다중 사용자, 다중 스레드 **SQL** 데이터베이스 서버입니다.[18]

Red Hat Enterprise Linux에서 **mariadb-server** 패키지는 **MariaDB**를 제공합니다. 다음 명령을 입력하여 **mariadb-server** 패키지가 설치되어 있는지 확인합니다.

```
~]$ rpm -q mariadb-server
package mariadb-server is not installed
```

설치되지 않은 경우 **yum** 유틸리티를 **root**로 사용하여 설치합니다.

```
~]# yum install mariadb-server
```

20.1. MARIADB 및 SELINUX

MariaDB가 활성화되면 기본적으로 제한된 상태로 실행됩니다. 제한된 프로세스는 자체 도메인에서 실행되며 다른 제한된 프로세스와 분리됩니다. **SELinux** 정책 구성에 따라 공격자가 제한된 프로세스가 손상되면 공격자가 리소스에 대한 액세스와 가능한 손상을 제한합니다. 다음 예제에서는 자체 도메인에서 실행되는 **MariaDB** 프로세스를 보여줍니다. 이 예제에서는 **mariadb-server** 패키지가 설치되어 있다고 가정합니다.

1. **getenforce** 명령을 실행하여 **SELinux**가 강제 모드로 실행 중인지 확인합니다.

   ```
   ~]$ getenforce
   Enforcing
   ```

   명령은 **SELinux**가 강제 모드에서 실행 중일 때 **Enforcing** (강제)을 반환합니다.

2. 다음 명령을 **root** 사용자로 입력하여 **mariadb** 를 시작합니다.

   ```
   ~]# systemctl start mariadb.service
   ```

   서비스가 실행 중인지 확인합니다. 출력에는 아래 정보가 포함되어야 합니다(시간 스탬프만 다릅니다).

```
~]# systemctl status mariadb.service
mariadb.service - MariaDB database server
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; disabled)
   Active: active (running) since Mon 2013-08-05 11:20:11 CEST; 3h 28min ago
```

3. 다음 명령을 입력하여 **mysqld** 프로세스를 확인합니다.

```
~]$ ps -eZ | grep mysqld
system_u:system_r:mysqld_safe_t:s0 12831 ?     00:00:00 mysqld_safe
system_u:system_r:mysqld_t:s0   13014 ?        00:00:00 mysqld
```

mysqld 프로세스와 연결된 **SELinux** 컨텍스트는 **system_u:system_r:mysqld_t:s0**입니다. 컨텍스트의 마지막 부분 **mysqld_t** 는 유형입니다. 유형은 프로세스 및 파일의 유형에 대한 도메인을 정의합니다. 이 경우 **mysqld** 프로세스는 **mysqld _t** 도메인에서 실행됩니다.

20.2. 유형

고급 프로세스 격리를 제공하기 위해 **SELinux** 대상 정책에 사용되는 기본 권한 제어 방법은 **Type Enforcement**입니다. 모든 파일과 프로세스는 유형으로 레이블이 지정됩니다. **type**은 프로세스의 **SELinux** 도메인과 파일의 **SELinux** 유형을 정의합니다. **SELinux** 정책 규칙은 유형에 액세스하는 도메인이든 다른 도메인에 액세스하는 도메인이든 관계없이 유형이 서로 액세스하는 방법을 정의합니다. 액세스는 허용하는 특정 **SELinux** 정책 규칙이 있는 경우에만 허용됩니다.

다음 유형은 **mysqld** 와 함께 사용됩니다. 다양한 유형을 사용하여 유연한 액세스를 구성할 수 있습니다.

mysqld_db_t

이 유형은 **MariaDB** 데이터베이스의 위치에 사용됩니다. **Red Hat Enterprise Linux**에서 데이터베이스의 기본 위치는 **/var/lib/mysql/** 디렉토리이지만 변경할 수 있습니다. **MariaDB** 데이터베이스의 위치가 변경되면 새 위치에 이 유형의 레이블이 지정되어야 합니다. 기본 데이터베이스 위치를 변경하는 방법과 새 섹션에 적절하게 레이블을 지정하는 방법에 대한 지침은 **20.4.1절. "MariaDB 데이터베이스 위치 변경"** 의 예제를 참조하십시오.

mysqld_etc_t

이 유형은 **MariaDB** 기본 구성 파일 **/etc/my.cnf** 및 **/etc/mysql/** 디렉터리에 있는 기타 구성 파일에 사용됩니다.

mysqld_exec_t

이 유형은 **Red Hat Enterprise Linux**의 **MariaDB** 바이너리의 기본 위치인 **/usr/libexec/mysqld** 에 있는 **mysqld** 바이너리에 사용됩니다. 다른 시스템은 이 유형의 레이블이 지정된 **/usr/sbin/mysqld**

에서 이 바이너리를 찾을 수 있습니다.

mysqld_unit_file_t

이 유형은 기본적으로 **Red Hat Enterprise Linux**에서 **/usr/lib/systemd/system/** 디렉터리에 있는 실행 가능한 **MariaDB** 관련 파일에 사용됩니다.

mysqld_log_t

적절한 작업을 위해서는 **MariaDB**의 로그에 이 유형으로 레이블이 지정되어야 합니다. **mysql.*** 와일드카드와 일치하는 **/var/log/** 디렉터리에 있는 모든 로그 파일에는 이 유형으로 레이블이 지정되어야 합니다.

mysqld_var_run_t

이 유형은 **/var/run/mariadb/** 디렉터리의 파일, 특히 실행 시 **mysqld** 데몬에 의해 생성되는 **/var/run/mariadb/mariadb.pid** 라는 프로세스 **ID(PID)**에서 사용합니다. 이 유형은 **/var/lib/mysql/mysql.sock** 과 같은 관련 소켓 파일에도 사용됩니다. 이러한 파일에는 제한된 서비스로 올바르게 작동하려면 이러한 파일에 레이블이 올바르게 지정되어야 합니다.

20.3. 부울

SELinux는 서비스를 실행하는 데 필요한 최소 액세스 수준을 기반으로 합니다. 서비스는 다양한 방법으로 실행할 수 있으므로 서비스 실행 방법을 지정해야 합니다. 다음 부울을 사용하여 **SELinux**를 설정합니다.

selinuxuser_mysql_connect_enabled

이 부울을 활성화하면 사용자가 로컬 **MariaDB** 서버에 연결할 수 있습니다.

exim_can_connect_db

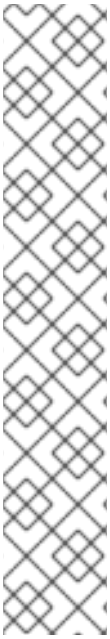
이 부울을 사용하면 **exim mailer** 가 데이터베이스 서버에 대한 연결을 시작할 수 있습니다.

ftpd_connect_db

이 부울을 사용하면 **ftp** 데몬이 데이터베이스 서버에 대한 연결을 시작할 수 있습니다.

httpd_can_network_connect_db

이 부울을 활성화하려면 웹 서버가 데이터베이스 서버와 통신하는 데 필요합니다.



참고

SELinux 정책의 지속적인 개발로 인해 위의 목록에 항상 서비스와 관련된 부울이 모두 포함되지 않을 수 있습니다. 나열하려면 다음 명령을 입력합니다.

```
~]$ getsebool -a | grep service_name
```

특정 부울에 대한 설명을 보려면 다음 명령을 입력합니다.

```
~]$ sepolicy booleans -b boolean_name
```

이 명령이 작동하려면 **sepolicy** 유틸리티를 제공하는 추가 **policycoreutils-devel** 패키지가 필요합니다.

20.4. 설정 예

20.4.1. MariaDB 데이터베이스 위치 변경

Red Hat Enterprise Linux를 사용하는 경우 데이터베이스를 저장할 **MariaDB**의 기본 위치는 **/var/lib/mysql/** 입니다. 여기에서 **SELinux**는 기본적으로 이를 예상하므로 이 영역에는 **mysqld_db_t** 유형을 사용하여 이미 적절하게 레이블이 지정됩니다.

데이터베이스가 저장되는 위치는 개별 환경 요구 사항 또는 기본 설정에 따라 변경할 수 있지만 **SELinux**는 이 새 위치를 인식하는 것이 중요합니다. 이 레이블은 적절하게 레이블이 지정됩니다. 이 예제에서는 **MariaDB** 데이터베이스의 위치를 변경한 다음 **SELinux**가 해당 콘텐츠를 기반으로 새 영역에 계속 보호 메커니즘을 제공할 수 있도록 새 위치에 레이블을 지정하는 방법을 설명합니다.

이 예제는 **SELinux**가 **MariaDB**에 미치는 영향을 보여줍니다. **MariaDB**에 대한 포괄적인 설명서는 이 문서의 범위를 벗어납니다. 자세한 내용은 공식 **MariaDB** 설명서 를 참조하십시오. 이 예제에서는 **mariadb-server** 및 **setroubleshoot-server** 패키지가 설치되어 있고, **auditd** 서비스가 실행 중이며, **/var/lib/mysql/** 의 기본 위치에 유효한 데이터베이스가 설치되어 있다고 가정합니다.

1. **mysql** 의 기본 데이터베이스 위치의 **SELinux** 컨텍스트 보기 :

```
~]# ls -lZ /var/lib/mysql
drwx------. mysql mysql system_u:object_r:mysqld_db_t:s0 mysql
```

그러면 데이터베이스 파일의 위치의 기본 컨텍스트 요소인 **mysqld_db_t** 가 표시됩니다. 이 컨텍스트가 제대로 작동하려면 이 예제에서 사용할 새 데이터베이스 위치에 수동으로 적용해야 합니다.

2. 다음 명령을 입력하고 **mysqld** 루트 암호를 입력하여 사용 가능한 데이터베이스를 표시합니다.

```
~]# mysqlshow -u root -p
Enter password: *******
+--------------------+
|     Databases      |
+--------------------+
| information_schema |
| mysql              |
| test               |
| wikidb             |
+--------------------+
```

3. **mariadb.service** 서비스를 중지합니다.

```
~]# systemctl stop mariadb.service
```

4. 데이터베이스의 새 위치에 대한 새 디렉터리를 만듭니다. 이 예에서는 **/mysql/** 가 사용됩니다.

```
~]# mkdir -p /mysql
```

5. 이전 위치의 데이터베이스 파일을 새 위치로 복사합니다.

```
~]# cp -R /var/lib/mysql/* /mysql/
```

6. **mysql** 사용자 및 그룹의 액세스를 허용하도록 이 위치의 소유권을 변경합니다. 이렇게 하면 **SELinux**가 계속 관찰할 기존 **Unix** 권한이 설정됩니다.

```
~]# chown -R mysql:mysql /mysql
```

7. 다음 명령을 입력하여 새 디렉터리의 초기 컨텍스트를 확인합니다.

```
~]# ls -lZ /mysql
drwxr-xr-x. mysql mysql unconfined_u:object_r:usr_t:s0   mysql
```

*새로 생성된 이 디렉터리의 컨텍스트 **usr_t** 는 현재 **MariaDB** 데이터베이스 파일의 위치로 **SELinux**에 적합하지 않습니다. 컨텍스트가 변경되면 **MariaDB**가 이 영역에서 제대로 작동할 수 있습니다.*

8. *텍스트 편집기를 사용하여 기본 **MariaDB** 구성 파일 **/etc/my.cnf** 를 열고 새 위치를 참조하도록 **datadir** 옵션을 수정합니다. 이 예에서 입력해야 하는 값은 **/mysql:**입니다.*

```
[mysqld]
datadir=/mysql
```

이 파일을 저장하고 종료합니다.

9. ***mariadb.service** 를 시작합니다. 서비스가 시작되지 않으면 거부 메시지가 **/var/log/messages** 파일에 기록됩니다.*

```
~]# systemctl start mariadb.service
Job for mariadb.service failed. See 'systemctl status mariadb.service' and 'journalctl -xn' for
details.
```

*그러나 감사 데몬이 **setroubleshoot** 서비스와 함께 실행 중인 경우 거부는 대신 **/var/log/audit/audit.log** 파일에 기록됩니다.*

```
SELinux is preventing /usr/libexec/mysqld "write" access on /mysql. For complete SELinux
messages. run sealert -l b3f01aff-7fa6-4ebe-ad46-abaef6f8ad71
```

*이러한 거부의 원인은 **/mysql/** 가 **MariaDB** 데이터 파일에 대해 올바르게 레이블이 지정되지 않기 때문입니다. **SELinux**는 **MariaDB**가 **usr_t** 로 레이블이 지정된 콘텐츠에 액세스하지 못하게 합니다. 이 문제를 해결하려면 다음 단계를 수행하십시오.*

10. *다음 명령을 입력하여 **/mysql/** 에 대한 컨텍스트 매핑을 추가합니다. **semanage** 유틸리티는 기본적으로 설치되지 않습니다. 시스템에 누락된 경우 **policycoreutils-python** 패키지를 설치합니다.*

```
~]# semanage fcontext -a -t mysqld_db_t "/mysql(/.*)?"
```

11. *이 매핑은 **/etc/selinux/targeted/contexts/files/file_contexts.local** 파일에 작성됩니다.*

```
~]# grep -i mysql /etc/selinux/targeted/contexts/files/file_contexts.local

/mysql(/.*)?    system_u:object_r:mysqld_db_t:s0
```

12. *이제 **restorecon** 유틸리티를 사용하여 이 컨텍스트 매핑을 실행 중인 시스템에 적용합니다.*

```
~]# restorecon -R -v /mysql
```

13. *이제 **/mysql/** 위치에 **MariaDB**에 대한 올바른 컨텍스트가 지정되어 **mysqld** 가 시작됩니다.*

```
~]# systemctl start mariadb.service
```

14. ***/mysql/** 에 대한 컨텍스트가 변경되었는지 확인합니다.*

```
~]$ ls -lZ /mysql
drwxr-xr-x. mysql mysql system_u:object_r:mysqld_db_t:s0 mysql
```

15. *위치가 변경 및 레이블이 지정되었으며 **mysqld** 가 성공적으로 시작되었습니다. 이제 정상적인 작업을 확인하기 위해 실행 중인 모든 서비스를 테스트해야 합니다.*

---

[18]
*자세한 내용은 **MariaDB** 프로젝트 페이지를 참조하십시오.*

21장. POSTGRESQL

PostgreSQL은 DBMS(Object-Relational Database Management System)입니다.[19]

Red Hat Enterprise Linux에서 postgresql-server 패키지는 PostgreSQL을 제공합니다. 다음 명령을 입력하여 postgresql-server 패키지가 설치되어 있는지 확인합니다.

```
~]# rpm -q postgresql-server
```

설치되지 않은 경우 yum 유틸리티를 root로 사용하여 설치합니다.

```
~]# yum install postgresql-server
```

21.1. POSTGRESQL 및 SELINUX

PostgreSQL이 활성화되면 기본적으로 제한된 상태로 실행됩니다. 제한된 프로세스는 자체 도메인에서 실행되며 다른 제한된 프로세스와 분리됩니다. SELinux 정책 구성에 따라 공격자가 제한된 프로세스가 손상되면 공격자가 리소스에 대한 액세스와 가능한 손상을 제한합니다. 다음 예제에서는 자체 도메인에서 실행되는 PostgreSQL 프로세스를 보여줍니다. 이 예제에서는 postgresql-server 패키지가 설치되어 있다고 가정합니다.

1. **getenforce 명령을 실행하여 SELinux가 강제 모드로 실행 중인지 확인합니다.**

   ```
   ~]$ getenforce
   Enforcing
   ```

   명령은 SELinux가 강제 모드에서 실행 중일 때 Enforcing (강제)을 반환합니다.

2. **다음 명령을 root 사용자로 입력하여 postgresql 을 시작합니다.**

   ```
   ~]# systemctl start postgresql.service
   ```

   서비스가 실행 중인지 확인합니다. 출력에는 아래 정보가 포함되어야 합니다(시간 스탬프만 다릅니다).

   ```
   ~]# systemctl start postgresql.service
   postgresql.service - PostgreSQL database server
   ```

```
   Loaded: loaded (/usr/lib/systemd/system/postgresql.service; disabled)
   Active: active (running) since Mon 2013-08-05 14:57:49 CEST; 12s
```

3. 다음 명령을 입력하여 **postgresql** 프로세스를 확인합니다.

```
~]$ ps -eZ | grep postgres
system_u:system_r:postgresql_t:s0 395 ?    00:00:00 postmaster
system_u:system_r:postgresql_t:s0 397 ?    00:00:00 postmaster
system_u:system_r:postgresql_t:s0 399 ?    00:00:00 postmaster
system_u:system_r:postgresql_t:s0 400 ?    00:00:00 postmaster
system_u:system_r:postgresql_t:s0 401 ?    00:00:00 postmaster
system_u:system_r:postgresql_t:s0 402 ?    00:00:00 postmaster
```

postgresql 프로세스와 연결된 **SELinux** 컨텍스트는 **system_u:system_r:postgresql_t:s0** 입니다. 컨텍스트의 두 번째 마지막 부분인 **postgresql_t** 는 유형입니다. 유형은 프로세스 및 파일의 유형에 대한 도메인을 정의합니다. 이 경우 **postgresql** 프로세스는 **postgresql_t** 도메인에서 실행됩니다.

21.2. 유형

고급 프로세스 격리를 제공하기 위해 **SELinux** 대상 정책에 사용되는 기본 권한 제어 방법은 **Type Enforcement**입니다. 모든 파일과 프로세스는 유형으로 레이블이 지정됩니다. **type**은 프로세스의 **SELinux** 도메인과 파일의 **SELinux** 유형을 정의합니다. **SELinux** 정책 규칙은 유형에 액세스하는 도메인이든 다른 도메인에 액세스하는 도메인이든 관계없이 유형이 서로 액세스하는 방법을 정의합니다. 액세스는 허용하는 특정 **SELinux** 정책 규칙이 있는 경우에만 허용됩니다.

다음 유형은 **postgresql** 과 함께 사용됩니다. 다양한 유형을 사용하여 유연한 액세스를 구성할 수 있습니다. 아래 목록에서는 사용 가능한 전체 위치와 일치하는 몇 가지 정규 표현식을 사용합니다.

postgresql_db_t

이 유형은 여러 위치에 사용됩니다. 이 유형으로 레이블이 지정된 위치는 **PostgreSQL**의 데이터 파일에 사용됩니다.

- **/usr/lib/pgsql/test/regres**
- **/usr/share/jonas/pgsql**
- **/var/lib/pgsql/data**

- */var/lib/postgres(ql)?*

postgresql_etc_t

*이 유형은 **/etc/postgresql/** 디렉터리의 구성 파일에 사용됩니다.*

postgresql_exec_t

*이 유형은 여러 위치에 사용됩니다. 이 유형으로 레이블이 지정된 위치는 **PostgreSQL**의 바이너리에 사용됩니다.*

- */usr/bin/initdb(.sepgsql)?*
- */usr/bin/(se)?postgres*
- */usr/lib(64)?/postgresql/bin/.**
- */usr/lib(64)?/pgsql/test/regress/pg_regress*

systemd_unit_file_t

*이 유형은 **/usr/lib/systemd/system/** 디렉터리에 있는 실행 가능한 **PostgreSQL** 관련 파일에 사용됩니다.*

postgresql_log_t

이 유형은 여러 위치에 사용됩니다. 이 유형으로 레이블이 지정된 위치는 로그 파일에 사용됩니다.

- */var/lib/pgsql/logfile*

- /var/lib/pgsql/pgstartup.log
- /var/lib/sepgsql/pgstartup.log
- /var/log/postgresql
- /var/log/postgres.log.*
- /var/log/rhdb/rhdb
- /var/log/sepostgresql.log.*

postgresql_var_run_t

이 유형은 /var/run/postgresql/ 디렉터리의 PID(프로세스 ID)와 같은 PostgreSQL의 런타임 파일에 사용됩니다.

21.3. 부울

SELinux는 서비스를 실행하는 데 필요한 최소 액세스 수준을 기반으로 합니다. 서비스는 다양한 방법으로 실행할 수 있으므로 서비스 실행 방법을 지정해야 합니다. 다음 부울을 사용하여 SELinux를 설정합니다.

selinuxuser_postgresql_connect_enabled

이 부울을 활성화하면 PostgreSQL에서 정의한 모든 사용자 도메인이 데이터베이스 서버에 연결할 수 있습니다.

참고

SELinux 정책의 지속적인 개발로 인해 위의 목록에 항상 서비스와 관련된 부울이 모두 포함되지 않을 수 있습니다. 나열하려면 다음 명령을 입력합니다.

```
~]$ getsebool -a | grep service_name
```

특정 부울에 대한 설명을 보려면 다음 명령을 입력합니다.

```
~]$ sepolicy booleans -b boolean_name
```

이 명령이 작동하려면 **sepolicy** 유틸리티를 제공하는 추가 **policycoreutils-devel** 패키지가 필요합니다.

21.4. 설정 예

21.4.1. PostgreSQL 데이터베이스 위치 변경

Red Hat Enterprise Linux를 사용하는 경우 데이터베이스를 저장할 **PostgreSQL**의 기본 위치는 **/var/lib/pgsql/data/** 입니다. 여기에서 **SELinux**는 기본적으로 이를 예상하므로 이 영역에는 **postgresql_db_t** 유형을 사용하여 이미 적절하게 레이블이 지정됩니다.

데이터베이스가 있는 영역은 개별 환경 요구 사항 또는 기본 설정에 따라 변경할 수 있지만 **SELinux**는 이 새 위치를 인식하는 것이 중요합니다. 이 영역에 따라 레이블이 지정되어야 합니다. 이 예에서는 **PostgreSQL** 데이터베이스의 위치를 변경한 다음 **SELinux**가 해당 콘텐츠를 기반으로 새 영역에 계속 보호 메커니즘을 제공할 수 있도록 새 위치에 레이블을 지정하는 방법을 설명합니다.

이 예제는 **SELinux**가 **PostgreSQL**에 미치는 영향을 보여줍니다. **PostgreSQL**의 포괄적인 설명서는 이 문서의 범위를 벗어납니다. 자세한 내용은 공식 **PostgreSQL** 설명서 를 참조하십시오. 이 예제에서는 **postgresql-server** 패키지가 설치되어 있다고 가정합니다.

1. **postgresql** 의 기본 데이터베이스 위치의 **SELinux** 컨텍스트를 확인합니다.

```
~]# ls -lZ /var/lib/pgsql
drwx------. postgres postgres system_u:object_r:postgresql_db_t:s0 data
```

데이터베이스 파일 위치의 기본 컨텍스트 요소인 **postgresql_db_t** 가 표시됩니다. 이 컨텍스트가 제대로 작동하려면 이 예제에서 사용할 새 데이터베이스 위치에 수동으로 적용해야 합니다.

2. 데이터베이스의 새 위치에 대한 새 디렉터리를 만듭니다. 이 예에서는 **/opt/postgresql/data/** 가 사용됩니다. 다른 위치를 사용하는 경우 다음 단계의 텍스트를 해당 위치로 바꿉니다.

```
~]# mkdir -p /opt/postgresql/data
```

3. 새 위치의 디렉터리 목록을 수행합니다. 새 디렉터리의 초기 컨텍스트는 **usr_t** 입니다. 이 컨텍스트는 **SELinux**가 **PostgreSQL**에 보호 메커니즘을 제공하는 데 충분하지 않습니다. 컨텍스트가 변경되면 새 영역에서 제대로 작동할 수 있습니다.

```
~]# ls -lZ /opt/postgresql/
drwxr-xr-x. root root unconfined_u:object_r:usr_t:s0   data
```

4. **postgres** 사용자 및 그룹의 액세스를 허용하도록 새 위치의 소유권을 변경합니다. 이렇게 하면 **SELinux**가 계속 관찰할 기존의 **Unix** 권한이 설정됩니다.

```
~]# chown -R postgres:postgres /opt/postgresql
```

5. 텍스트 편집기를 사용하여 **/etc/systemd/system/postgresql.service** 파일을 열고 **PGDATA** 및 **PGLOG** 변수를 수정하여 새 위치를 가리킵니다.

```
~]# vi /etc/systemd/system/postgresql.service
PGDATA=/opt/postgresql/data
PGLOG=/opt/postgresql/data/pgstartup.log
```

이 파일을 저장하고 텍스트 편집기를 종료합니다.

/etc/systemd/system/postgresql.service 파일이 없는 경우 파일을 생성하고 다음 콘텐츠를 삽입합니다.

```
.include /lib/systemd/system/postgresql.service
[Service]

# Location of database directory
Environment=PGDATA=/opt/postgresql/data
Environment=PGLOG=/opt/postgresql/data/pgstartup.log
```

6. 새 위치에서 데이터베이스를 초기화합니다.

```
~]$ su - postgres -c "initdb -D /opt/postgresql/data"
```

7. 데이터베이스 위치를 변경하면 이 시점에 서비스 시작이 실패합니다.

```
~]# systemctl start postgresql.service
Job for postgresql.service failed. See 'systemctl status postgresql.service' and 'journalctl -xn'
for details.
```

SELinux로 인해 서비스가 시작되지 않았습니다. 새 위치의 레이블이 올바르지 않기 때문입니다. 다음 단계에서는 새 위치(**/opt/postgresql/**)에 레이블을 지정하고 **postgresql** 서비스를 제대로 시작하는 방법을 설명합니다.

8. **semanage** 유틸리티를 사용하여 **/opt/postgresql/** 및 그 안에 있는 기타 디렉터리/ 파일에 대한 컨텍스트 매핑을 추가합니다.

```
~]# semanage fcontext -a -t postgresql_db_t "/opt/postgresql(/.*)?"
```

9. 이 매핑은 **/etc/selinux/targeted/contexts/files/file_contexts.local** 파일에 작성됩니다.

```
~]# grep -i postgresql /etc/selinux/targeted/contexts/files/file_contexts.local

/opt/postgresql(/.*)?    system_u:object_r:postgresql_db_t:s0
```

10. 이제 **restorecon** 유틸리티를 사용하여 이 컨텍스트 매핑을 실행 중인 시스템에 적용합니다.

```
~]# restorecon -R -v /opt/postgresql
```

11. 이제 **/opt/postgresql/** 위치에 **PostgreSQL**에 대한 올바른 컨텍스트로 레이블이 지정되었으므로 **postgresql** 서비스가 성공적으로 시작됩니다.

```
~]# systemctl start postgresql.service
```

12. **/opt/postgresql/** 에 대한 컨텍스트가 올바른지 확인합니다.

```
~]$ ls -lZ /opt
drwxr-xr-x. root root system_u:object_r:postgresql_db_t:s0 postgresql
```

13. ***postgresql** 프로세스가 새 위치를 표시하는 **ps** 명령으로 확인합니다.*

```
~]# ps aux | grep -i postmaster

postgres 21564  0.3  0.3  42308  4032 ?        S    10:13   0:00 /usr/bin/postmaster -p 5432 -D
/opt/postgresql/data/
```

14. *위치가 변경 및 레이블이 지정되었으며 **postgresql** 이 성공적으로 시작되었습니다. 이제 정상적인 작업을 확인하기 위해 실행 중인 모든 서비스를 테스트해야 합니다.*

---

[19]
*자세한 내용은 **PostgreSQL** 프로젝트 페이지를 참조하십시오.*

22장. RSYNC

rsync 유틸리티는 빠른 파일 전송을 수행하고 시스템 간에 데이터 동기화에 사용됩니다. [20]

Red Hat Enterprise Linux를 사용하는 경우 **rsync** 패키지는 **rsync** 를 제공합니다. 다음 명령을 입력하여 **rsync** 패키지가 설치되어 있는지 확인합니다.

```
~]$ rpm -q rsync
package rsync is not installed
```

설치되지 않은 경우 **yum** 유틸리티를 **root**로 사용하여 설치합니다.

```
~]# yum install rsync
```

22.1. RSYNC 및 SELINUX

SELinux에는 파일 유형을 정의하기 위해 파일에 확장된 특성이 있어야 합니다. 정책은 이러한 파일에 필요한 액세스 데몬을 제어합니다. **rsync** 데몬을 사용하여 파일을 공유하려면 파일 및 디렉터리에 **public_content_t** 유형의 레이블을 지정해야 합니다. **SELinux**가 **rsync** 를 통해 보호 메커니즘을 수행하려면 대부분의 서비스와 마찬가지로 올바른 레이블 지정이 필요합니다.[21]

22.2. 유형

고급 프로세스 격리를 제공하기 위해 **SELinux** 대상 정책에 사용되는 기본 권한 제어 방법은 **Type Enforcement**입니다. 모든 파일과 프로세스는 유형으로 레이블이 지정됩니다. **type**은 프로세스의 **SELinux** 도메인과 파일의 **SELinux** 유형을 정의합니다. **SELinux** 정책 규칙은 유형에 액세스하는 도메인이든 다른 도메인에 액세스하는 도메인이든 관계없이 유형이 서로 액세스하는 방법을 정의합니다. 액세스는 허용하는 특정 **SELinux** 정책 규칙이 있는 경우에만 허용됩니다.

다음 유형은 **rsync** 와 함께 사용됩니다. 유연한 액세스를 구성할 수 있는 다양한 유형:

public_content_t

rsync 를 사용하여 공유할 파일 위치(및 실제 파일)에 사용되는 일반적인 유형입니다. **rsync** 와 공유할 파일을 배치하기 위해 특수 디렉토리를 만드는 경우 디렉토리 및 해당 콘텐츠에 이 레이블이 적용되어야 합니다.

rsync_exec_t

이 유형은 /usr/bin/rsync 시스템 바이너리에 사용됩니다.

rsync_log_t

이 유형은 기본적으로 /var/log/rsync.log 에 있는 rsync 로그 파일에 사용됩니다. rsync 로그의 위치를 로 변경하려면 런타임 시 rsync 명령에 --log-file=FILE 옵션을 사용합니다.

rsync_var_run_t

이 유형은 /var/run/rsync d.lock에 있는 rsync d 잠금 파일에 사용됩니다. 이 잠금 파일은 rsync 서버에서 연결 제한을 관리하는 데 사용됩니다.

rsync_data_t

이 유형은 rsync 도메인으로 사용하려는 파일 및 디렉터리에 사용되며 다른 서비스의 액세스 범위에서 분리합니다. 또한 public_content_t 는 파일 또는 디렉터리가 여러 서비스(예: rsync 도메인으로 FTP 및 NFS 디렉터리)와 상호 작용할 때 사용할 수 있는 일반적인 SELinux 컨텍스트 유형입니다.

rsync_etc_t

이 유형은 /etc 디렉토리의 rsync 관련 파일에 사용됩니다.

22.3. 부울

SELinux는 서비스를 실행하는 데 필요한 최소 액세스 수준을 기반으로 합니다. 서비스는 다양한 방법으로 실행할 수 있으므로 서비스 실행 방법을 지정해야 합니다. 다음 부울을 사용하여 SELinux를 설정합니다.

rsync_anon_write

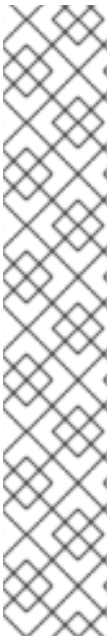
이 부울을 활성화하면 rsync _t 도메인의 rsync가 public_ content_rw_t 유형의 파일, 링크 및 디렉토리를 관리할 수 있습니다. 공용 파일 전송 서비스에 사용되는 공용 파일인 경우가 많습니다. 파일과 디렉터리에는 이 유형의 레이블이 지정되어야 합니다.

rsync_client

이 부울을 활성화하면 rsync 가 rsync_port_t 로 정의된 포트에 대한 연결을 시작하고 데몬이 rsync_data_t 유형의 파일, 링크 및 디렉토리를 관리할 수 있습니다. SELinux가 제어 권한을 적용하려면 rsync_t 도메인에 rsync_t 도메인이 있어야 합니다. 이 장의 구성 예제에서는 rsync _t 도메인에서 실행되는 rsync 를 보여줍니다.

rsync_export_all_ro

이 부울을 활성화하면 **rsync _t** 도메인의 **rsync** 가 클라이언트에 대한 읽기 전용 액세스 권한이 있는 **NFS** 및 **CIFS** 볼륨을 내보낼 수 있습니다.



참고

SELinux 정책의 지속적인 개발로 인해 위의 목록에 항상 서비스와 관련된 부울이 모두 포함되지 않을 수 있습니다. 나열하려면 다음 명령을 입력합니다.

```
~]$ getsebool -a | grep service_name
```

특정 부울에 대한 설명을 보려면 다음 명령을 입력합니다.

```
~]$ sepolicy booleans -b boolean_name
```

이 명령이 작동하려면 **sepolicy** 유틸리티를 제공하는 추가 **policycoreutils-devel** 패키지가 필요합니다.

22.4. 설정 예

22.4.1. 데몬으로 rsync

Red Hat Enterprise Linux를 사용하는 경우 **rsync**를 데몬으로 사용하여 여러 클라이언트가 중앙 집중식 파일을 보관하고 동기화된 상태로 유지하도록 중앙 집중식 서버로 직접 통신할 수 있습니다. 다음 예제에서는 올바른 도메인에서 네트워크 소켓을 통해 **rsync**를 데몬으로 실행하는 방법과 **SELinux**에서 이 데몬이 사전 정의된(**SELinux** 정책) **TCP** 포트에서 실행될 것으로 예상하는 방법을 보여줍니다. 이 예제에서는 **rsync** 데몬이 비표준 포트에서 정상적으로 실행되도록 **SELinux** 정책을 수정하는 방법을 보여줍니다.

이 예제는 단일 시스템에서 **SELinux** 정책 및 로컬 데몬 및 프로세스에 대한 제어 권한을 시연하기 위해 수행됩니다. 이 예제는 **SELinux**가 **rsync**에 어떤 영향을 미칠 수 있는지를 보여줍니다. **rsync**의 포괄적인 설명서는 이 문서의 범위를 벗어납니다. 자세한 내용은 공식 **rsync** 설명서 를 참조하십시오. 이 예제에서는 **rsync,setroubleshoot-server** 및 **audit** 패키지가 설치되어 있다고 가정하고, **SELinux** 대상 정책이 사용되며 **SELinux**가 강제 모드로 실행 중이라고 가정합니다.

절차 **22.1. rsync**가 **rsync**로 시작됨_t

1. **getenforce** 명령을 실행하여 **SELinux**가 강제 모드로 실행 중인지 확인합니다.

```
~]$ getenforce
Enforcing
```

명령은 **SELinux**가 강제 모드에서 실행 중일 때 **Enforcing** (강제)을 반환합니다.

2. **rsync** 바이너리가 시스템 경로에 있는지 확인하려면 **which** 명령을 실행합니다.

```
~]$ which rsync
/usr/bin/rsync
```

3. 데몬으로 **rsync**를 실행하는 경우 구성 파일을 사용하고 **/etc/rsyncd.conf** 로 저장해야 합니다. 이 예에서 사용된 다음 구성 파일은 매우 간단하며 사용 가능한 모든 옵션을 나타내는 것은 아니며 **rsync** 데몬을 시연하기에 충분합니다.

```
log file = /var/log/rsync.log
pid file = /var/run/rsyncd.pid
lock file = /var/run/rsync.lock
[files]
 path = /srv/rsync
        comment = file area
        read only = false
        timeout = 300
```

4. 이제 **rsync**가 데몬 모드에서 작동하도록 간단한 구성 파일이 있으므로 다음 명령을 실행하여 시작할 수 있습니다.

```
~]# systemctl start rsyncd.service
```

rsyncd 가 성공적으로 시작되었는지 확인합니다(출력은 아래와 유사하게 표시되며 타임스탬프만 다릅니다).

```
~]# systemctl status rsyncd.service
rsyncd.service - fast remote file copy program daemon
   Loaded: loaded (/usr/lib/systemd/system/rsyncd.service; disabled)
   Active: active (running) since Thu 2014-02-27 09:46:24 CET; 2s ago
 Main PID: 3220 (rsync)
   CGroup: /system.slice/rsyncd.service
           └─3220 /usr/bin/rsync --daemon --no-detach
```

이제 **SELinux**는 **rsync _t** 도메인에서 실행 중이므로 **rsync** 데몬을 통해 보호 메커니즘을 적용할 수 있습니다.

```
~]$ ps -eZ | grep rsync
system_u:system_r:rsync_t:s0     3220 ?        00:00:00 rsync
```

*이 예제에서는 **rsync _t** 도메인에서 **rsyncd** 를 실행하는 방법을 보여줍니다. **rsync**는 소켓이 활성화된 서비스로도 실행할 수 있습니다. 이 경우 클라이언트가 서비스에 연결을 시도할 때까지 **rsyncd** 가 실행되지 않습니다. **rsyncd** 가 **socket-activated** 서비스로 실행되도록 하려면 위의 단계를 따르십시오. **rsyncd**를 **socket-activated** 서비스로 시작하려면 다음 명령을 루트로 입력합니다.*

```
~]# systemctl start rsyncd.socket
```

*다음 예제에서는 이 데몬이 기본이 아닌 포트에서 성공적으로 실행되도록 하는 방법을 보여줍니다. 다음 예제에서는 **TCP** 포트 **10000**이 사용됩니다.*

*절차 **22.2.** 기본이 아닌 포트에서 **rsync** 데몬 실행*

1. *`/etc/rsyncd.conf` 파일을 수정하고 글로벌 구성 영역(즉, 파일 영역을 정의하기 전에)의 파일 맨 위에 **port = 10000** 행을 추가합니다. 새 구성 파일은 다음과 같이 나타납니다.*

```
log file = /var/log/rsyncd.log
pid file = /var/run/rsyncd.pid
lock file = /var/run/rsync.lock
port = 10000
[files]
        path = /srv/rsync
        comment = file area
        read only = false
 timeout = 300
```

2. *이 새로운 설정으로 **rsync** 데몬을 시작한 후 다음과 유사한 거부 메시지가 **SELinux**에서 로깅합니다.*

```
Jul 22 10:46:59 localhost setroubleshoot: SELinux is preventing the rsync (rsync_t) from binding to port 10000. For complete SELinux messages, run sealert -l c371ab34-639e-45ae-9e42-18855b5c2de8
```

3. *`semanage` 유틸리티를 사용하여 **rsync_port_t** 의 **SELinux** 정책에 **TCP** 포트 **10000**을 추가합니다.*

```
~]# semanage port -a -t rsync_port_t -p tcp 10000
```

4. *이제 **rsync_port_t** 의 **SELinux** 정책에 **TCP** 포트 **10000**이 추가되었으므로**rsyncd** 가 이 포트에서 정상적으로 시작되고 작동합니다.*

```
~]# systemctl start rsyncd.service
```

```
~]# netstat -lnp | grep 10000
tcp        0      0 0.0.0.0:10000   0.0.0.0:*       LISTEN      9910/rsync
```

SELinux는 정책을 수정했으며 이제 rsyncd 가 TCP 포트 10000에서 작동할 수 있게 되었습니다.

---

[20]
자세한 내용은 Rsync 프로젝트 페이지를 참조하십시오.

[21]
rsync 및 SELinux에 대한 자세한 내용은 rsync_selinux(8) 도움말 페이지를 참조하십시오.

23장. POSTFIX

Postfix는 LDAP, SMTP AUTH(SASL) 및 TLS와 같은 프로토콜을 지원하는 오픈 소스MTA(메일 전송 에이전트)입니다.[22]

Red Hat Enterprise Linux에서 postfix 패키지는 Postfix를 제공합니다. 다음 명령을 입력하여 postfix 패키지가 설치되어 있는지 확인합니다.

```
~]$ rpm -q postfix
package postfix is not installed
```

설치되지 않은 경우 yum utility root를 사용하여 설치합니다.

```
~]# yum install postfix
```

23.1. POSTFIX 및 SELINUX

Postfix가 활성화되면 기본적으로 제한된 상태로 실행됩니다. 제한된 프로세스는 자체 도메인에서 실행되며 다른 제한된 프로세스와 분리됩니다. SELinux 정책 구성에 따라 공격자가 제한된 프로세스가 손상되면 공격자가 리소스에 대한 액세스와 가능한 손상을 제한합니다. 다음 예제에서는 자체 도메인에서 실행되는 Postfix 및 관련 프로세스를 보여줍니다. 이 예제에서는 postfix 패키지가 설치되어 있고 Postfix 서비스가 시작되었다고 가정합니다.

1. getenforce 명령을 실행하여 SELinux가 강제 모드로 실행 중인지 확인합니다.

```
~]$ getenforce
Enforcing
```

명령은 SELinux가 강제 모드에서 실행 중일 때 Enforcing (강제)을 반환합니다.

2. root 사용자로 다음 명령을 입력하여 postfix 를 시작합니다.

```
~]# systemctl start postfix.service
```

서비스가 실행 중인지 확인합니다. 출력에는 아래 정보가 포함되어야 합니다(시간 스탬프만 다릅니다).

```
~]# systemctl status postfix.service
postfix.service - Postfix Mail Transport Agent
   Loaded: loaded (/usr/lib/systemd/system/postfix.service; disabled)
   Active: active (running) since Mon 2013-08-05 11:38:48 CEST; 3h 25min ago
```

3. 다음 명령을 실행하여 **postfix** 프로세스를 확인합니다.

```
~]$ ps -eZ | grep postfix
system_u:system_r:postfix_master_t:s0 1651 ?   00:00:00 master
system_u:system_r:postfix_pickup_t:s0 1662 ?   00:00:00 pickup
system_u:system_r:postfix_qmgr_t:s0 1663 ?     00:00:00 qmgr
```

위의 출력에서 **Postfix** 마스터 프로세스와 연결된 **SELinux** 컨텍스트는 **system_u:system_r:postfix_master_t:s0**입니다. 컨텍스트의 마지막 부분인 **postfix_master_t**는 이 프로세스의 유형입니다. 유형은 프로세스 및 파일의 유형에 대한 도메인을 정의합니다. 이 경우 마스터 프로세스는 **postfix_master_t** 도메인에서 실행됩니다.

23.2. 유형

고급 프로세스 격리를 제공하기 위해 **SELinux** 대상 정책에 사용되는 기본 권한 제어 방법은 **Type Enforcement**입니다. 모든 파일과 프로세스는 유형으로 레이블이 지정됩니다. **type**은 프로세스의 **SELinux** 도메인과 파일의 **SELinux** 유형을 정의합니다. **SELinux** 정책 규칙은 유형에 액세스하는 도메인이든 다른 도메인에 액세스하는 도메인이든 관계없이 유형이 서로 액세스하는 방법을 정의합니다. 액세스는 허용하는 특정 **SELinux** 정책 규칙이 있는 경우에만 허용됩니다.

다음 유형은 **Postfix**와 함께 사용됩니다. 유연한 액세스를 구성할 수 있는 다양한 유형:

postfix_etc_t

이 유형은 **/etc/postfix/** 디렉토리의 **Postfix**의 구성 파일에 사용됩니다.

postfix_data_t

이 유형은 **/var/lib/postfix/** 디렉터리의 **Postfix** 데이터 파일에 사용됩니다.

postfix_var_run_t

이 유형은 **/run/** 디렉터리에 저장된 **Postfix** 파일에 사용됩니다.

postfix_initrc_exec_t

Postfix 실행 파일은 **postfix_initrc_exec_t** 유형으로 레이블이 지정됩니다. 실행하면 **postfix_initrc_t** 도메인으로 전환됩니다.

postfix_spool_t

이 유형은 **/var/spool/** 디렉터리에 저장된 **Postfix** 파일에 사용됩니다.

참고

전체 파일 목록과 **Postfix**의 유형을 보려면 다음 명령을 입력합니다.

```
~]$ grep postfix /etc/selinux/targeted/contexts/files/file_contexts
```

23.3. 부울

SELinux는 서비스를 실행하는 데 필요한 최소 액세스 수준을 기반으로 합니다. 서비스는 다양한 방법으로 실행할 수 있으므로 서비스 실행 방법을 지정해야 합니다. 다음 부울을 사용하여 **SELinux**를 설정합니다.

postfix_local_write_mail_spool

이 부울을 활성화하면 **Postfix**가 시스템의 로컬 메일 스풀에 쓸 수 있습니다. 로컬 스풀을 사용하는 경우 정상적인 작업에 이 부울을 활성화해야 합니다.

참고

SELinux 정책의 지속적인 개발로 인해 위의 목록에 항상 서비스와 관련된 부울이 모두 포함되지 않을 수 있습니다. 나열하려면 다음 명령을 입력합니다.

```
~]$ getsebool -a | grep service_name
```

특정 부울에 대한 설명을 보려면 다음 명령을 입력합니다.

```
~]$ sepolicy booleans -b boolean_name
```

이 명령이 작동하려면 **sepolicy** 유틸리티를 제공하는 추가 **policycoreutils-devel** 패키지가 필요합니다.

23.4. 설정 예

23.4.1. SpamAssassin 및 Postfix

SpamAssasin은 수신 이메일에서 원하지 않는 이메일(스왑 메시지)을 필터링하는 방법을 제공하는 오픈 소스 메일 필터입니다.[23]

Red Hat Enterprise Linux를 사용하는 경우 스파스**ass in** 패키지는 **SpamAssassin**을 제공합니다. 다음 명령을 입력하여 스파스**assin** 패키지가 설치되어 있는지 확인합니다.

```
~]$ rpm -q spamassassin
package spamassassin is not installed
```

설치되지 않은 경우 **yum** 유틸리티를 **root**로 사용하여 설치합니다.

```
~]# yum install spamassassin
```

SpamAssassin은 스팸 필터링 기능을 제공하기 위해 **Postfix**와 같은 메일러와 함께 작동합니다. **SpamAssassin**이 메일을 효과적으로 가로채기, 분석 및 필터링하려면 네트워크 인터페이스에서 수신 대기해야 합니다. **SpamAssassin**의 기본 포트는 **TCP/783**이지만 변경할 수 있습니다. 다음 예제에서는 **SELinux**가 기본적으로 특정 포트에 대한 액세스만 허용하여 **SpamAssassin**을 보완하는 방법을 보여주는 실제 데모를 제공합니다. 이 예제에서는 포트를 변경하는 방법을 보여주며 기본이 아닌 포트에서 **SpamAssassin**이 작동하도록 합니다.

이 예제는 **SELinux**가 **SpamAssassin**의 간단한 설정에 어떤 영향을 줄 수 있는지를 보여줍니다. **SpamAssassin**에 대한 포괄적인 설명서는 이 문서의 범위를 벗어납니다. 자세한 내용은 공식 **SpamAssassin** 설명서 를 참조하십시오. 이 예제에서는 스파스 키언이 설치되어 있다고 가정하고, 사용 중인 포트에 대한 액세스를 허용하도록 방화벽이 구성되었다고 가정하고, **SELinux** 대상 정책이 사용 중이며 **SELinux**가 강제 모드로 실행 중이라고 가정합니다.

절차 23.1. 기본이 아닌 포트에서 SpamAssassin 실행

1. **semanage** 유틸리티를 **root**로 사용하여 **SELinux**가 스팸 데몬이 기본적으로 수신 대기할 수 있는 포트를 표시합니다.

   ```
   ~]# semanage port -l | grep spamd
   spamd_port_t  tcp 783
   ```

   이 출력은 **TCP/783**이 작동할 **SpamAssassin**의 포트로 **spamd_port_t** 에 정의되어 있음을 보여줍니다.

2.

/etc/sysconfig/spa bootstrapass in 구성 파일을 편집하고 예제 포트 TCP/10000 예제에서 SpamAssassin을 시작하도록 수정합니다.

```
# Options to spamd
SPAMDOPTIONS="-d -p 10000 -c m5 -H"
```

이 행은 이제 SpamAssassin이 포트 10000에서 작동하도록 지정합니다. 이 예제의 나머지 부분은 이 소켓을 열 수 있도록 SELinux 정책을 수정하는 방법을 보여줍니다.

3. *시작 SpamAssassin 및 다음과 유사한 오류 메시지가 나타납니다.*

```
~]# systemctl start spamassassin.service
Job for spamassassin.service failed. See 'systemctl status spamassassin.service' and
'journalctl -xn' for details.
```

이 출력은 SELinux가 이 포트에 대한 액세스를 차단했음을 의미합니다.

4. *다음과 유사한 거부 메시지가 SELinux에 의해 기록됩니다.*

```
SELinux is preventing the spamd (spamd_t) from binding to port 10000.
```

5. *루트로 SpamAssassin이 예제 포트(TCP/10000)에서 작동할 수 있도록 semanage 를 실행하여 SELinux 정책을 수정합니다.*

```
~]# semanage port -a -t spamd_port_t -p tcp 10000
```

6. *이제 SpamAssassin이 TCP 포트 10000에서 시작되고 작동하는지 확인합니다.*

```
~]# systemctl start spamassassin.service

~]# netstat -lnp | grep 10000
tcp 0 0 127.0.0.1:10000 0.0.0.0:* LISTEN 2224/spamd.pid
```

7. *이때 스팸은 SELinux 정책에 의해 해당 포트에 액세스할 수 있으므로 TCP 포트 10000에서 올바르게 작동합니다.*

[22]
*자세한 내용은 시스템 관리자 가이드의 **Postfix** 섹션을 참조하십시오.*

[23]
*자세한 내용은 시스템 관리자 가이드의 **Spam Filters** 섹션을 참조하십시오.*

24장. DHCP

dhcpd 데몬은 **Red Hat Enterprise Linux**에서 클라이언트의 **3** 계층 **TCP/IP** 세부 정보를 동적으로 제공하고 구성합니다.

dhcp 패키지에서는 **DHCP** 서버와 **dhcpd** 데몬을 제공합니다. 다음 명령을 입력하여 **dhcp** 패키지가 설치되어 있는지 확인합니다.

```
~]# rpm -q dhcp
package dhcp is not installed
```

설치되지 않은 경우 **yum** 유틸리티를 **root**로 사용하여 설치합니다.

```
~]# yum install dhcp
```

24.1. DHCP 및 SELINUX

dhcpd 가 활성화되면 기본적으로 제한된 상태로 실행됩니다. 제한된 프로세스는 자체 도메인에서 실행되며 다른 제한된 프로세스와 분리됩니다. **SELinux** 정책 구성에 따라 공격자가 제한된 프로세스가 손상되면 공격자가 리소스에 대한 액세스와 가능한 손상을 제한합니다. 다음 예제에서는 자체 도메인에서 실행 중인 **dhcpd** 및 관련 프로세스를 보여줍니다. 이 예제에서는 **dhcp** 패키지가 설치되어 있고 **dhcpd** 서비스가 시작되었다고 가정합니다.

1. **getenforce** 명령을 실행하여 **SELinux**가 강제 모드로 실행 중인지 확인합니다.

   ```
   ~]$ getenforce
   Enforcing
   ```

   명령은 **SELinux**가 강제 모드에서 실행 중일 때 **Enforcing** (강제)을 반환합니다.

2. **root** 사용자로 다음 명령을 입력하여 **dhcpd** 를 시작합니다.

   ```
   ~]# systemctl start dhcpd.service
   ```

   서비스가 실행 중인지 확인합니다. 출력에는 아래 정보가 포함되어야 합니다(시간 스탬프만 다릅니다).

```
~]# systemctl status dhcpd.service
dhcpd.service - DHCPv4 Server Daemon
   Loaded: loaded (/usr/lib/systemd/system/dhcpd.service; disabled)
   Active: active (running) since Mon 2013-08-05 11:49:07 CEST; 3h 20min ago
```

3. 다음 명령을 실행하여 **dhcpd** 프로세스를 확인합니다.

```
~]$ ps -eZ | grep dhcpd
system_u:system_r:dhcpd_t:s0 5483 ?        00:00:00 dhcpd
```

dhcpd 프로세스와 연결된 **SELinux** 컨텍스트는 **system_u:system_r:dhcpd_t:s0**입니다.

24.2. 유형

고급 프로세스 격리를 제공하기 위해 **SELinux** 대상 정책에 사용되는 기본 권한 제어 방법은 **Type Enforcement**입니다. 모든 파일과 프로세스는 유형으로 레이블이 지정됩니다. **type**은 프로세스의 **SELinux** 도메인과 파일의 **SELinux** 유형을 정의합니다. **SELinux** 정책 규칙은 유형에 액세스하는 도메인이든 다른 도메인에 액세스하는 도메인이든 관계없이 유형이 서로 액세스하는 방법을 정의합니다. 액세스는 허용하는 특정 **SELinux** 정책 규칙이 있는 경우에만 허용됩니다.

DHCP와 함께 다음 유형이 사용됩니다.

dhcp_etc_t

이 유형은 구성 파일을 포함하여 **/etc** 디렉토리의 파일에 주로 사용됩니다.

dhcpd_var_run_t

이 유형은 **/var/run/** 디렉터리에서 **dhcpd** 의 **PID** 파일에 사용됩니다.

dhcpd_exec_t

이 유형은 **DHCP** 실행 파일을 **dhcpd_t** 도메인으로 전환하는 데 사용됩니다.

dhcpd_initrc_exec_t

이 유형은 **DHCP** 실행 파일을 **dhcpd_initrc_t** 도메인으로 전환하는 데 사용됩니다.

참고

*전체 파일 목록과 **dhcpd** 의 유형을 보려면 다음 명령을 입력합니다.*

```
~]$ grep dhcp /etc/selinux/targeted/contexts/files/file_contexts
```

25장. RED HAT의 OPENSHIFT

Red Hat의 OpenShift는 개발자가 웹 애플리케이션을 빌드하고 배포할 수 있는 PaaS(서비스로서의 플랫폼)입니다. OpenShift는 Java, Ruby 및 PHP를 비롯한 다양한 프로그래밍 언어 및 프레임워크를 제공합니다. 또한 Eclipse 통합, JBoss Developer Studio 및 Jenkins를 비롯한 애플리케이션 라이프사이클을 지원하는 통합 개발자 툴도 제공합니다. OpenShift는 오픈 소스 에코시스템을 사용하여 모바일 애플리케이션, 데이터베이스 서비스 등을 위한 플랫폼을 제공합니다. [24]

Red Hat Enterprise Linux에서 openshift-clients 패키지는 OpenShift 클라이언트 툴을 제공합니다. 다음 명령을 입력하여 해당 명령이 설치되었는지 확인합니다.

```
~]$ rpm -q openshift-clients
package openshift-clients is not installed
```

openshift-clients 패키지가 설치되지 않은 경우 OpenShift 클라이언트 도구 설치 가이드 및 OpenShift 온라인 클라이언트 도구 설치 가이드를 참조하십시오.

중요

이전에는 rhc 패키지에서 OpenShift 클라이언트 툴을 제공했습니다. 최신 OpenShift 버전에서는 이 패키지가 더 이상 사용되지 않으며 Red Hat에서 더 이상 지원되지 않습니다. 따라서 OpenShift 버전 2 이후에 rhc 패키지는 지원되는 OpenShift 버전에 사용되는 OpenShift 클라이언트 툴을 제공하는 openshift-clients 패키지로 교체됩니다.

25.1. OPENSHIFT 및 SELINUX

SELinux는 모든 프로세스가 SELinux 정책에 따라 레이블이 지정되므로 OpenShift를 사용하는 애플리케이션에 대해 더 나은 보안 제어를 제공합니다. 따라서 SELinux는 동일한 노드에서 실행되는 다양한 기어 내에서 가능한 악의적인 공격으로부터 OpenShift를 보호합니다.

SELinux 및 OpenShift에 대한 자세한 내용은 Dan Walsh의 프레젠테이션을 참조하십시오.

25.2. 유형

고급 프로세스 격리를 제공하기 위해 SELinux 대상 정책에 사용되는 기본 권한 제어 방법은 Type Enforcement입니다. 모든 파일과 프로세스는 유형으로 레이블이 지정됩니다. type은 프로세스의 SELinux 도메인과 파일의 SELinux 유형을 정의합니다. SELinux 정책 규칙은 유형에 액세스하는 도메인이든 다른 도메인에 액세스하는 도메인이든 관계없이 유형이 서로 액세스하는 방법을 정의합니다. 액세스는 허용하는 특정 SELinux 정책 규칙이 있는 경우에만 허용됩니다.

다음 유형은 **OpenShift**와 함께 사용됩니다. 다양한 유형을 사용하여 유연한 액세스를 구성할 수 있습니다.

프로세스 유형

openshift_t

OpenShift 프로세스는 **openshift_t SELinux** 유형과 연결됩니다.

실행 파일 유형

openshift_cgroup_read_exec_t

SELinux를 사용하면 이 유형의 파일이 **openshift_cgroup_read_t** 도메인으로 실행 파일을 전환할 수 있습니다.

openshift_cron_exec_t

SELinux를 사용하면 이 유형의 파일이 **openshift_cron_t** 도메인으로 실행 파일을 전환할 수 있습니다.

openshift_initrc_exec_t

SELinux를 사용하면 이 유형의 파일이 **openshift_initrc_t** 도메인으로 실행 파일을 전환할 수 있습니다.

쓰기 가능한 유형

openshift_cgroup_read_tmp_t

이 유형을 사용하면 **OpenShift** 제어 그룹(**cgroup**)에서 **/tmp** 디렉토리의 임시 파일을 읽고 액세스할 수 있습니다.

openshift_cron_tmp_t

이 유형을 사용하면 **OpenShift cron** 작업의 임시 파일을 **/tmp** 에 저장할 수 있습니다.

openshift_initrc_tmp_t

이 유형을 사용하면 **OpenShift initrc** 임시 파일을 **/tmp** 에 저장할 수 있습니다.

openshift_log_t

이 유형의 파일은 **OpenShift** 로그 데이터로 취급되며 일반적으로 **/var/log/** 디렉터리에 저장됩니다.

openshift_rw_file_t

OpenShift는 이 유형으로 레이블이 지정된 파일에 읽고 쓸 수 있는 권한을 갖습니다.

openshift_tmp_t

이 유형은 **OpenShift** 임시 파일을 **/tmp** 에 저장하는 데 사용됩니다.

openshift_tmpfs_t

이 유형을 사용하면 **OpenShift** 데이터를 **tmpfs** 파일 시스템에 저장할 수 있습니다.

openshift_var_lib_t

이 유형을 사용하면 **OpenShift** 파일을 **/var/lib/** 디렉터리에 저장할 수 있습니다.

openshift_var_run_t

이 유형을 사용하면 **OpenShift** 파일을 **/run/** 또는 **/var/run /** 디렉터리에 저장할 수 있습니다.

25.3. 부울

SELinux는 서비스를 실행하는 데 필요한 최소 액세스 수준을 기반으로 합니다. 서비스는 다양한 방법으로 실행할 수 있으므로 서비스 실행 방법을 지정해야 합니다. 다음 부울을 사용하여 **SELinux**를 설정합니다.

openshift_use_nfs

이 부울을 활성화하면 **NFS** 공유에 **OpenShift**를 설치할 수 있습니다.

참고

SELinux 정책의 지속적인 개발로 인해 위의 목록에 항상 서비스와 관련된 부울이 모두 포함되지 않을 수 있습니다. 나열하려면 다음 명령을 입력합니다.

```
~]$ getsebool -a | grep service_name
```

특정 부울에 대한 설명을 보려면 다음 명령을 입력합니다.

```
~]$ sepolicy booleans -b boolean_name
```

이 명령이 작동하려면 **sepolicy** 유틸리티를 제공하는 추가 **policycoreutils-devel** 패키지가 필요합니다.

25.4. 설정 예

25.4.1. 기본 OpenShift 디렉터리 변경

기본적으로 **OpenShift**는 **openshift_ var_lib_t SELinux** 유형으로 레이블이 지정된 **/var/lib/openshift/** 디렉터리에 데이터를 저장합니다. **OpenShift**가 다른 디렉터리에 데이터를 저장할 수 있도록 하려면 새 디렉터리의 레이블을 적절한 **SELinux** 컨텍스트로 지정합니다.

다음 절차에서는 데이터를 **/srv/openshift/** 로 저장하기 위해 기본 **OpenShift** 디렉터리를 변경하는 방법을 보여줍니다.

절차 **25.1.** 데이터 저장을 위한 기본 **OpenShift** 디렉터리 변경

1. **root**로 **/srv** 디렉터리에 새 **openshift/** 디렉터리를 생성합니다. 새 디렉터리는 **var_t** 유형으로 레이블이 지정됩니다.

```
~]# mkdir /srv/openshift
```

```
~]$ ls -Zd /srv/openshift
drwxr-xr-x. root root unconfined_u:object_r:var_t:s0   openshift/
```

2. **root**로 **semanage** 유틸리티를 사용하여 **/srv/openshift/** 를 적절한 **SELinux** 컨텍스트에 매핑합니다.

```
~]# semanage fcontext -a -e /var/lib/openshift /srv/openshift
```

3. 그런 다음 **restorecon** 유틸리티를 **root**로 사용하여 변경 사항을 적용합니다.

```
~]# restorecon -R -v /srv/openshift
```

4. 이제 **/srv/openshift/** 디렉터리에 올바른 **openshift_var_lib_t** 유형으로 레이블이 지정됩니다.

```
~]$ ls -Zd /srv/openshift
drwxr-xr-x. root root unconfined_u:object_r:openshift_var_lib_t:s0   openshift/
```

---

[24] **OpenShift**에 대한 자세한 내용은 **OpenShift Container Platform** 제품 설명서 및 **OpenShift Online** 제품 설명서를 참조하십시오.

26장. IDM (IDENTITY MANAGEMENT)

IdM(Identity Management)은 PAM, LDAP, Kerberos, DNS, NTP, 인증서 서비스를 비롯한 표준 정의 일반 네트워크 서비스를 위한 통합 환경을 제공합니다. IdM 을 사용하면 Red Hat Enterprise Linux 시스템이 도메인 컨트롤러 역할을 할 수 있습니다.[25]

Red Hat Enterprise Linux에서 ipa-server 패키지는 IdM 서버를 제공합니다. 다음 명령을 입력하여 ipa-server 패키지가 설치되어 있는지 확인합니다.

```
~]$ rpm -q ipa-server
package ipa-server is not installed
```

설치되지 않은 경우 root 사용자로 다음 명령을 입력하여 설치합니다.

```
~]# yum install ipa-server
```

26.1. ID 관리 및 SELINUX

ID 관리는 IdM 사용자가 IdM 액세스 권한에 대한 SELinux 컨텍스트를 지정할 수 있도록 호스트당 SELinux 역할을 구성하도록 매핑할 수 있습니다. 사용자 로그인 프로세스 중에SSSD(System Security Services Daemon)는 특정 IdM 사용자에게 정의된 액세스 권한을 쿼리합니다. 그런 다음 pam_selinux 모듈은 IdM 액세스 권한에 따라 적절한 SELinux 컨텍스트로 사용자 프로세스를 시작하기 위해 커널에 요청을 보냅니다(예: guest_u:guest_r:guest_t:s0).

ID 관리 및 SELinux에 대한 자세한 내용은 Red Hat Enterprise Linux 7용 Linux 도메인, ID, 인증 및 정책 가이드를 참조하십시오.

26.1.1. Active Directory 도메인 신뢰

이전 버전의 Red Hat Enterprise Linux에서 ID 관리는 WinSync 유틸리티를 사용하여AD(Active Directory) 도메인의 사용자가 IdM 도메인에 저장된 데이터에 액세스할 수 있도록 했습니다. 이를 위해 WinSync 는 AD 서버에서 로컬 서버로 사용자 및 그룹 데이터를 복제하고 데이터를 동기화해야 했습니다.

Red Hat Enterprise Linux 7에서 SSSD 데몬은 AD와 사용자가 IdM 및 AD 도메인 간에 신뢰할 수 있는 관계를 생성할 수 있도록 개선되었습니다. 사용자 및 그룹 데이터는 AD 서버에서 직접 읽습니다. 또한 AD 및 IdM 도메인 간에SSO(Single Sign-On) 인증을 허용하는 Kerberos 교차 영역 신뢰도 제공됩니다.

SSO가 설정된 경우 AD 도메인의 사용자는 암호 없이 IdM 도메인에 저장된 Kerberos로 보호되는 데이터에 액세스할 수 있습니다.

이 기능은 기본적으로 설치되지 않습니다. 이를 사용하려면 추가 ipa-server-trust-ad 패키지를 설치합니다.

26.2. 설정 예

26.2.1. IdM 사용자에게 SELinux 사용자를 매핑

다음 절차에서는 새 SELinux 매핑을 생성하는 방법과 이 매핑에 새 IdM 사용자를 추가하는 방법을 보여줍니다.

절차 26.1. SELinux 맵핑에 사용자 추가 방법

1. *새 SELinux 매핑을 생성하려면 SELinux_mapping이 새 SELinux 매핑의 이름이고 --selinuxuser 옵션은 특정 SELinux 사용자를 지정하는 다음 명령을 입력합니다.*

```
~]$ ipa selinuxusermap-add SELinux_mapping --selinuxuser=staff_u:s0-s0:c0.c1023
```

2. *다음 명령을 입력하여 사용자 이름이 tuser 인 IdM 사용자를 SELinux 매핑에 추가합니다.*

```
~]$ ipa selinuxusermap-add-user --users=tuser SELinux_mapping
```

3. *SELinux 매핑에 ipaclient.example.com 이라는 새 호스트를 추가하려면 다음 명령을 입력합니다.*

```
~]$ ipa selinuxusermap-add-host --hosts=ipaclient.example.com SELinux_mapping
```

4. *ipaclient.example.com 호스트에 로그인하면 tuser 사용자는 staff_u:s0-s0:c0.c1023 레이블을 가져옵니다.*

```
[tuser@ipa-client]$ id -Z
staff_u:staff_r:staff_t:s0-s0:c0.c1023
```

[25] ID 관리에 대한 자세한 내용은 Red Hat Enterprise Linux 7용 Linux 도메인, ID, 인증 및 정책 가이드를 참조하십시오.

27장. RED HAT GLUSTER STORAGE

Red Hat Gluster Storage 는 기업에 유연하고 경제적인 비구조적 데이터 스토리지를 제공합니다. Gluster의 주요 구성 요소인 GlusterFS 는 스택 가능한 사용자 공간 설계를 기반으로 하며 다양한 스토리지 서버를 네트워크를 통해 집계하고 이를 하나의 대형 병렬 네트워크 파일 시스템으로 상호 연결합니다. XFS 파일 시스템 형식을 사용하여 디스크에 데이터를 저장하는 POSIX 호환 GlusterFS 서버는 NFS 및 CIFS를 포함한 업계 표준 액세스 프로토콜을 사용하여 액세스할 수 있습니다.

자세한 내용은 제품 설명서에서 Red Hat Gluster Storage 컬렉션 가이드를 참조하십시오.

glusterfs-server 패키지는 Red Hat Gluster Storage를 제공합니다. 설치 프로세스에 대한 자세한 내용은 Red Hat Gluster Storage 설치 가이드를 참조하십시오.

27.1. RED HAT GLUSTER STORAGE 및 SELINUX

활성화되면 SELinux는 glusterd (GlusterFS 관리 서비스) 및 NFS(Glusterfs d) 프로세스에 대해 유연한 필수 액세스 제어를 Red Hat Gluster Storage의 일부로 제공하여 추가 보안 계층으로 작동합니다. 이러한 프로세스는 glusterd_t SELinux 유형으로 바인딩되지 않은 고급 프로세스 격리를 제공합니다.

27.2. 유형

고급 프로세스 격리를 제공하기 위해 SELinux 대상 정책에 사용되는 기본 권한 제어 방법은 Type Enforcement입니다. 모든 파일과 프로세스는 유형으로 레이블이 지정됩니다. type은 프로세스의 SELinux 도메인과 파일의 SELinux 유형을 정의합니다. SELinux 정책 규칙은 유형에 액세스하는 도메인이든 다른 도메인에 액세스하는 도메인이든 관계없이 유형이 서로 액세스하는 방법을 정의합니다. 액세스는 허용하는 특정 SELinux 정책 규칙이 있는 경우에만 허용됩니다.

Red Hat Gluster Storage와 함께 다음 유형을 사용합니다. 다양한 유형을 사용하여 유연한 액세스를 구성할 수 있습니다.

프로세스 유형

glusterd_t

Gluster 프로세스는 glusterd_t SELinux 유형과 연결됩니다.

실행 파일 유형

glusterd_initrc_exec_t

Gluster init 스크립트 파일의 SELinux별 스크립트 유형 컨텍스트입니다.

glusterd_exec_t

Gluster 실행 파일의 SELinux 특정 실행 파일 유형 컨텍스트입니다.

포트 유형

gluster_port_t

이 유형은 glusterd 에 대해 정의됩니다. 기본적으로 glusterd 는 204007-24027 및 38465-38469 TCP 포트를 사용합니다.

파일 문맥

glusterd_brick_t

이 유형은 glusterd brick 데이터로 위협된 파일에 사용됩니다.

glusterd_conf_t

이 유형은 glusterd 구성 데이터와 연결되며 일반적으로 /etc 디렉토리에 저장됩니다.

glusterd_log_t

이 유형의 파일은 glusterd 로그 데이터로 처리되며 일반적으로 /var/log/ 디렉터리에 저장됩니다.

glusterd_tmp_t

이 유형은 glusterd 임시 파일을 /tmp 디렉터리에 저장하는 데 사용됩니다.

glusterd_var_lib_t

이 유형을 사용하면 glusterd 파일을 /var/lib/ 디렉토리에 저장할 수 있습니다.

glusterd_var_run_t

이 유형을 사용하면 glusterd 파일을 /run/ 또는 /var/run / 디렉토리에 저장할 수 있습니다.

27.3. 부울

SELinux는 서비스를 실행하는 데 필요한 최소 액세스 수준을 기반으로 합니다. 서비스는 다양한 방법으로 실행할 수 있으므로 서비스 실행 방법을 지정해야 합니다. 다음 부울을 사용하여 **SELinux**를 설정합니다.

gluster_export_all_ro

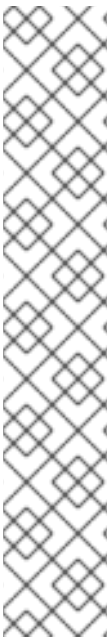
이 부울을 활성화하면 **glusterfsd** 가 파일 및 디렉토리를 읽기 전용으로 공유할 수 있습니다. 이 부울은 기본적으로 비활성화되어 있습니다.

gluster_export_all_rw

이 부울을 활성화하면 **glusterfsd** 가 읽기 및 쓰기 액세스 권한으로 파일 및 디렉토리를 공유할 수 있습니다. 이 부울은 기본적으로 활성화되어 있습니다.

gluster_anon_write

이 부울을 활성화하면 **glusterfsd** 에서 **public _content_rw_t SELinux** 유형으로 레이블이 지정된 공용 파일을 수정할 수 있습니다.



참고

SELinux 정책의 지속적인 개발로 인해 위의 목록에 항상 서비스와 관련된 부울이 모두 포함되지 않을 수 있습니다. 나열하려면 다음 명령을 입력합니다.

```
~]$ getsebool -a | grep service_name
```

특정 부울에 대한 설명을 보려면 다음 명령을 입력합니다.

```
~]$ sepolicy booleans -b boolean_name
```

이 명령이 작동하려면 **sepolicy** 유틸리티를 제공하는 추가 **policycoreutils-devel** 패키지가 필요합니다.

27.4. 설정 예

27.4.1. Gluster brick에 레이블 지정

Gluster brick은 신뢰할 수 있는 스토리지 풀에 있는 서버의 내보내기 디렉터리입니다. **brick**에 올바른 **SELinux** 컨텍스트(**glusterd _brick_t**)로 레이블이 지정되지 않은 경우 **SELinux**는 특정 파일 액세스

작업을 거부하고 다양한 **AVC** 메시지를 생성합니다.

다음 절차에서는 **Gluster brick**에 올바른 **SELinux** 컨텍스트로 레이블을 지정하는 방법을 보여줍니다. 이 절차에서는 이전에 논리 볼륨(예: **/dev/rhgs/gluster**)을 **Gluster brick**으로 사용하도록 생성하고 포맷했다고 가정합니다.

Gluster brick에 대한 자세한 내용은 **Red Hat Gluster Storage** 관리 가이드 의 **Red Hat Gluster Storage** 볼륨 장을 참조하십시오.

절차 **27.1. Gluster Brick**에 레이블을 지정하는 방법

1. 디렉터리를 만들어 이전에 포맷한 논리 볼륨을 마운트합니다. 예를 들어 다음과 같습니다.

```
~]# mkdir /mnt/brick1
```

2. 논리 볼륨(이 경우 **/dev/vg-group/gluster**)을 이전 단계에서 만든 **/mnt/brick1/** 디렉터리에 마운트합니다.

```
~]# mount /dev/vg-group/gluster /mnt/brick1/
```

mount 명령은 장치를 일시적으로만 마운트합니다. 장치를 영구적으로 마운트하려면 다음 항목과 유사한 항목을 **/etc/fstab** 파일에 추가합니다.

```
/dev/vg-group/gluster    /mnt/brick1  xfs rw,inode64,noatime,nouuid      1 2
```

자세한 내용은 **fstab(5)** 도움말 페이지를 참조하십시오.

3. **/mnt/brick1/** 의 **SELinux** 컨텍스트를 확인합니다.

```
~]$ ls -lZd /mnt/brick1/
drwxr-xr-x. root root system_u:object_r:unlabeled_t:s0 /mnt/brick1/
```

디렉터리는 **unlabeled_t SELinux** 유형으로 레이블이 지정됩니다.

4. **/mnt/brick1/** 의 **SELinux** 유형을 **glusterd_brick_t SELinux** 유형으로 변경합니다.

```
~]# semanage fcontext -a -t glusterd_brick_t "/mnt/brick1(/.*)?"
```

5. ***restorecon** 유틸리티를 사용하여 변경 사항을 적용합니다.*

```
~]# restorecon -Rv /mnt/brick1
```

6. *마지막으로 컨텍스트가 성공적으로 변경되었는지 확인합니다.*

```
~]$ ls -lZd /mnt/brick1
drwxr-xr-x. root root system_u:object_r:glusterd_brick_t:s0 /mnt/brick1/
```

28장. 참고 자료

다음 참조는 **SELinux**와 관련이 있지만 본 가이드의 범위를 벗어난 추가 정보에 대한 포인터입니다. **SELinux**의 신속한 개발로 인해 이 자료 중 일부는 **Red Hat Enterprise Linux**의 특정 릴리스에만 적용될 수 있습니다.

서적

에제별 SELinux

Mayer, MacMillan 및 Caplan

Pcurrentice Hall, 2007년

SELinux: NSA의 오픈 소스 보안 강화 Linux

Bill Mcty

O'Reilly Media Inc., 2004

튜토리얼 및 도움말

Russell Coker의 튜토리얼 및 대화

http://www.coker.com.au/selinux/talks/ibmtu-2004/

Dan Walsh's Journal

http://danwalsh.livejournal.com/

Red Hat Knowledgebase

https://access.redhat.com/site/

일반 정보

NSA SELinux 메인 웹 사이트

https://www.nsa.gov/What-We-Do/Research/SELinux/

NSA SELinux FAQ

https://www.nsa.gov/What-We-Do/Research/SELinux/FAQs/

메일링 리스트

NSA SELinux 메일링 리스트

https://www.nsa.gov/What-We-Do/Research/SELinux/Mailing-List/

Fedora SELinux 메일링 리스트

http://www.redhat.com/mailman/listinfo/fedora-selinux-list

커뮤니티

SELinux 프로젝트 Wiki

http://selinuxproject.org/page/Main_Page

SELinux 커뮤니티 페이지

http://selinux.sourceforge.net/

IRC

iRC.freenode.net, #selinux

부록 A. 개정 내역

고침 0.3-06 Fri Aug 9 2019 Mirek Jahoda
7.7 GA 게시 버전.

고침 0.3-05 Sat Oct 20 2018 Mirek Jahoda
7.6 GA 게시 버전.

고침 0.3-03 Tue Apr 3 2018 Mirek Jahoda
7.5 GA 게시 버전.

고침 0.3-01 Thu Jul 13 2017 Mirek Jahoda
7.4 GA 게시 버전.

고침 0.2-18 Wed Nov 2 2016 Mirek Jahoda
7.3 GA 게시 버전.

고침 0.2-11 Sun Jun 26 2016 Mirek Jahoda
수정 사항이 포함된 Async 릴리스.

고침 0.2-10 Sun Feb 14 2016 Robert Krátký
수정 사항이 포함된 Async 릴리스.

고침 0.2-9 Thu Dec 10 2015 Barbora Ančincová
Red Hat Gluster Storage 장이 추가되었습니다.

고침 0.2-8 Thu Nov 11 2015 Barbora Ančincová
Red Hat Enterprise Linux 7.2 GA GA 릴리스.

고침 0.2-7 Thu Aug 13 2015 Barbora Ančincová
Red Hat Enterprise Linux 7.2 베타 릴리스.

고침 0.2-6 Wed Feb 18 2015 Barbora Ančincová
Red Hat Enterprise Linux 7.1 GA 릴리스, 이 책.

고침 0.2-5 Fri Dec 05 2014 Barbora Ančincová
Red Hat 고객 포털에서 주문을 정렬하도록 업데이트.

고침 0.2-4 Thu Dec 04 2014 Barbora Ančincová
Red Hat Enterprise Linux 7.1 베타 버전의 설명서 릴리스.

고침 0.1-41 Tue May 20 2014 Tomáš Čapek
스타일 변경에 대해 다시 빌드합니다.

고침 0.1-1 Tue Jan 17 2013 Tomáš Čapek
Red Hat Enterprise Linux 7에 대한 책 초기 작성