Red Hat Enterprise Linux 7

보안 가이드

RHEL 서버 및 워크스테이션의 보안을 위한 개념 및 기술

Last Updated: 2025-01-21

Red Hat Enterprise Linux 7 보안 가이드

RHEL 서버 및 워크스테이션의 보안을 위한 개념 및 기술

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 사용자와 관리자가 로컬 및 원격 침입, 악용 및 악용 활동으로부터 워크스테이션 및 서버 보안 프로세스 및 방법을 학습할 수 있도록 지원합니다. 이는 Red Hat Enterprise Linux를 대상으로 하지만 개념과 기술은 모든 Linux 시스템에 적용할 수 있으며, 데이터 센터, 직장 및 가정을 위한 안전한 컴퓨팅 환경을 만드는 데 관련된 계획과 툴에 대해 자세히 설명합니다. 적절한 관리 지식, 경계 시스템 및 툴을 통해 Linux를 실행하는 시스템의 기능을 최대한 활용하여 가장 일반적인 침입과 악용 기술로 부터 시스템 보안을 유지할 수 있습니다.

차례

1장. 보안 주제 개요

기업 운영을 지원하고 개인 정보를 추적하는 데 도움이 되는 강력한 네트워크로 연결된 컴퓨터에 대한 의존도가 높아짐에 따라 산업 전체가 네트워크 및 컴퓨터 보안의 실례를 중심으로 형성되었습니다. 기업은 시스템을 적절히 감사하고 조직의 운영 요구 사항에 맞게 솔루션을 조정할 수 있도록 보안 전문가의 지식과 기술을 요청해 왔습니다. 대부분의 조직은 본질적으로 점차 동적인 상황으로 인해 핵심 기업 IT 리소스에 로컬 및 원격으로 액세스하고 있기 때문에 안전한 컴퓨팅 환경에 대한 요구가 더욱 높아졌습니다.

안타깝게도 많은 조직 (및 개별 사용자)들은 보안을 나중으로 미루고 성능 향상, 생산성, 편의성, 사용 편의성, 예산 문제 등에 더 중점을 둡니다. 보안 구현은 일반적으로 시스템 침입이 *발생한 후* 수행됩니다. 인터넷과 같이 신뢰할 수 없는 네트워크에 사이트를 연결하기 전에 올바른 조치를 취하는 것이 침입 시도로 부터 시스템을 방어하는 효과적인 방법입니다.

참고

이 문서에서는 **/lib** 디렉토리에 있는 파일에 대한 여러 참조를 만듭니다. 64비트 시스템을 사용하는 경우, 대신 언급된 일부 파일은 **/lib64**에 위치할 수 있습니다.

1.1. 컴퓨터 보안이란 무엇입니까?

컴퓨터 보안은 광범위한 컴퓨팅 및 정보 처리 영역을 포괄하는 일반적인 용어입니다. 일상 비즈니스 트랜잭션을 수행하고 중요한 정보에 액세스하는 컴퓨터 시스템과 네트워크에 의존하는 산업은 데이터를 전체 자산의 중요한 부분으로 간주합니다. 총 소유 비용(TCO), 투자 수익률(ROI), QoS(Quality of Service)와 같은 용어 및 평가 지표가 일상적인 비즈니스 용어로 사용되도록 되어 있습니다. 이러한 지표를 사용하여 업계는 계획 및 프로세스 관리 비용의 일부로 데이터 무결성 및 HA(고가용성)와 같은 측면을 계산할 수 있습니다. 전자 상거래와 같은 일부 업계에서 데이터의 가용성과 신뢰성은 성공과 실패의 차이를 의미합니다.

1.1.1. 보안 표준화

모든 업계의 기업들은 AMA(American Medical Association) 또는 IEEE(Institute of Electrical andcomputer Engineers)와 같은 표준 제작 기관에서 정하는 규정 및 규칙에 의존하고 있습니다. 정보 보안에도 동일한 개념이 적용됩니다. 많은 보안 컨설턴트와 벤더가 CIA 또는 *기밀성, 무결성 및 가용성* 이라는 표준 보안 모델을 채용하고 있습니다. 이 3계층 모델은 중요한 정보의 리스트를 평가하고 보안 정책을 설정하는 데 일반적으로 채용되는 구성 요소입니다. 다음은 CIA 모델을 자세히 설명합니다.

- 기밀성 - 중요한 정보는 사전 정의된 개인에게만 제공되어야 합니다. 정보의 무단 전송 및 사용을 제한해야 합니다. 예를 들어, 정보의 기밀성으로 인해 고객의 개인 정보 또는 금융 정보는 권한이 없는 개인이 신원 도용이나 신용 사기와 같은 악의적인 목적으로 무단으로 수집되지 않도록 합니다.

- 무결성 - 불완전하거나 잘못된 방식으로 정보를 변경할 수 없습니다. 권한이 없는 사용자가 민감한 정보를 수정하거나 제거할 수 없도록 제한되어야 합니다.

- 가용성 - 필요한 경우 권한 있는 사용자가 필요에 따라 정보에 액세스할 수 있어야 합니다. 가용성은 합의된 빈도와 타임라인을 통해 정보를 얻을 수 있음을 보장합니다. 이는 종종 백분율로 측정되며 네트워크 서비스 공급자 및 해당 엔터프라이즈 클라이언트가 사용하는 SLA(서비스 수준 계약)에서 공식적으로 동의합니다.

1.1.2. 암호화 소프트웨어 및 인증

다음 Red Hat 지식베이스 문서는 Red Hat Enterprise Linux 코어 암호화 구성 요소에 대한 개요를 제공하고, 어떤 요소를 선택했는지, 운영 체제에 어떻게 통합되는지, 하드웨어 보안 모듈 및 스마트 카드를 지원하는 방법, 암호화 인증의 적용 방법에 대해 설명합니다.

- RHEL7 코어 암호화 구성 요소

1.2. 보안 제어

컴퓨터 보안은 종종 *컨트롤* 이라고 하는 세 가지 개별 마스터 범주로 나뉩니다.

- 물리적
- 기술적
- 관리적

이 세 가지 범주는 보안을 적절하게 구현하는 주요 목적을 정의합니다. 이러한 제어에는 제어에 대한 세부 정보와 구현 방법을 자세히 설명하는 하위 범주가 있습니다.

1.2.1. 물리적 제어

물리적 제어는 중요한 자료에 대한 무단 액세스를 차단하거나 방지하기 위해 정의된 구조로 구현된 보안 조치입니다 물리적 제어의 예는 다음과 같습니다.

- CCTV 카메라
- 모션 또는 온도 감지 경보 시스템
- 경비원
- 사진이 부착된 신분증
- 금속 도어 잠금
- 생체 인식(지문, 음성, 얼굴, 홍채, 필적 및 개인 식별을 위한 기타 자동 인식 방법 포함)

1.2.2. 기술 제어

기술 제어는 물리적 구조와 네트워크 상에서 중요한 데이터의 액세스 및 사용을 제어하기 위한 기반으로 기술을 사용합니다. 기술 제어 범위는 다음 기술을 포함하여 매우 광범위합니다.

- 암호화
- 스마트 카드
- 네트워크 인증
- ACL(액세스 제어 목록)
- 파일 무결성 감사 소프트웨어

1.2.3. 관리 제어

관리 제어는 보안의 인적 요소를 정의합니다. 이들은 조직 내에서 모든 수준의 인력을 투입하고 다음과 같이 어떤 리소스와 정보에 액세스할 수 있는지 결정합니다.

- 교육 및 인식 향상
- 재해 준비 및 복구 계획

- 인원 채용 및 분리 전략
- 인사 등록 및 회계

1.3. 취약점 평가

시간, 리소스 및 동기를 감안할 경우 공격자는 거의 모든 시스템에 침입할 수 있습니다. 현재 사용 가능한 모든 보안 절차와 기술은 침입으로부터 모든 시스템이 완전히 안전하다고 보장할 수 없습니다. 라우터는 인터넷 게이트웨이 보안에 도움이 됩니다. 방화벽은 네트워크의 에지를 보호하는 데 도움이 됩니다. 가상 사설 네트워크는 암호화된 스트림에 데이터를 안전하게 전송합니다. 침입 감지 시스템은 악의적인 활동에 대해 경고합니다. 그러나 이러한 각 기술의 성공 여부는 다음과 같은 여러 변수에 따라 달라집니다.

- 기술 구성, 모니터링 및 유지 관리를 담당하는 직원의 전문 지식
- 서비스와 커널을 빠르고 효율적으로 패치하고 업데이트하는 기능
- 네트워크를 지속적으로 관리해야 하는 사용자의 능력

데이터 시스템과 기술의 역동적인 상태를 고려할 때 엔터프라이즈 리소스 보안은 매우 복잡할 수 있습니다. 이러한 복잡성 때문에 모든 시스템에 대한 전문가 리소스를 찾기가 어려울 수 있습니다. 정보 보안의 많은 분야에서 높은 수준의 지식을 갖춘 인력을 보유할 수는 있지만, 여러 분야를 전문으로 하는 인력을 확보하는 것은 쉽지 않습니다. 이는 주로 정보 보안의 각 전문 분야에서 지속적인 관심과 집중이 필요하기 때문입니다. 정보 보안은 그대로 유지되지 않습니다.

취약점 평가는 네트워크 및 시스템 보안에 대한 내부 감사입니다. 이 결과는 네트워크의 기밀성, 무결성 및 가용성(1.1.1절. “보안 표준화”에서 설명됨)을 나타냅니다. 일반적으로 취약점 평가는 조정 단계부터 시작하여 대상 시스템 및 리소스와 관련된 중요한 데이터를 수집합니다. 이 단계는 시스템 준비 단계로 이어지며, 대상에서 기본적으로 알려진 모든 취약점이 있는지 확인합니다. 준비 단계는 보고 단계에서 최고조에 달하며, 여기서 결과는 높은 위험, 중간 위험, 낮은 위험의 범주로 분류되며, 대상의 보안 개선(또는 취약점의 위험 완화) 방법이 논의됩니다.

집의 취약성 평가를 수행하는 경우 집의 모든 문이 닫혀 있고 잠겨 있는지 확인합니다. 또한 모든 창을 검사하여 완전히 닫힌지 확인하고 올바르게 잠겨있는지 확인해야 합니다. 시스템, 네트워크 및 전자 데이터에도 동일한 개념이 적용됩니다. 악의적인 사용자는 데이터를 훔치고 파괴합니다. 악의적인 사용자가 사용하는 툴, 사고 방식 및 동기에 집중하면 악의적인 사용자의 행동에 신속하게 대응할 수 있습니다.

1.3.1. 평가 및 테스트 정의

취약점 평가는*외부적 평가* 및 *내부적 평가*의 두 가지 유형으로 분류할 수 있습니다.

외부적 평가에서 취약점 평가를 수행할 때 외부에서 시스템을 손상시키려고 합니다. 귀하의 회사 외부에 있다는 것은 크래커의 관점을 제공합니다. 공개적으로 라우팅 가능한 IP 주소, *DMZ* 시스템, 방화벽의 외부 인터페이스 등 공격자가 볼 수 있는 것을 확인할 수 있습니다. DMZ는 회사 사설 LAN과 같은 신뢰할 수 있는 내부 네트워크와 공용 인터넷과 같은 신뢰할 수 없는 외부 네트워크 사이에 있는 컴퓨터 또는 소규모 하위 네트워크에 해당하는 "비무장 영역"을 나타냅니다. 일반적으로 Clevis에는 웹(HTTP) 서버, FTP 서버, SMTP(e-mail) 서버 및 DNS 서버와 같은 인터넷 트래픽에 액세스할 수 있는 장치가 포함되어 있습니다.

내부적 취약점 평가를 수행하는 경우 내부적이고 상태가 신뢰할 수 있기 때문에 이점이 있습니다. 이는 사용자 및 동료가 시스템에 한 번 로그인할 수 있는 관점입니다. 인쇄 서버, 파일 서버, 데이터베이스 및 기타 리소스가 표시됩니다.

두 가지 유형의 취약점 평가 사이에는 차이점이 있습니다. 사내 사용자는 외부 사용자보다 더 많은 권한을 얻게 됩니다. 대부분의 조직에서는 침입을 방지하도록 보안이 구성되어 있습니다. 조직의 내부(예: 부서 방화벽, 사용자 수준 액세스 제어 및 내부 리소스의 인증 절차)를 보안하기 위한 작업은 거의 수행되지 않습

니다. 일반적으로 대부분의 시스템이 회사 내부에 있기 때문에 내부적으로는 더 많은 리소스가 있습니다. 회사 외부에 있게 되면 귀하의 상태를 신뢰할 수 없습니다. 외부에서 사용할 수 있는 시스템과 리소스는 일반적으로 매우 제한적입니다.

취약점 평가와 *침투 테스트* 간의 차이점을 고려하십시오. 취약점 평가를 침투 테스트의 첫 단계로 생각하십시오. 평가에서 얻은 정보는 테스트에 사용됩니다. 평가는 허점과 잠재적인 취약점을 확인하기 위해 수행되는 반면, 침투 테스트는 실제로 결과를 악용하려고 시도합니다.

네트워크 인프라 평가는 동적 프로세스입니다. 보안 정보와 물리적 보안은 동적입니다. 평가를 수행하면 개요가 명확 해지고 오류 감지 (False positives) 및 감지 누출 (False negatives)이 표시 될 수 있습니다 잘못된 긍정적인 결과는 툴이 실제로 존재하지 않는 취약점을 찾아내는 결과입니다. 잘못된 음수는 실제 취약점을 생략하는 것입니다.

보안 관리자의 역량은 사용하는 툴과 관리자가 보유한 지식에 따라 다릅니다. 현재 사용 가능한 평가 도구 중 하나를 사용하여 시스템에 대해 실행하십시오. 이는 거의 잘못된 선택 사항이 있다는 것을 보장할 수 있습니다. 프로그램 폴트 또는 사용자 오류로 결과가 동일합니다. 도구는 오탐을 찾을 수도 있고, 더 심각한 경우 잘못된 음수를 찾을 수 있습니다.

취약점 평가와 침투 테스트의 차이점이 정의되었으므로, 평가 결과를 신중하게 검토하고 새로운 모범 사례 접근법의 일부로 침투 테스트를 수행하십시오.

주의

프로덕션 시스템에서 취약점을 악용하지 마십시오. 이렇게 하면 시스템 및 네트워크의 생산성과 효율성에 부정적인 영향을 미칠 수 있습니다.

다음 목록에서는 취약점 평가를 수행하는 몇 가지 이점을 소개합니다.

- 능동적으로 정보 보안에 중점을 둘 수 있습니다.
- 공격자가 발견하기 전에 잠재적인 취약점을 찾을 수 있습니다.
- 시스템을 최신 상태로 유지하고 패치를 적용할 수 있습니다.
- 직원의 성장 및 전문성 개발을 지원합니다.
- 경제적 손실과 부정적 대중 이미를 줄일 수 있습니다.

1.3.2. 취약점 평가를 위한 방법 설정

취약점 평가 툴을 선택하는 데 도움이 되는 것은 취약점 평가 방법론을 설정하는 데 도움이 됩니다. 하지만 지금은 사전 정의된 업계 승인 방법론이 없지만 일반적인 의미와 모범 사례가 충분한 가이드로 사용될 수 있습니다.

대상이란 무엇입니까? 하나의 서버를 보고 있습니까? 아니면 전체 네트워크와 네트워크 내의 모든 것을 보고 있습니까? 회사 외부입니까? 아니면 회사 내부입니까? 이러한 질문에 대한 대답은 어떤 툴을 선택할지 여부와 이를 사용하는 방식을 결정하는 데 도움이 되기 때문에 중요합니다.

방법론 설정에 대한 자세한 내용은 다음 웹 사이트를 참조하십시오.

- https://www.owasp.org/ – *The Open Web Application Security Project*

1.3.3. 취약점 평가 툴

평가는 일부 형태의 정보 수집 툴을 사용하여 시작할 수 있습니다. 전체 네트워크를 평가할 때 레이아웃을 먼저 매핑하여 실행 중인 호스트를 찾습니다. 호스트의 위치를 확인한 후 각 호스트를 개별적으로 검사합니다. 이러한 호스트에 중점을 두려면 다른 툴 세트가 필요합니다. 사용할 툴을 아는 것은 취약점을 찾는 데 가장 중요한 단계일 수 있습니다.

일상 생활의 어떤 측면과 마찬가지로 동일한 작업을 수행하는 다양한 툴이 있습니다. 이 개념은 취약점 평가를 수행하는 데도 적용됩니다. 운영 체제, 애플리케이션 및 네트워크 고유의 툴이 있습니다(사용되는 프로토콜 기반). 일부 도구는 무료이며 다른 도구는 무료가 아닙니다. 일부 도구는 직관적이고 사용하기 쉽지만 다른 툴은 암호 해독 및 문서화되어 있지만 다른 툴에는 그렇지 않은 기능이 있습니다.

올바른 도구를 찾는 것은 어려운 작업이 될 수 있으며, 결국 경험 수 있습니다. 가능한 경우 테스트 랩을 설정하고 가능한 한 많은 툴을 시도하여 각각의 강점과 약점을 파악하십시오. 툴의 **README** 파일 또는 man 페이지를 검토합니다. 또한, 문서, 단계별 가이드 또는 도구와 관련된 메일링 목록과 같은 자세한 정보는 인터넷을 참조하십시오.

아래에서 설명하는 툴은 사용 가능한 도구의 작은 샘플링일 뿐입니다.

1.3.3.1. Nmap을 사용하여 호스트 검색

Nmap은 네트워크 레이아웃을 결정하는 데 널리 사용되는 툴입니다. **Nmap**은 수년 동안 사용할 수 있으며 정보를 수집할 때 가장 자주 사용되는 툴입니다. 옵션 및 사용에 대한 자세한 설명을 제공하는 우수한 도움말 페이지가 포함되어 있습니다. 관리자는 네트워크에서 **Nmap**을 사용하여 호스트 시스템을 찾고 해당 시스템에서 포트를 열 수 있습니다.

Nmap은 취약점 평가의 첫 번째 단계입니다. 네트워크 내의 모든 호스트를 매핑하고 **Nmap** 이 특정 호스트에서 실행되는 운영 체제를 식별할 수 있도록 하는 옵션을 전달할 수 있습니다. **Nmap**은 보안 서비스를 사용하고 사용되지 않는 서비스를 제한하는 정책을 수립하기 위한 좋은 기반입니다.

Nmap을 설치하려면 **yum install nmap** 명령을 **root** 사용자로 실행합니다.

1.3.3.1.1. Nmap 사용

Nmap은 **nmap** 명령 뒤에 검사할 머신의 호스트 이름 또는 **IP** 주소를 입력하여 쉘 프롬프트에서 실행할 수 있습니다.

```
nmap <hostname>
```

예를 들어 호스트 이름이 **foo.example.com**인 머신을 스캔하려면 쉘 프롬프트에 다음을 입력합니다.

```
~]$ nmap foo.example.com
```

기본 검사의 결과(호스트가 있는 위치와 기타 네트워크 조건이 있는 위치에 따라 몇 분 정도 걸릴 수 있음)는 다음과 유사합니다.

```
Interesting ports on foo.example.com:
Not shown: 1710 filtered ports
PORT    STATE  SERVICE
22/tcp  open   ssh
53/tcp  open   domain
80/tcp  open   http
113/tcp closed auth
```

Nmap은 수신 대기 또는 대기 서비스에 대한 가장 일반적인 네트워크 통신 포트를 테스트합니다. 이 지식이 필요하지 않거나 사용하지 않는 서비스를 종료하려는 관리자에게 유용할 수 있습니다.

Nmap 사용에 대한 자세한 내용은 다음 URL에서 공식 홈페이지를 참조하십시오.

http://www.insecure.org/

1.3.3.2. Nessus

Nessus는 전체 서비스 보안 스캐너입니다. **Nessus**의 플러그인 아키텍처를 통해 사용자는 시스템 및 네트워크에 맞게 사용자 지정할 수 있습니다. 모든 스캐너와 마찬가지로 **Nessus**는 의존하는 서명 데이터베이스만큼 우수합니다. 다행히도 **Nessus** 는 자주 업데이트되며 전체 보고, 호스트 스캔 및 실시간 취약성 검색 기능을 제공합니다. 도구에서 강력하고 **Nessus** 로서 자주 업데이트되는 경우도 거짓 긍정과 거짓 부정이 있을 수 있습니다.

참고

Nessus 클라이언트 및 서버 소프트웨어를 사용하려면 서브스크립션이 필요합니다. 이 문서에는 이 인기 있는 애플리케이션을 사용하는 데 관심이 있는 사용자에 대한 참조로 포함되어 있습니다.

Nessus에 대한 자세한 내용은 다음 URL에서 공식 웹 사이트를 참조하십시오.

http://www.nessus.org/

1.3.3.3. OpenVAS

OpenVAS (*Open Vulnerability Assessment System*)는 취약점과 포괄적인 취약점 관리를 검사하는 데 사용할 수 있는 일련의 툴 및 서비스입니다. **OpenVAS** 프레임워크는 솔루션의 다양한 구성 요소를 제어하기 위한 다양한 웹 기반, 데스크탑 및 명령줄 툴을 제공합니다. **OpenVAS** 의 핵심 기능은 보안 스캐너를 통해 제공되며, 이 스캐너는 33000개 이상의 일일 업데이트 네트워크 취약점 테스트(NVT)를 사용합니다. **Nessus** (1.3.3.2절. "**Nessus**"참조)와 달리 **OpenVAS** 에는 서브스크립션이 필요하지 않습니다.

OpenVAS에 대한 자세한 내용은 다음 URL에서 공식 웹 사이트를 참조하십시오.

http://www.openvas.org/

1.3.3.4. Nikto

Nikto 는 뛰어난 *일반 게이트웨이 인터페이스* (CGI) 스크립트 스캐너입니다. **Nikto**는 CGI 취약점을 확인할 뿐만 아니라 광범위한 방식으로 침입을 탐지하기 위해 이를 수행합니다. 이 프로그램은 프로그램을 실행하기 전에 신중하게 검토해야하는 철저한 문서와 함께 제공됩니다. CGI 스크립트를 제공하는 웹 서버가 있는 경우 **Nikto**는 이러한 서버의 보안을 확인하는 데 유용한 리소스가 될 수 있습니다.

Nikto에 대한 자세한 내용은 다음 URL에서 확인할 수 있습니다.

http://cirt.net/nikto2

1.4. 보안 위협

1.4.1. 네트워크 보안에 대한 위협

네트워크의 다음 측면을 구성할 때 잘못된 사례로 인해 공격 위험이 증가할 수 있습니다.

비보안 아키텍처
잘못 구성된 네트워크는 권한이 없는 사용자의 주요 진입점이 됩니다. 신뢰도가 높은 오픈 로컬 네트워크를 보안성이 높은 인터넷에 취약하게 두는 것은 마치 암묵적인 취약성에 문을 열어 두는 것과 비슷합니다. 임의 시간 동안 아무 일도 발생하지는 않지만, *결국* 이 기회를 악용합니다.

브로드캐스트 네트워크
시스템 관리자는 보안 체계에서 네트워킹 하드웨어의 중요성을 인식하지 못하는 경우가 많습니다. 허브 및 라우터와 같은 단순한 하드웨어는 브로드캐스트 또는 비스위칭 원칙을 사용합니다. 즉, 노드에서 네트워크를 통해 데이터를 수신자 노드로 전송할 때마다 허브 또는 라우터는 수신자 노드가 데이터를 수신하고 처리할 때까지 데이터 패킷의 브로드캐스트를 전송합니다. 이 방법은 로컬 호스트의 외부 침입자 및 권한 없는 사용자에 의해*ARP*(Address Resolution Protocol) 또는*MAC*(Media Access Control) 주소 스푸핑에 가장 취약합니다.

중앙 집중식 서버
또 다른 잠재적인 네트워킹 위험은 중앙 집중식 컴퓨팅을 사용하는 것입니다. 많은 기업에서 일반적인 비용 절감 방법은 모든 서비스를 하나의 강력한 시스템에 통합하는 것입니다. 이 기능은 여러 서버 구성보다 관리 및 비용이 훨씬 적기 때문에 편리합니다. 그러나 중앙 집중식 서버에서는 네트워크에서 단일 장애 지점을 발생시킵니다. 중앙 서버가 손상되면 네트워크가 완전히 사용되지 않거나 저하될 수 있으며 데이터 조작이나 도난이 발생할 수 있습니다. 이러한 상황에서 중앙 서버는 전체 네트워크에 액세스할 수 있는 열린 문이 됩니다.

1.4.2. 서버 보안에 대한 위협

서버 보안은 서버에서 종종 조직의 중요한 정보를 보유하고 있기 때문에 네트워크 보안만큼 중요합니다. 서버가 손상되면 공격자가 수정하거나 조작할 수 있도록 모든 콘텐츠를 사용할 수 있습니다. 다음 섹션에서는 몇 가지 주요 문제에 대해 자세히 설명합니다.

사용되지 않는 서비스 및 오픈 포트
Red Hat Enterprise Linux 7의 전체 설치에는 1000개 이상의 애플리케이션 및 라이브러리 패키지가 포함되어 있습니다. 그러나 대부분의 서버 관리자는 배포판에 모든 패키지를 설치하지 않고 여러 서버 애플리케이션을 포함하여 기본 패키지 설치를 선호합니다. 설치된 패키지 수와 추가 리소스의 수를 제한해야 하는 이유에 대한 설명은 2.3절. “필요한 최소 패키지 설치”를 참조하십시오.

시스템 관리자 사이에서 흔히 발생하는 현상은 어떤 프로그램이 실제로 설치되고 있는지 확인하지 않고 운영 체제를 설치하는 것입니다. 이는 불필요한 서비스를 설치하고, 기본 설정으로 구성되고, 켜질 수 있기 때문에 문제가 될 수 있습니다. 이로 인해 Telnet, DHCP 또는 DNS와 같은 원하지 않는 서비스가 관리자가 인식하지 않고 서버 또는 워크스테이션에서 실행되도록 할 수 있습니다. 이로 인해 서버에 대한 원하지 않는 트래픽이나 공격자를 위한 시스템로의 잠재적인 경로도 발생할 수 있습니다. 닫기 포트 및 사용되지 않는 서비스 비활성화에 대한 정보는 4.3절. “서비스 보안”를 참조하십시오.

패치되지 않은 서비스
기본 설치에 포함된 대부분의 서버 애플리케이션은 철저하게 테스트된 소프트웨어입니다. 수년 동안 프로덕션 환경에서 사용하던 코드는 철저하게 수정되었으며 많은 버그를 찾아 수정했습니다.

그러나 완벽한 소프트웨어는 없으며 항상 더 개선할 여지가 있습니다. 더욱이 최신 소프트웨어는 최근 프로덕션 환경에 도착했거나 다른 서버 소프트웨어만큼 인기가 없기 때문에 예상만큼 엄격하게 테스트되지 않는 경우가 많습니다.

개발자와 시스템 관리자는 서버 애플리케이션에서 악용할 수 있는 버그를 찾고 버그 추적 및 보안 관련 웹사이트(예: Bugtraq 메일링 리스트(http://www.cert.org) 또는 암호 긴급 대응 팀(http://www.cert.org)에 대한 정보를 게시합니다. 이러한 메커니즘이 커뮤니티에 보안 취약점을 경고하는 효과적인 방법이지만 시스템 관리자가 시스템을 즉시 패치하는 것이 중요합니다. 크래커는 이러한 취약점 추적 서비스에 액세스할 수 있고 정보를 사용하여 할 수 있을 때마다 패치되지 않은 시스템을 공격하기 때문에 특히 그렇습니다. 우수한 시스템 관리를 위해서는 보다 안전한 컴퓨팅 환경을 보장하기 위해 경계, 지속적인 버그 추적, 적절한 시스템 유지 관리가 필요합니다.

시스템을 최신 상태로 유지하는 방법에 대한 자세한 내용은 3장. *시스템을 최신 상태로 유지*를 참조하십시오.

편집할 수 없음
관리자가 시스템을 패치하지 않는 것은 서버 보안에 대한 가장 큰 위협 중 하나입니다. *SysAdmin, Audit, Network, Security Institute* (*SANS*)에 따르면 컴퓨터 보안 취약점의 주요 원인은 "보안 유지를 위해 규제되지 않은 사람들을 할당하고 교육이나 작업을 배울 수 있도록 할 시간이나 작업을 수행할 수 있도록 할 시간"입니다."[1] 이는 관리자의 경험 미숙과 관리자의 과신과 동기 부여의 낮음 등도 원인이 됩니다.

일부 관리자는 서버와 워크스테이션을 패치하지 못하지만 다른 관리자는 시스템 커널 또는 네트워크 트래픽에서 로그 메시지를 감시하지 못합니다. 또 다른 일반적인 오류는 서비스에 대한 기본 암호나 키가 변경되지 않은 경우입니다. 예를 들어 데이터베이스 개발자는 시스템 관리자가 설치 후 즉시 이러한 암호를 변경할 것이라고 가정하므로 일부 데이터베이스에는 기본 관리 암호가 있습니다. 데이터베이스 관리자가 이 암호를 변경하지 못하는 경우 경험이 부족한 공격자도 잘 알려진 기본 암호를 사용하여 데이터베이스의 관리 권한을 얻을 수 있습니다. 이는 의도하지 않은 관리로 인해 서버가 손상될 수 있음을 보여주는 몇 가지 예에 불과합니다.

본질적으로 비보안 서비스
자신이 선택하는 네트워크 서비스가 본질적으로 안전하지 않을 경우 가장 취약한 조직도 취약점으로 훼손될 수 있습니다. 예를 들어, 신뢰할 수 있는 네트워크에서 사용한다고 가정하여 개발된 많은 서비스가 있지만, 이러한 가정은 본질적으로 신뢰할 수 없는 인터넷을 통해 서비스를 사용할 수 있게 되는 즉시 실패합니다.

안전하지 않은 네트워크 서비스의 한 카테고리는 인증을 위해 암호화되지 않은 사용자 이름과 암호가 필요한 것입니다. Telnet과 FTP는 두 가지 서비스입니다. 패킷이 원격 사용자와 이러한 서비스 사용자 이름과 암호 간의 트래픽을 모니터링하는 경우 쉽게 인터셉트할 수 있습니다.

본질적으로 이러한 서비스는 보안 업계에서 말하는 *중간자* 공격의 희생양이 되기 쉽습니다. 이러한 유형의 공격에서는 크래커가 의도한 서버 대신 자신의 시스템을 가리키도록 네트워크의 이름 서버를 속임으로써 네트워크 트래픽을 리디렉션합니다. 다른 사용자가 서버에 대한 원격 세션을 열면 공격자의 시스템이 보이지 않는 탐정 역할을 하여 원격 서비스와 잘못된 사용자 캡처 정보 사이에 침묵합니다. 이러한 방식으로 크래커는 서버나 사용자가 인식하지 않고도 관리 암호 및 원시 데이터를 수집할 수 있습니다.

또 다른 종류의 비보안 서비스에는 LAN 사용을 위해 명시적으로 개발되지만 안타깝게도 WAN(원격 사용자의 경우)으로 확장되는 NFS 또는 NIS와 같은 네트워크 파일 시스템 및 정보 서비스가 포함됩니다. 기본적으로 NFS에는 크래커가 NFS 공유를 마운트하고 여기에 포함된 모든 항목에 액세스하지 못하도록 인증 또는 보안 메커니즘이 구성되어 있지 않습니다. NIS에도 일반 텍스트 ASCII 또는 DBM(ASCII 파생) 데이터베이스 내에서 암호 및 파일 권한을 포함하여 네트워크의 모든 컴퓨터에서 알려야 하는 중요한 정보가 있습니다. 이 데이터베이스에 액세스할 수 있는 크래커는 관리자 계정을 포함하여 네트워크의 모든 사용자 계정에 액세스할 수 있습니다.

기본적으로 Red Hat Enterprise Linux 7은 이러한 모든 서비스가 꺼진 상태로 릴리스됩니다. 그러나 관리자가 이러한 서비스를 사용해야 하는 경우가 많으므로 신중하게 구성해야 합니다. 안전한 방식으로 서비스 설정에 대한 자세한 내용은 4.3절. “서비스 보안”를 참조하십시오.

1.4.3. 워크스테이션 및 홈 PC 보안에 대한 위협

워크스테이션 및 홈 PC는 네트워크 또는 서버로서 공격을하기 쉽지 않을 수 있지만 종종 중요한 데이터(예: 신용 카드 정보)를 포함하기 때문에 시스템 크래커의 대상이 됩니다. 워크스테이션은 또한 사용자의 지식 없이도 공동 선택할 수 있으며 공격자가 조정된 공격의 "슬레이브" 시스템으로 사용할 수 있습니다. 이러한 이유로 워크스테이션의 취약점을 파악하면 사용자에게 운영 체제를 다시 설치하거나 데이터 도난으로부터 복구하는 번거로움을 줄일 수 있습니다.

잘못된 암호
잘못된 암호는 공격자가 시스템에 액세스할 수 있는 가장 쉬운 방법 중 하나입니다. 암호를 만들 때 일반적인 문제가 발생하지 않도록 하는 방법에 대한 자세한 내용은 4.1.1절. “암호 보안” 을 참조하십시오.

취약한 클라이언트 애플리케이션
관리자가 완전히 안전하고 패치가 적용된 서버를 갖고 있다고 해도 원격 사용자가 서버에 액세스할 때 안전하다는 의미는 아닙니다. 예를 들어 서버가 공용 네트워크를 통해 Telnet 또는 FTP 서비스를 제공하는 경우 공격자는 네트워크를 통해 전달되는 일반 텍스트 사용자 이름과 암호를 캡처한 다음 계정 정보를 사용하여 원격 사용자의 워크스테이션에 액세스할 수 있습니다.

SSH와 같은 보안 프로토콜을 사용하는 경우에도 클라이언트 애플리케이션을 계속 업데이트하지 않으면 원격 사용자가 특정 공격에 취약해질 수 있습니다. 예를 들어 v.1 SSH 클라이언트는 악성 SSH 서버의 X 전달 공격에 취약합니다. 서버에 연결되면 공격자는 네트워크를 통해 클라이언트가 만든 키 입력 및 마우스 클릭을 자동으로 캡처할 수 있습니다. 이 문제는 v.2 SSH 프로토콜에서 해결되었지만 필요한 취약점이 있는 애플리케이션을 추적하고 필요에 따라 업데이트하는 것은 사용자에게 달려 있습니다.

4.1절. "데스크탑 보안" 관리자와 홈 사용자가 컴퓨터 워크스테이션의 취약점을 제한하기 위해 취해야 하는 단계를 자세히 설명합니다.

1.5. 일반적인 EXPLOITS 및 ATTACKS

표 1.1. "일반적인 위협" 조직 네트워크 리소스에 액세스하기 위해 침입자가 사용하는 가장 일반적인 공격 및 진입점에 대해 자세히 설명합니다. 이러한 일반적인 위협의 핵심은 수행 방법 및 관리자가 이러한 공격으로부터 네트워크를 적절히 보호하는 방법에 대한 설명입니다.

표 **1.1.** 일반적인 위협

악용	설명	참고
null 또는 기본 암호	관리 암호를 비워 두거나 제품 벤더가 설정한 기본 암호를 사용합니다. 라우터 및 방화벽과 같은 하드웨어에서 가장 일반적이지만 Linux에서 실행되는 일부 서비스에는 기본 관리자 암호도 포함될 수 있습니다(Red Hat Enterprise Linux 7에서는 제공되지 않음).	일반적으로 라우터, 방화벽, VPN 및 NAS(네트워크 연결 스토리지) 어플라이언스와 같은 네트워킹 하드웨어와 관련이 있습니다. 많은 기존 운영 체제, 특히 번들 서비스(예: UNIX 및 Windows)에서 일반적으로 사용됩니다. 관리자는 계정을 검색하는 악의적인 사용자에 대한 완벽한 진입점을 만들어 관리자가 권한 있는 사용자 계정을 신속하게 생성하고 암호를 null로 남겨두는 경우가 있습니다.
기본 공유 키	보안 서비스는 개발 또는 평가 테스트를 위해 기본 보안 키를 패키징하는 경우가 있습니다. 이러한 키가 변경되지 않고 인터넷의 운영 환경에 배치되면 기본 키가 동일한 모든 사용자와 해당 공유 키 리소스에 대한 액세스 권한 및 해당 키에 포함된 중요한 정보에 액세스할 수 있습니다.	무선 액세스 지점과 사전 구성된 보안 서버 어플라이언스에서 가장 일반적입니다.

악용	설명	참고
IP 스푸핑	원격 시스템은 로컬 네트워크의 노드 역할을 하며, 서버의 취약점을 발견하고, 백도어 프로그램 또는 트로이 목마를 설치하여 네트워크 리소스를 제어합니다.	스푸핑은 공격자가 대상 시스템에 대한 연결을 조정하기 위해 TCP/IP 시퀀스 번호를 예측하지만 크래커가 이러한 취약점을 수행하는 데 도움이 되는 몇 가지 도구를 사용할 수 있기 때문에 스푸핑은 매우 어렵습니다. PKI 또는 **ssh** 또는 SSL/TLS에 사용된 암호화된 인증과 비교하면 권장되지 않는, **rsh**,**telnet**, FTP 등과 같은 대상 시스템(예: rsh , telnet , FTP 등)에 따라 다릅니다.
도청	두 노드 간 연결을 도청하여 네트워크에서 두 개의 활성 노드 간에 전달되는 데이터를 수집합니다.	이러한 유형의 공격은 대부분 Telnet, FTP 및 HTTP 전송과 같은 일반 텍스트 전송 프로토콜에서 작동합니다. 원격 공격자는 이러한 공격을 수행하기 위해 LAN에서 손상된 시스템에 액세스할 수 있어야 합니다. 일반적으로 크래커는 시스템을 LAN에 손상시키기 위해 액티브 공격(예: IP 스푸핑 또는 중간자)을 사용했습니다. 예방 조치에는 키 교환이 포함된 서비스, 일회성 암호 또는 암호 스누핑을 방지하기 위한 암호화된 인증이 포함됩니다. 전송 중 강력한 암호화도 권장합니다.
서비스 취약점	공격자는 인터넷을 통해 실행되는 서비스에서 결함이나 허점이 발견됩니다. 이 취약점을 통해 공격자는 전체 시스템과 데이터가 손상되고 네트워크에 있는 다른 시스템을 손상시킬 수 있습니다.	CGI와 같은 HTTP 기반 서비스는 원격 명령 실행 및 대화형 쉘 액세스에 취약합니다. HTTP 서비스가 "nobody"와 같은 권한이 없는 사용자로 실행되는 경우에도 구성 파일 및 네트워크 맵과 같은 정보를 읽을 수 있고 공격자가 시스템 리소스를 드레이닝하거나 다른 사용자가 사용할 수 없게 되는 서비스 거부를 시작할 수 있습니다. 이러한 취약점(예: 버퍼 오버플로우와 같은 *버퍼 오버플로)이 애플리케이션 메모리 버퍼* 를 사용하여 서비스를 중단하여 공격자에게 임의의 명령을 실행할 수 있는 대화형 명령 프롬프트를 제공하여 공격자에게 완전한 관리 제어 권한을 부여할 수 있습니다. 관리자는 서비스가 루트 사용자로 실행되지 않도록 해야 하며 벤더 또는 보안 조직의 애플리케이션이나 CERT 및 CVE 등의 애플리케이션에 대한 패치 및 에라타 업데이트를 유지해야 합니다.

악용	설명	참고
애플리케이션 취약점	공격자는 데스크탑 및 워크스테이션 애플리케이션(예: 이메일 클라이언트)에서 오류를 찾고 임의의 코드를 실행하거나, 향후 손상 또는 크래시 시스템을 위해 트로이 목마를 제거합니다. 손상된 워크스테이션에 네트워크의 나머지 부분에서 관리 권한이 있는 경우 추가 공격이 발생할 수 있습니다.	워크 스테이션과 데스크탑은 악용하기가 더 쉽습니다. 작업자는 타협을 방지하거나 탐지하기 위한 전문 지식이나 경험이 없으므로 악용하기 쉽습니다. 무단 소프트웨어를 설치하거나 원하지 않는 이메일 첨부 파일을 열 때 발생할 위험이 있음을 알려주는 것이 중요합니다. 이메일 클라이언트 소프트웨어가 첨부 파일을 자동으로 열거나 실행하지 않도록 보호 장치를 구현할 수 있습니다. 또한 Red Hat Network 또는 기타 시스템 관리 서비스를 사용하여 워크스테이션 소프트웨어의 자동 업데이트로 다중 기술 보안 배포의 부담을 덜 수 있습니다.
서비스 거부(DoS) 공격	공격자 또는 공격자 그룹이 조직의 네트워크 또는 서버 리소스에 대해 대상 호스트(서버, 라우터 또는 워크스테이션)로 무단 패킷을 전송하여 조정됩니다. 이렇게 하면 합법적인 사용자가 리소스를 사용할 수 없게 됩니다.	미국에서 가장 보고된 DoS 사례는 2000년에 발생했습니다. 트래픽이 많은 상용 및 정부 사이트에서는 조정된 ping 플러드 공격으로 인해 *좀비* 역할을 하는 대역폭이 높은 여러 시스템을 사용하거나 브로드캐스팅 노드를 리디렉션할 수 없게 되었습니다. 소스 패킷은 일반적으로 위조되고 리브로드캐스트되므로 공격의 실제 소스에 대해서는 조사가 어렵습니다. Snort와 같은 **iptables** 및 네트워크 침입 탐지 시스템을 사용하여 수신 필터링 (IETF rfc2267)의 발전은 관리자가 분산 DoS 공격을 추적하고 방지하는 데 도움이 됩니다.

[1] http://www.sans.org/security-resources/mistakes.php

2장. 설치를 위한 보안 팁

Red Hat Enterprise Linux 7을 설치하기 위해 CD 또는 DVD를 디스크 드라이브에 처음 배치한 상태에서 보안이 시작됩니다. 처음부터 안전하게 시스템을 구성하면 나중에 추가 보안 설정을 더 쉽게 구현할 수 있습니다.

2.1. BIOS 보안

BIOS(또는 이에 상응하는 BIOS) 및 부트 로더에 대한 암호 보호는 시스템에 대한 물리적 액세스 권한이 없는 사용자가 이동식 미디어를 사용하여 부팅하거나 단일 사용자 모드를 통해 root 권한을 얻는 것을 방지할 수 있습니다. 이러한 공격으로부터 보호하기 위해 수행해야 하는 보안 조치는 워크스테이션에 있는 정보의 민감도와 시스템의 위치에 따라 달라집니다.

예를 들어, 시스템이 무역 박람회에 사용되며 중요한 정보가 포함되지 않은 경우 이러한 공격을 방지하는 것이 중요하지 않을 수 있습니다. 그러나 개인으로 인해 회사 네트워크에 대해 암호화되지 않은 SSH 키가 있는 직원의 랩탑이 동일한 거래 쇼에서 역순으로 남아 있는 경우 회사 전체의 RAM으로 인한 주요 보안 위반으로 이어질 수 있습니다.

반면에 권한이 있거나 신뢰할 수 있는 사람만 액세스할 수 있는 장소에 워크스테이션이 있는 경우 BIOS 또는 부트 로더 보안이 필요하지 않을 수 있습니다.

2.1.1. BIOS 암호

컴퓨터의 BIOS를 암호로 보호하는 두 가지 주요 이유는 다음과 같습니다.[2]:

1. *BIOS 설정 변경 사항 방지* - 침입자가 BIOS에 액세스할 수 있는 경우 CD-ROM 또는 플래쉬 드라이브에서 부팅하도록 설정할 수 있습니다. 이를 통해 복구 모드 또는 단일 사용자 모드로 전환할 수 있으므로 시스템에서 임의의 프로세스를 시작하거나 중요한 데이터를 복사할 수 있습니다.

2. *시스템 부팅 방지* - 일부 BIOS는 부팅 프로세스의 암호를 보호합니다. 활성화되면 공격자는 BIOS가 부트 로더를 시작하기 전에 암호를 입력해야 합니다.

BIOS 암호를 설정하는 방법은 컴퓨터 제조업체마다 다르기 때문에 특정 지침은 컴퓨터 설명서를 참조하십시오.

BIOS 암호를 잊어버린 경우 마더보드의 점퍼를 사용하거나 CMOS 배터리의 연결을 해제하여 재설정할 수 있습니다. 따라서 가능한 경우 컴퓨터 케이스를 잠그는 것이 좋습니다. 그러나 CMOS 배터리의 연결을 해제하기 전에 컴퓨터 또는 마더보드에 대한 설명서를 참조하십시오.

2.1.1.1. 비BIOS 기반 시스템 보안

다른 시스템 및 아키텍처는 서로 다른 프로그램을 사용하여 x86 시스템의 BIOS와 거의 동일한 수준의 작업을 수행합니다. 예를 들어*UEFI*(*Unified Extensible Firmware Interface*) 쉘이 있습니다.

BIOS와 같은 프로그램을 보호하는 암호에 대한 지침은 제조업체의 지침을 참조하십시오.

2.2. 디스크 파티션

Red Hat은**/boot**, /, **/home**, **/tmp**, **/var/tmp**/ 디렉토리에 대해 별도의 파티션을 만드는 것을 권장합니다. 각각의 이유는 다르고, 각 파티션을 다룹니다.

/boot

이 파티션은 부팅 중에 시스템에서 읽은 첫 번째 파티션입니다. 시스템을 Red Hat Enterprise Linux 7로 부팅하는 데 사용되는 부트 로더 및 커널 이미지는 이 파티션에 저장됩니다. 이 파티션은 암호화해서는

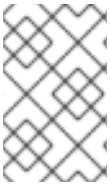
안 됩니다. 이 파티션이 /에 포함되고 해당 파티션이 암호화되거나 기타를 사용할 수 없게 되면 시스템이 부팅되지 않습니다.

/home

사용자 데이터(**/home**)가 별도의 파티션 대신 /에 저장되어 파티션이 채워지면 운영 체제가 불안정해집니다. 또한 시스템을 Red Hat Enterprise Linux 8의 다음 버전으로 업그레이드할 때 데이터를 **/home** 파티션에 덮어쓰지 않으므로 업그레이드가 더 쉽습니다. 루트 파티션(/)이 손상되면 데이터가 영구적으로 손실될 수 있습니다. 별도의 파티션을 사용하면 데이터 손실을 조금 더 완화할 수 있습니다. 이 파티션을 빈번한 백업의 대상으로 지정할 수도 있습니다.

/tmp 및 **/var/tmp/**

/tmp 및 **/var/tmp/** 디렉터리는 모두 장기간 저장하지 않아도 되는 데이터를 저장하는 데 사용됩니다. 그러나 이러한 디렉토리 중 하나에 많은 데이터가 범람하는 경우 모든 스토리지 공간을 소비할 수 있습니다. 이 경우 이러한 디렉토리가 / 내에 저장되면 시스템이 불안정해 충돌할 수 있습니다. 따라서 이러한 디렉터리를 해당 파티션으로 이동하는 것이 좋습니다.



참고

설치 프로세스 중에 파티션을 암호화할 수 있는 옵션이 있습니다. 암호를 제공해야 합니다. 이 암호는 파티션의 데이터를 보호하는 데 사용되는 대량 암호화 키의 잠금을 해제하는 키 역할을 합니다. 자세한 내용은 4.9.1절. "LUKS 디스크 암호화 사용"의 내용을 참조하십시오.

2.3. 필요한 최소 패키지 설치

컴퓨터에 있는 각 소프트웨어에 취약점이 있을 수 있으므로 사용할 패키지만 설치하는 것이 좋습니다. DVD 미디어에서 설치하는 경우 설치 중에 설치할 패키지를 정확하게 선택할 수 있습니다. 다른 패키지가 필요한 경우 나중에 시스템에 항상 추가할 수 있습니다.

최소 설치 환경 설치에 대한 자세한 내용은 Red Hat Enterprise Linux 7 설치 가이드의 소프트웨어 선택 장을 참조하십시오. **--nobase** 옵션을 사용하여 Kickstart 파일에서 최소 설치를 수행할 수도 있습니다. Kickstart 설치에 대한 자세한 내용은 Red Hat Enterprise Linux 7 설치 가이드의 패키지 선택 섹션을 참조하십시오.

2.4. 설치 프로세스 중 네트워크 연결 제한

Red Hat Enterprise Linux를 설치할 때 설치 매체는 특정 시간에 시스템의 스냅샷을 나타냅니다. 이로 인해 최신 보안 수정 사항이 최신 상태가 아닐 수 있으며 설치 미디어에서 제공한 시스템이 릴리스된 후에만 수정된 특정 문제에 취약해질 수 있습니다.

잠재적으로 취약한 운영 체제를 설치하는 경우 항상 가장 필요한 네트워크 영역으로만 노출을 제한합니다. 가장 안전한 선택은 "네트워크 없음" 영역으로, 설치 프로세스 중에 시스템의 연결이 끊어진 상태로 두는 것을 의미합니다. 인터넷 연결이 가장 위험한 경우에는 LAN 또는 인트라넷 연결만으로도 충분합니다. 최상의 보안 사례를 따르려면 네트워크에서 Red Hat Enterprise Linux를 설치하는 동안 리포지토리와 가장 가까운 영역을 선택합니다.

네트워크 연결 구성에 대한 자세한 내용은 Red Hat Enterprise Linux 7 설치 가이드의 네트워크 및 호스트 이름 장을 참조하십시오.

2.5. 설치 후 작업

다음 단계는 Red Hat Enterprise Linux 설치 후 즉시 수행해야 하는 보안 관련 절차입니다.

1. 시스템을 업데이트합니다. root로 다음 명령을 입력합니다.

```
~]# yum update
```

2. 방화벽 서비스 **firewalld**는 Red Hat Enterprise Linux 설치를 통해 자동으로 활성화되지만 Kickstart 구성과 같이 명시적으로 비활성화할 수 있습니다. 이러한 경우 방화벽을 다시 활성화하는 것이 좋습니다.

   firewalld를 시작하려면 root로 다음 명령을 입력합니다.

```
~]# systemctl start firewalld
~]# systemctl enable firewalld
```

3. 보안을 강화하려면 필요하지 않은 서비스를 비활성화합니다. 예를 들어 컴퓨터에 프린터가 설치되어 있지 않은 경우 다음 명령을 사용하여 **cups** 서비스를 비활성화합니다.

```
~]# systemctl disable cups
```

   활성 서비스를 검토하려면 다음 명령을 입력합니다.

```
~]$ systemctl list-units | grep service
```

2.6. 추가 리소스

일반적인 설치에 대한 자세한 내용은 Red Hat Enterprise Linux 7 설치 가이드를 참조하십시오 .

---

[2] 시스템 BIOS는 제조업체마다 다르므로 일부 유형은 두 유형의 암호 보호를 지원하지 않을 수 있지만 다른 유형은 두 유형 중 하나를 지원하지만 다른 하나는 지원하지 않을 수 있습니다.

3장. 시스템을 최신 상태로 유지

이 장에서는 시스템을 최신 상태로 유지하는 프로세스를 설명합니다. 여기에는 보안 업데이트 설치 방식을 계획 및 구성하고, 새로 업데이트된 패키지에서 도입된 변경 사항을 적용하고, Red Hat 고객 포털을 사용하여 보안 권고를 추적할 수 있습니다.

3.1. 설치된 소프트웨어 유지 관리

보안 취약점이 발견되면 잠재적인 보안 위험을 제한하려면 영향을 받는 소프트웨어를 업데이트해야 합니다. Red Hat은 현재 지원되는 Red Hat Enterprise Linux 배포판 내 패키지의 일부인 경우 최대한 빨리 취약점을 수정하는 업데이트된 패키지를 출시하기 위해 최선을 다하고 있습니다.

종종 특정 보안 공격에 대한 발표는 문제를 해결하는 패치(또는 소스 코드)와 함께 제공됩니다. 이 패치는 Red Hat Enterprise Linux 패키지에 적용되며 에라타 업데이트로 테스트 및 릴리스됩니다. 그러나 발표에 패치가 포함되어 있지 않은 경우 Red Hat 개발자는 먼저 문제를 해결하기 위해 소프트웨어의 유지 관리와 협력합니다. 문제가 해결되면 패키지를 테스트하여 에라타 업데이트로 릴리스됩니다.

시스템에서 사용되는 소프트웨어에 대해 에라타 업데이트가 릴리스되는 경우 시스템을 잠재적으로 취약한 시간을 최소화하기 위해 영향을 받는 패키지를 가능한 한 빨리 업데이트하는 것이 좋습니다.

3.1.1. 보안 업데이트 계획 및 구성

모든 소프트웨어에는 버그가 있습니다. 종종 이러한 버그는 악의적인 사용자에게 시스템을 노출할 수 있는 취약점을 유발할 수 있습니다. 업데이트되지 않은 패키지는 컴퓨터 침입의 일반적인 원인입니다. 발견된 취약점을 신속하게 제거하기 위해 보안 패치를 적시에 설치 계획을 구현해 악용할 수 없습니다.

보안 업데이트를 사용할 수 있을 때 테스트하고 설치를 위해 예약합니다. 업데이트 릴리스와 시스템의 설치 사이의 기간 동안 시스템을 보호하려면 추가 컨트롤을 사용해야 합니다. 이러한 제어는 정확한 취약성에 따라 달라지지만 추가 방화벽 규칙, 외부 방화벽 사용 또는 소프트웨어 설정 변경이 포함될 수 있습니다.

지원되는 패키지의 버그는 에라타 메커니즘을 사용하여 해결되었습니다. 에라타는 특정 에라타가 처리하는 문제에 대한 간략한 설명과 함께 하나 이상의 RPM 패키지로 구성됩니다. 모든 에라타는 **Red Hat Subscription Management** 서비스를 통해 활성 서브스크립션을 통해 고객에게 배포됩니다. 보안 문제를 해결하는 에라타를 *Red Hat Security Advisories* 라고 합니다.

보안 에라타 사용에 대한 자세한 내용은 3.2.1절. “고객 포털에서 보안 권고 보기” 을 참조하십시오. **RHN classic** 에서 마이그레이션하는 방법에 대한 지침을 포함하여 **Red Hat** 서브스크립션 관리 서비스에 대한 자세한 내용은 이 서비스와 관련된 설명서를 참조하십시오. Red Hat 서브스크립션 관리 .

3.1.1.1. Yum의 보안 기능 사용

YUM 패키지 관리자에는 보안 에라타를 검색, 나열, 표시 및 설치하는 데 사용할 수 있는 여러 가지 보안 관련 기능이 포함되어 있습니다. 이러한 기능을 통해 **Yum**을 사용하여 보안 업데이트도 설치할 수 있습니다.

시스템에서 사용 가능한 보안 관련 업데이트를 확인하려면 **root** 로 다음 명령을 입력합니다.

```
~]# yum check-update --security
Loaded plugins: langpacks, product-id, subscription-manager
rhel-7-workstation-rpms/x86_64                | 3.4 kB  00:00:00
No packages needed for security; 0 packages available
```

위의 명령은 비대화형 모드로 실행되므로 사용 가능한 업데이트가 있는지 자동 검사를 위해 스크립트에서 사용할 수 있습니다. 이 명령은 사용 가능한 보안 업데이트가 있고 0 가 없는 경우 종료 값 100 을 반환합니다. 오류가 발생하면 1을 반환합니다.

이와 유사하게 다음 명령을 사용하여 보안 관련 업데이트만 설치합니다.

```
~]# yum update --security
```

updateinfo 하위 명령을 사용하여 사용 가능한 업데이트에 대해 리포지토리에서 제공한 정보를 표시하거나 작동합니다. **updateinfo** 하위 명령 자체는 여러 명령을 허용하지만 보안 관련 사용과 관련된 몇 가지 명령도 있습니다. 이러한 명령의 개요는 표 3.1. "yum updateinfo에서 사용할 수 있는 보안 관련 명령" 을 참조하십시오.

표 3.1. yum updateinfo에서 사용할 수 있는 보안 관련 명령

명령	설명
advisory [advisories]	하나 이상의 권고에 대한 정보를 표시합니다. *advisories*를 권고 번호 또는 숫자로 바꿉니다.
cves	CVE (*Common Vulnerabilities and Exposures*)와 관련된 정보의 하위 집합을 표시합니다.
보안 또는 초	모든 보안 관련 정보를 표시합니다.
심각도 **[**심각도_**level]** 또는 **sev [severity_level]**	제공된 *severity_level* 의 보안 관련 패키지에 대한 정보를 표시합니다.

3.1.2. 패키지 업데이트 및 설치

시스템에서 소프트웨어를 업데이트할 때는 신뢰할 수 있는 소스에서 업데이트를 다운로드해야 합니다. 공격자는 문제를 해결하려는 것과 동일한 버전 번호가 있는 패키지를 쉽게 다시 빌드할 수 있지만, 다른 보안 취약점을 악용하여 인터넷에서 릴리스할 수 있습니다. 이 경우 원래 RPM 에 대한 파일 확인과 같은 보안 수단을 사용하면 이 취약점을 탐지하지 않습니다. 따라서 Red Hat과 같은 신뢰할 수 있는 소스에서 RPM만 다운로드하고 패키지 서명을 확인하여 무결성을 확인하는 것이 매우 중요합니다.

YUM 패키지 관리자 사용 방법에 대한 자세한 내용은 Red Hat Enterprise Linux 7 시스템 관리자 가이드의 **YUM** 장을 참조하십시오. https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/system_administrators_guide/ch-yum

3.1.2.1. 서명된 패키지 확인

모든 Red Hat Enterprise Linux 패키지는 Red Hat **GPG** 키를 사용하여 서명합니다. **GPG** 는 분산 파일의 진위 여부를 확인하는 데 사용되는 무료 소프트웨어 패키지인 **GNU Privacy Guard** 또는 **GnuPG** 를 나타냅니다. 패키지 서명 확인이 실패하면 패키지가 변경되어 신뢰할 수 없습니다.

YUM 패키지 관리자는 설치 또는 업그레이드에 사용되는 모든 패키지를 자동으로 확인할 수 있습니다. 이 기능은 기본적으로 활성화되어 있습니다. 시스템에서 이 옵션을 구성하려면 **/etc/yum.conf** 설정 파일에서 **gpgcheck** 설정 지시문이 **1**로 설정되어 있는지 확인합니다.

다음 명령을 사용하여 파일 시스템에서 패키지 파일을 수동으로 확인합니다.

```
rpmkeys --checksig package_file.rpm
```

Red Hat 패키지 서명 관행에 대한 자세한 내용은 Red Hat 고객 포털의 제품 서명(GPG) 키 문서를 참조하십시오.

3.1.2.2. 서명된 패키지 설치

검증된 패키지를 설치하려면(3.1.2.1절. "서명된 패키지 확인" 파일 시스템에서 패키지를 확인하는 방법에 대한 자세한 내용은 다음과 같이 **yum install** 명령을 **root** 사용자로 사용합니다.

```
yum install package_file.rpm
```

쉘 글을 사용하여 한 번에 여러 패키지를 설치합니다. 예를 들어 다음 명령은 현재 디렉터리에 모든 **.rpm** 패키지를 설치합니다.

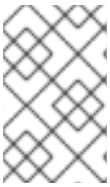
```
yum install *.rpm
```

중요

보안 에라타를 설치하기 전에 에라타 보고서에 포함된 특수 지침을 읽고 적절하게 실행하십시오. 에라타 업데이트로 인한 변경 사항 적용에 대한 일반적인 지침은 3.1.3절. "설치된 업데이트를 통해 변경 사항 적용"를 참조하십시오.

3.1.3. 설치된 업데이트를 통해 변경 사항 적용

보안 에라타 및 업데이트를 다운로드하여 설치한 후에는 이전 소프트웨어의 사용을 중단하고 새 소프트웨어 사용을 시작하는 것이 중요합니다. 이 작업을 수행하는 방법은 업데이트된 소프트웨어 유형에 따라 다릅니다. 다음 목록은 일반적인 소프트웨어 카테고리를 나열하고 패키지 업그레이드 후 업데이트된 버전을 사용하는 방법에 대한 지침을 제공합니다.



참고

일반적으로 시스템을 재부팅하는 것은 최신 버전의 소프트웨어 패키지가 사용되는지 확인하는 가장 좋은 방법입니다. 그러나 이 옵션은 항상 필수는 아니며 시스템 관리자가 항상 사용할 수 있는 것은 아닙니다.

애플리케이션

사용자 공간 애플리케이션은 사용자가 시작할 수 있는 모든 프로그램입니다. 일반적으로 이러한 애플리케이션은 사용자, 스크립트 또는 자동화된 작업 유틸리티가 이를 시작할 때만 사용됩니다.

이러한 사용자 공간 애플리케이션이 업데이트되면 시스템에서 애플리케이션의 인스턴스를 중지한 후 프로그램을 다시 시작하여 업데이트된 버전을 사용합니다.

커널

커널은 Red Hat Enterprise Linux 7 운영 체제의 핵심 소프트웨어 구성 요소입니다. 메모리, 프로세서 및 주변 장치에 대한 액세스를 관리하고 모든 작업을 예약합니다.

중앙 역할 때문에 컴퓨터를 재부팅하지 않고 커널을 다시 시작할 수 없습니다. 따라서 시스템을 재부팅할 때까지 업데이트된 커널 버전을 사용할 수 없습니다.

KVM

qemu-kvm 및 libvirt 패키지가 업데이트되면 모든 게스트 가상 머신을 중지하고 관련 가상화 모듈을 다시 로드하거나(또는 호스트 시스템을 재부팅) 가상 시스템을 다시 시작해야 합니다.

lsmod 명령을 사용하여 로드되는 모듈(kvm , **kvm -intel** 또는 **kvm-amd**)을 확인합니다. 그런 다음 **modprobe -r** 명령을 사용하여 **modprobe -a** 명령을 제거한 후 영향을 받는 모듈을 다시 로드합니다. 예:

```
~]# lsmod | grep kvm
kvm_intel             143031  0
kvm                   460181  1 kvm_intel
~]# modprobe -r kvm-intel
~]# modprobe -r kvm
~]# modprobe -a kvm kvm-intel
```

공유 라이브러리

공유 라이브러리는 여러 애플리케이션 및 서비스에서 사용하는 **glibc**와 같은 코드 단위입니다. 공유 라이브러리를 사용하는 애플리케이션은 일반적으로 애플리케이션이 초기화될 때 공유 코드를 로드하므로 업데이트된 라이브러리를 사용하는 모든 애플리케이션을 중단하고 다시 시작해야 합니다.

특정 라이브러리에 대해 어떤 애플리케이션 링크를 실행 중인지 확인하려면 **lsof** 명령을 사용합니다.

```
lsof library
```

예를 들어 **libwrap.so.0** 라이브러리에 대해 어떤 애플리케이션이 링크되는지 확인하려면 다음을 입력합니다.

```
~]# lsof /lib64/libwrap.so.0
COMMAND     PID USER  FD   TYPE DEVICE SIZE/OFF     NODE NAME
pulseaudi 12363 test mem    REG  253,0    42520 34121785 /usr/lib64/libwrap.so.0.7.6
gnome-set 12365 test mem    REG  253,0    42520 34121785 /usr/lib64/libwrap.so.0.7.6
gnome-she 12454 test mem    REG  253,0    42520 34121785 /usr/lib64/libwrap.so.0.7.6
```

이 명령은 host-access 컨트롤에 **TCP** 래퍼를 사용하는 실행 중인 모든 프로그램 목록을 반환합니다. 따라서 tcp_wrappers 패키지가 업데이트될 때 나열된 모든 프로그램을 중단하고 다시 시작해야 합니다.

systemd 서비스

systemd 서비스는 일반적으로 부팅 프로세스 중에 시작되는 영구 서버 프로그램입니다. systemd 서비스의 예로는 **sshd** 또는 **vsftpd**가 있습니다.

이러한 프로그램은 일반적으로 시스템이 실행되는 동안 메모리에 유지되므로 패키지를 업그레이드한 후 업데이트된 systemd 서비스를 중단하고 다시 시작해야 합니다. 이 작업은 **systemctl** 명령을 사용하여 **root** 사용자로 수행할 수 있습니다.

```
systemctl restart service_name
```

*service_name*을 재시작할 서비스 이름으로 교체합니다 (예: **sshd**).

기타 소프트웨어

아래 링크된 리소스에서 설명하는 지침에 따라 다음 애플리케이션을 올바르게 업데이트합니다.

- **Red Hat Directory Server** - https://access.redhat.com/documentation/en-US/Red_Hat_Directory_Server/ 에 해당하는 Red Hat Directory Server 버전에 대한 *릴리스 노트*를 참조하십시오.
- **Red Hat Enterprise Virtualization Manager** - https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Virtualization/ 에 해당하는 Red Hat Enterprise Virtualization 버전의 *설치 가이드*를 참조하십시오.

3.2. RED HAT 고객 포털 사용

https://access.redhat.com/의 Red Hat 고객 포털은 Red Hat 제품과 관련된 공식 정보를 위한 주요 고객 지향 리소스입니다. 이를 사용하여 문서를 찾고, 서브스크립션을 관리하고, 제품 및 업데이트를 다운로드하고, 지원 케이스를 열고, 보안 업데이트에 대해 알아볼 수 있습니다.

3.2.1. 고객 포털에서 보안 권고 보기

활성 서브스크립션이 있는 시스템과 관련된 보안 권고 (errata)를 보려면 https://access.redhat.com/에서 고객 포털에 로그인하고 기본 페이지의 제품 및 업데이트 다운로드 버튼을 클릭합니다. **Software & Download Center** (소프트웨어 및 다운로드 센터) 페이지를 입력하면 에라타 버튼을 클릭하여 등록된 시스템과 관련된 권고 목록을 확인합니다.

모든 활성 Red Hat 제품에 대한 모든 보안 업데이트 목록을 찾으려면 페이지 상단에 있는 탐색 메뉴를 사용하여 **Security → Security Updates → Active Products** 로 이동하십시오.

테이블의 왼쪽 부분에서 에라타 코드를 클릭하면 개별 권고에 대한 자세한 정보가 표시됩니다. 다음 페이지에는 원인, 결과 및 필수 수정 사항을 포함하여 지정된 에라타에 대한 설명뿐만 아니라 특정 에라타가 업데이트 적용 방법에 대한 지침과 함께 모든 패키지 목록이 포함되어 있습니다. 페이지에는 관련 CVE와 같은 관련 참조 링크도 포함되어 있습니다.

3.2.2. CVE 고객 포털 페이지 탐색

MITRE Corporation이 관리하는 CVE (*Common Vulnerabilities and Exposures*) 프로젝트는 취약점 및 보안 노출에 대한 표준화된 이름 목록입니다. 고객 포털의 Red Hat 제품과 관련된 CVE 목록을 검색하려면 https://access.redhat.com/ 에서 계정에 로그인하고 페이지 상단의 탐색 메뉴를 사용하여 보안 → 리소스 → **CVE Database** 로 이동합니다.

테이블의 왼쪽 부분에 있는 CVE 코드를 클릭하여 개별 취약점에 대한 자세한 정보를 표시합니다. 다음 페이지에는 지정된 CVE 에 대한 설명뿐만 아니라 관련 Red Hat 에라타 링크와 함께 영향을 받는 Red Hat 제품 목록이 포함되어 있습니다.

3.2.3. 심각도 등급 이해

Red Hat 제품에서 발견된 모든 보안 문제에는 문제의 심각도에 따라 *Red Hat 제품 보안* 의 영향 등급이 지정되었습니다. 4개 등급은 다음 수준으로 구성됩니다. 낮음, 보통, 중요 및 심각. 그 외에도 모든 보안 문제는CVSS(*Common Vulnerability Scoring System*) 기본 점수를 사용하여 평가됩니다.

이러한 평점은 보안 문제의 영향을 이해하는 데 도움이되므로 시스템의 업그레이드 전략을 예약하고 우선 순위를 지정할 수 있습니다. 이 등급은 현재 위협 수준이 아닌 버그 분석을 기반으로 하는 지정된 취약점의 잠재적 위험을 반영합니다. 즉, 특정 결함에 대한 취약점이 릴리스되면 보안 영향 등급이 변경되지 않습니다.

고객 포털에서 개별 심각도 등급에 대한 자세한 설명을 보려면 심각도 등급 페이지를 참조하십시오.

3.3. 추가 리소스

보안 업데이트, 적용 방법, Red Hat 고객 포털 및 관련 항목에 대한 자세한 내용은 아래 정보를 참조하십시오.

설치된 문서

- yum(8) - **Yum** 패키지 관리자의 도움말 페이지에서는 yum을 사용하여 시스템에서 패키지를 설치, 업데이트, 제거하는 데 사용할 수 있는 방법에 대한 정보를 제공합니다.

- rpmkeys(8) - **rpmkeys** 유틸리티의 도움말 페이지에서 다운로드한 패키지의 진위 여부를 확인하는 데 이 프로그램을 사용할 수 있는 방법을 설명합니다.

온라인 문서

- Red Hat Enterprise Linux 7 시스템 관리자 가이드 - Red Hat Enterprise Linux 7용 *시스템 관리자 안내서* 에서는 Red Hat Enterprise Linux 7 시스템에서 패키지를 설치, 업데이트 및 제거하는 데 사용되는 **YUM** 및 **rpm** 명령 사용에 대한 문서화를 문서화합니다.

- Red Hat Enterprise Linux 7 SELinux 사용자 및 관리자 가이드 - Red Hat Enterprise Linux 7용 *SELinux 사용자 및 관리자* 안내서는 **SELinux** *필수 액세스 제어* 메커니즘의 구성을 문서화합니다.

Red Hat 고객 포털

- Red Hat 고객 포털, 보안 - 고객 포털의 보안 섹션에는 Red Hat CVE 데이터베이스 및 Red Hat 제품 보안 연락처를 포함한 가장 중요한 리소스에 대한 링크가 포함되어 있습니다.

- Red Hat 보안 블로그 - Red Hat 보안 전문가의 최신 보안 관련 문제에 대해 설명합니다.

다음을 참조하십시오.

- 2장. *설치를 위한 보안 팁* 나중에 추가 보안 설정을 보다 쉽게 구현할 수 있도록 처음부터 시스템을 안전하게 구성하는 방법을 설명합니다.

- 4.9.2절. “GPG 키 생성” 통신을 인증하기 위해 개인 **GPG** 키 세트를 생성하는 방법을 설명합니다.

4장. 툴 및 서비스로 시스템 강화

4.1. 데스크탑 보안

Red Hat Enterprise Linux 7은 공격에 대해 데스크탑을 강화하고 무단 액세스를 방지할 수 있는 몇 가지 방법을 제공합니다. 이 섹션에서는 이동식 미디어의 사용자 암호, 세션 및 계정 잠금 및 안전한 처리 방법에 대해 설명합니다.

4.1.1. 암호 보안

암호는 Red Hat Enterprise Linux 7에서 사용자의 ID를 확인하는 데 사용하는 기본 방법입니다. 따라서 사용자, 워크스테이션 및 네트워크를 보호하는 데 암호 보안이 매우 중요합니다.

보안상의 이유로 설치 프로그램은 보안 *해시 알고리즘 512(SHA512*) 및 섀도우 암호를 사용하도록 시스템을 구성합니다. 이러한 설정을 변경하지 않는 것이 좋습니다.

설치 중에 섀도우 암호를 선택 해제하는 경우 모든 암호가 읽기 쉬운 **/etc/passwd** 파일에서 단방향 해시로 저장되므로 시스템은 오프라인 암호 크래핑 공격에 취약해집니다. 침입자가 일반 사용자로 시스템에 액세스할 수 있는 경우 **/etc/passwd** 파일을 자체 시스템에 복사하고 여러 암호 크래킹 프로그램을 실행할 수 있습니다. 파일에 안전하지 않은 암호가 있는 경우 암호 크래커가 이를 발견하기 전에 시간 문제 일뿐입니다.

shadow 암호는 root 사용자만 읽을 수 있는 **/etc/shadow** 파일에 암호 해시를 저장하여 이러한 유형의 공격을 제거합니다.

이로 인해 잠재적인 공격자는 SSH 또는 FTP와 같은 시스템의 네트워크 서비스에 로그인하여 원격으로 암호 크래킹을 시도하게 됩니다. 이러한 종류의 무차별 공격은 훨씬 느리고 시스템 파일에 실패한 로그인 시도가 수백 개이므로 명확한 추적이 유지됩니다. 물론, 크래커가 약한 암호가 있는 시스템에서 야간간 공격을 시작하면, 크래커는 dawn 전에 액세스할 수 있고 로그 파일을 편집하여 트랙들을 커버할 수 있습니다.

형식 및 스토리지 고려 사항 외에 콘텐츠 문제이기도 합니다. 사용자가 비밀번호 크래킹 공격으로부터 계정을 보호하기 위해 할 수있는 가장 중요한 일은 강력한 비밀번호를 만드는 것입니다.

참고

Red Hat IdM(Identity Management)과 같은 중앙 인증 솔루션을 사용하는 것이 좋습니다. 중앙 솔루션을 사용하는 것이 로컬 암호를 사용하는 것보다 선호됩니다. 자세한 내용은 다음을 참조하십시오.

- Red Hat Identity Management 소개
- 암호 정책 정의

4.1.1.1. 강력한 암호 만들기

보안 암호를 만들 때 사용자는 긴 암호가 짧고 복잡한 암호보다 더 강함을 알아야 합니다. 숫자, 특수 문자 및 대문자가 포함 된 경우에도 8자의 암호를 만드는 것은 좋지 않습니다. 존 The Ripper와 같은 암호 크래킹 도구는 사람이 기억하기 어려운 이러한 암호를 손상시키는 데 최적화되어 있습니다.

정보 이론에서 엔트로피는 임의의 변수와 관련된 불확실한 수준이며 비트로 표시됩니다. 엔트로피 값이 높을수록 암호가 더 안전합니다. NIST SP 800-63-1에 따르면 일반적으로 선택한 암호로 구성된 사전에 암호에 10비트 이상의 엔트로피가 있어야 합니다. 따라서 4개의 임의 단어로 구성된 암호에는 약 40비트의 엔트로피가 포함됩니다. 보안을 위해 여러 단어로 구성된 긴 암호는 *암호*라고도 합니다. 예를 들면 다음과 같습니다.

-

```
randomword1 randomword2 randomword3 randomword4
```

시스템이 대문자, 숫자 또는 특수 문자를 사용하도록 강제하는 경우 위 권장 사항 뒤에 있는 암호를 간단하게 수정할 수 있습니다. 예를 들어 첫 번째 문자를 대문자로 변경하고 "**1!**"를 추가합니다. 이러한 수정으로 인해 암호의 보안이 크게 증가 *되지 않습니다*.

암호를 직접 생성하는 또 다른 방법은 암호 생성기를 사용하는 것입니다. **pwmake** 는 대문자, 소문자, 숫자, 특수 문자 모두 네 개의 문자 그룹으로 구성된 임의의 암호를 생성하는 명령줄 도구입니다. 유틸리티를 사용하면 암호를 생성하는 데 사용되는 엔트로피 비트 수를 지정할 수 있습니다. 엔트로피는 **/dev/urandom** 에서 가져옵니다. 지정할 수 있는 최소 비트 수는 56이며 무차별력 공격이 거의 없는 시스템 및 서비스에서 암호하기에 충분합니다. 64비트는 공격자가 암호 해시 파일에 직접 액세스할 수 없는 애플리케이션에 적합합니다. 공격자가 암호 해시에 대한 직접 액세스 권한을 얻거나 암호가 암호화 키로 사용될 수 있는 경우 80~128비트를 사용해야 합니다. 잘못된 수의 엔트로피 비트를 지정하면 **pwmake** 가 기본값을 사용합니다. 128비트 암호를 만들려면 다음 명령을 입력합니다.

```
pwmake 128
```

보안 암호를 만드는 방법에는 여러 가지가 있지만 항상 다음 잘못된 사례를 방지합니다.

- 단일 사전 단어, 외부 언어로 된 단어, 반전된 단어 또는 숫자만 사용합니다.
- 암호 또는 암호에 10자 미만을 사용합니다.
- 키보드 레이아웃의 키 시퀀스 사용.
- 암호 쓰기.
- 생년월일, 가족 구성원 이름 또는 반려동물 이름과 같은 암호에 개인 정보를 사용합니다.
- 여러 시스템에서 동일한 암호 또는 암호를 사용합니다.

보안 암호를 생성하는 것이 필수 사항이지만, 특히 대규모 조직 내의 시스템 관리자에게도 적절하게 관리해야 합니다. 다음 섹션에서는 조직 내에서 사용자 암호를 만들고 관리하는 모범 사례를 자세히 설명합니다.

4.1.1.2. 강제 Strong 암호

조직에 많은 사용자가 있는 경우 시스템 관리자에게 강력한 암호 사용을 강제 적용하는 데 사용할 수 있는 두 가지 기본 옵션을 사용할 수 있습니다. 사용자를 위한 암호를 만들거나 암호를 확인하는 동안 사용자가 자신의 암호를 만들 수 있도록 할 수 있습니다.

사용자의 암호를 생성하면 암호가 양호하게 되지만 조직이 확장됨에 따라 어려운 작업이 됩니다. 또한 사용자가 암호를 아래로 쓰는 위험을 증가시켜 노출할 수 있습니다.

이러한 이유로 대부분의 시스템 관리자는 사용자가 고유 암호를 생성하는 것을 선호하지만 이러한 암호가 충분히 강하는지 적극적으로 확인합니다. 경우에 따라 관리자는 암호 사용 기간을 통해 사용자가 암호를 정기적으로 변경하도록 강제 적용할 수 있습니다.

사용자가 암호를 만들거나 변경하라는 메시지가 표시되면 *PAM*을 인식하는 **passwd** 명령줄 유틸리티(*플러그 가능 인증 모듈*)를 사용할 수 있으며 암호가 너무 짧거나 그렇지 않으면 쉽게 끊을 수 있는지 확인합니다. 이 검사는 **pam_datasources.so** PAM 모듈에서 수행합니다.

참고

Red Hat Enterprise Linux 7에서 **pam_** manage PAM 모듈은 Red Hat Enterprise Linux 6에서 암호 품질 검사를 위한 기본 모듈로 사용된 **pam_cracklib** 를 대체했습니다. **pam_cracklib**와 동일한 백엔드를 사용합니다.

pam_framework 모듈은 일련의 규칙에 대해 암호의 강도를 확인하는 데 사용됩니다. 절차는 두 단계로 구성됩니다. 먼저 제공된 암호가 사전에 있는지 확인합니다. 그렇지 않은 경우, 계속 많은 추가 검사로 계속됩니다. **pam_** ResourceOverride는 **/etc/pam.d/passwd** 파일의 암호 구성 요소에 있는 다른 PAM 모듈과 함께 스택화되며 사용자 지정 규칙 세트가 **/etc/security/ingressgateway.conf** 설정 파일에 지정됩니다. 이러한 검사의 전체 목록은 **pwquality.conf (8)** 매뉴얼 페이지를 참조하십시오.

예 **4.1. pwquality.conf**에서 암호 강도 검사 구성

pam_quality를 사용하여 활성화하려면 **/etc/pam.d/passwd** 파일의 **password** 스택에 다음 행을 추가하십시오.

```
password    required    pam_pwquality.so retry=3
```

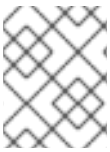
검사 옵션은 한 줄에 하나씩 지정됩니다. 예를 들어 4개의 문자 클래스를 모두 포함하여 최소 8자 길이의 암호가 필요한 경우 **/etc/security/dockerfile.conf** 파일에 다음 행을 추가합니다.

```
minlen = 8
minclass = 4
```

문자 순서와 동일한 연속 문자에 대한 암호 강도를 설정하려면 **/etc/security/ResourceOverride.conf**에 다음 행을 추가합니다.

```
maxsequence = 3
maxrepeat = 3
```

이 예에서 입력한 암호는 **abcd**와 같은 단조 시퀀스에 3자 이상 포함할 수 없으며 **1111** 과 같은 3개 이상의 연속된 문자도 포함할 수 없습니다.



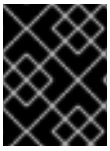
참고

root 사용자는 암호 생성에 대한 규칙을 적용하는 사용자이므로 경고 메시지가 있어도 암호를 자체 또는 일반 사용자에 대해 설정할 수 있습니다.

4.1.1.3. 암호 기간 구성

암호 변경은 시스템 관리자가 조직 내에서 잘못된 암호를 보호하는 데 사용하는 또 다른 기술입니다. 암호 기간 연장이란 지정된 기간(일반적으로 90일) 후에 새 암호를 입력하라는 메시지가 표시됩니다. 이것의 배경은 사용자가 암호를 정기적으로 변경하도록 강제하는 경우, 크래킹된 비밀번호는 제한된 시간 동안 침입자에게만 유용하기 때문입니다. 그러나 암호 기간 외에도 사용자가 암호를 아래로 작성할 가능성이 큽니다.

Red Hat Enterprise Linux 7에서 암호 사용 기간을 지정하려면 **chage** 명령을 사용하십시오.



중요

Red Hat Enterprise Linux 7에서는 기본적으로 shadow 암호가 활성화됩니다. 자세한 내용은 Red Hat Enterprise Linux 7 시스템 관리자 가이드 를 참조하십시오.

chage 명령의 **-M** 옵션은 암호가 유효한 최대 일 수를 지정합니다. 예를 들어 90일 후에 만료되도록 사용자 암호를 설정하려면 다음 명령을 사용합니다.

```
chage -M 90 username
```

위 명령에서 *username*을 사용자 이름으로 교체합니다. 암호 만료를 비활성화하려면 **-M** 옵션 뒤에 **-1** 값을 사용합니다.

chage 명령으로 사용 가능한 옵션에 대한 자세한 내용은 아래 표를 참조하십시오.

표 4.1. chage 명령줄 옵션

옵션	설명
-d *days*	1970년 1월 1일 이후 암호가 변경된 일 수를 지정합니다.
-E *date*	계정이 유휴 상태인 날짜를 YYYY-MM-DD 형식으로 지정합니다. 191970년 1월 1일 이후의 날짜 대신 사용할 수도 있습니다.
-I *days*	계정을 잠그기 전에 암호 만료 후 비활성 일 수를 지정합니다. 값이 **0** 이면 암호가 만료된 후 계정이 잠겼습니다.
-l	현재 계정 변경 설정을 나열합니다.
-m *days*	사용자가 암호를 변경해야 하는 최소 일 수를 지정합니다. 값이 **0** 이면 암호가 만료되지 않습니다.
- m*days*	암호가 유효한 최대 일 수를 지정합니다. 이 옵션에 지정된 일 수와 **-d** 옵션으로 지정된 일 수가 현재 날짜보다 작으면 계정을 사용하기 전에 암호를 변경해야 합니다.
-W *days*	사용자에게 경고할 암호 만료일 전의 일 수를 지정합니다.

대화형 모드에서 **chage** 명령을 사용하여 여러 암호 사용 기간 및 계정 세부 정보를 수정할 수도 있습니다. 다음 명령을 사용하여 대화형 모드를 시작합니다.

```
chage <username>
```

다음은 이 명령을 사용하는 샘플 대화식 세션입니다.

```
~]# chage juan
Changing the aging information for juan
Enter the new value, or press ENTER for the default
Minimum Password Age [0]: 10
Maximum Password Age [99999]: 90
Last Password Change (YYYY-MM-DD) [2006-08-18]:
```

```
Password Expiration Warning [7]:
Password Inactive [-1]:
Account Expiration Date (YYYY-MM-DD) [1969-12-31]:
```

사용자가 처음 로그인할 때 암호가 만료되도록 구성할 수 있습니다. 이렇게 하면 사용자가 즉시 암호를 변경해야 합니다.

1. 초기 암호를 설정합니다. 기본 암호를 할당하려면 쉘 프롬프트에 **root**로 다음 명령을 입력합니다.

   ```
   passwd username
   ```

   주의

   passwd 유틸리티에는 null 암호를 설정하는 옵션이 있습니다. null 암호를 사용하는 것은 편리하지만, 타사가 안전하지 않은 사용자 이름을 사용하여 시스템에 로그인하고 액세스할 수 있으므로 매우 안전하지 않은 방법입니다. 가능한 경우 null 암호를 사용하지 마십시오. 사용할 수 없는 경우 항상 null 암호로 계정을 잠금 해제하기 전에 사용자가 로그인할 준비가 되었는지 확인하십시오.

2. **root**로 다음 명령을 실행하여 즉각적인 암호 만료를 강제 적용합니다.

   ```
   chage -d 0 username
   ```

   이 명령은 암호가 마지막으로 변경된 날짜(1970년 1월 1일) 값을 설정합니다. 이 값은 암호 사용 기간 정책이 있는 경우 즉시 암호 만료를 강제 적용합니다.

처음 로그인하면 사용자에게 새 암호를 입력하라는 메시지가 표시됩니다.

4.1.2. 계정 잠금

Red Hat Enterprise Linux 7에서 **pam_faillock** PAM 모듈을 사용하면 시스템 관리자가 지정된 횟수의 실패한 시도 후 사용자 계정을 잠글 수 있습니다. 사용자 로그인 시도 제한은 주로 사용자의 계정 암호를 얻기 위해 가능한 무차별 강제 공격을 방지하는 보안 조치로 사용됩니다.

pam_faillock 모듈을 사용하면 실패한 로그인 시도가 **/var/run/faillock** 디렉터리의 각 사용자에 대해 별도의 파일에 저장됩니다.

참고

실패한 시도 로그 파일의 행 순서가 중요합니다. 이 순서의 변경 사항은 **even_deny_ root** 옵션이 사용되는 경우 루트 사용자 계정을 포함하여 모든 사용자 계정을 잠글 수 있습니다.

다음 단계에 따라 계정 잠금을 구성합니다.

1. 3번 시도에 실패하는 후 root가 아닌 사용자를 잠그고 10분 후에 잠금을 해제하려면 **/etc/pam.d/system- auth** 및 **/etc/pam.d/password-auth** 파일의 auth 섹션에 두 줄을 추가합니다. 편집 후 두 파일의 전체 **auth** 섹션은 다음과 같아야 합니다.

```
auth        required      pam_env.so
auth        required      pam_faillock.so preauth silent audit deny=3 unlock_time=600
auth        sufficient    pam_unix.so nullok try_first_pass
auth        [default=die] pam_faillock.so authfail audit deny=3 unlock_time=600
auth        requisite     pam_succeed_if.so uid >= 1000 quiet_success
auth        required      pam_deny.so
```

라인 번호 2 및 4가 추가되었습니다.

2. 이전 단계에서 지정된 두 파일의 **account** 섹션에 다음 줄을 추가합니다.

```
account     required      pam_faillock.so
```

3. root 사용자에 대한 계정 잠금을 적용하려면 **even_deny_root** 옵션을 **/etc/ pam.d/system-auth** 및 **/etc/pam.d/password-auth** 파일의 **pam_** RuntimeConfig 항목에 추가합니다.

```
auth        required      pam_faillock.so preauth silent audit deny=3 even_deny_root
unlock_time=600
auth        sufficient    pam_unix.so nullok try_first_pass
auth        [default=die] pam_faillock.so authfail audit deny=3 even_deny_root
unlock_time=600

account     required      pam_faillock.so
```

이전에 세 번 로그인하지 못한 후 사용자 **john** 이 네 번째 시간 동안 로그인을 시도하면 네 번째 시도 시 계정이 잠깁니다.

```
~]$ su - john
Account locked due to 3 failed logins
su: incorrect password
```

여러 로그인 실패 후에도 시스템이 사용자를 잠그지 못하도록 하려면 **/etc/pam.d/system-auth** 및 **/etc/pam.d/password-auth** 에서 **pam_faillock** 가 처음 호출되는 행 바로 위에 다음 행을 추가하십시오. 또한 **user1**,**user2**, **user3** 을 실제 사용자 이름으로 바꿉니다.

```
auth [success=1 default=ignore] pam_succeed_if.so user in user1:user2:user3
```

사용자당 실패한 시도 수를 보려면 **root**로 다음 명령을 실행합니다.

```
~]$ faillock
john:
When                Type  Source                                           Valid
2013-03-05 11:44:14 TTY   pts/0                                               V
```

사용자 계정의 잠금을 해제하려면 **root**로 다음 명령을 실행합니다.

```
faillock --user <username> --reset
```

중요

cron 작업을 실행하면 cron 작업을 실행 중인 해당 사용자의 load counter가 해당 사용자의 실패 카운터가 재설정되므로, **cron** 에 대해 should not be configured for **cron**. 자세한 내용은 **Knowledge Centered Support (KCS)** 솔루션을 참조하십시오.

authconfig로 사용자 정의 설정 유지

authconfig 유틸리티를 사용하여 인증 구성을 수정할 때 **system-auth** 및 **password-auth** 파일을 **authconfig** 유틸리티의 설정으로 덮어씁니다. 이 문제는 **authconfig**가 인식하고 덮어쓰지 않는 구성 파일 대신 심볼릭 링크를 생성하여 방지할 수 있습니다. 구성 파일 및 **authconfig**에서 사용자 지정 설정을 동시에 사용하려면 다음 단계를 사용하여 계정 잠금을 구성합니다.

1. **system-auth** 및 **password-auth** 파일이 이미 **system-auth-ac** 및 **password-auth-ac** 을 가리키는 심볼릭 링크인지 확인합니다(시스템 기본값임).

```
~]# ls -l /etc/pam.d/{password,system}-auth
```

출력이 다음과 유사한 경우 심볼릭 링크가 제 위치에 있고 단계 번호 **3**으로 건너뛸 수 있습니다.

```
lrwxrwxrwx. 1 root root 16 24. Feb 09.29 /etc/pam.d/password-auth -> password-auth-ac
lrwxrwxrwx. 1 root root 28 24. Feb 09.29 /etc/pam.d/system-auth -> system-auth-ac
```

system-auth 및 **password-auth** 파일이 심볼릭 링크가 아닌 경우 다음 단계를 계속합니다.

2. 구성 파일의 이름을 변경합니다.

```
~]# mv /etc/pam.d/system-auth /etc/pam.d/system-auth-ac
~]# mv /etc/pam.d/password-auth /etc/pam.d/password-auth-ac
```

3. 사용자 지정 설정으로 구성 파일을 생성합니다.

```
~]# vi /etc/pam.d/system-auth-local
```

/etc/pam.d/system-auth-local 파일에는 다음 행이 포함되어야 합니다.

```
auth        required      pam_faillock.so preauth silent audit deny=3 unlock_time=600
auth        include       system-auth-ac
auth        [default=die] pam_faillock.so authfail silent audit deny=3 unlock_time=600
```

```
account     required      pam_faillock.so
account     include       system-auth-ac

password    include       system-auth-ac

session     include       system-auth-ac
```

```
~]# vi /etc/pam.d/password-auth-local
```

/etc/pam.d/password-auth-local 파일에는 다음 행이 포함되어야 합니다**.**

```
auth        required      pam_faillock.so preauth silent audit deny=3 unlock_time=600
auth        include       password-auth-ac
auth        [default=die] pam_faillock.so authfail silent audit deny=3 unlock_time=600

account     required      pam_faillock.so
account     include       password-auth-ac

password    include       password-auth-ac

session     include       password-auth-ac
```

4. 다음 심볼릭 링크를 만듭니다**.**

```
~]# ln -sf /etc/pam.d/system-auth-local /etc/pam.d/system-auth
~]# ln -sf /etc/pam.d/password-auth-local /etc/pam.d/password-auth
```

다양한 **pam_faillock** 구성 옵션에 대한 자세한 내용은 **pam_faillock(8)** 매뉴얼 페이지를 참조하십시오**.**

nullok 옵션 제거

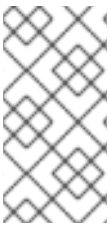
nullok 옵션을 사용하면 기본적으로 **/etc/shadow** 파일의 **password** 필드가 비어 있는 경우 빈 암호로 로그인할 수 있습니다**. nullok** 옵션을 비활성화하려면 **/etc/pam.d/system-auth** 또는 **/etc/pam.d/password-auth** 와 같은 **/etc/pam.d/** 디렉토리의 구성 파일에서 **nullok** 문자열을 제거합니다**.**

Will nullok 옵션을 사용하여 사용자가 암호를 입력하지 않고 로그인할 수 있도록 허용합니까**?** 더 많은 정보를 위한 **KCS** 솔루션

4.1.3. 세션 잠금

사용자는 일상적인 작업 중에 여러 가지 이유로 워크스테이션을 무력화해야 할 수도 있습니다**.** 이로 인해 공격자는 특히 물리적 보안 조치가 부족한 환경에서 물리적으로 머신에 액세스할 수 있는 기회를 제공

할 수 있습니다(1.2.1절. “물리적 제어”참조). 노트북은 특히 이동성이 물리적 보안을 방해하기 때문에 노출됩니다. 올바른 암호가 입력될 때까지 시스템 액세스를 방지하는 세션 잠금 기능을 사용하여 이러한 위험을 줄일 수 있습니다.



참고

로그아웃하는 대신 화면을 잠그는 주요 이점은 잠금을 통해 사용자의 프로세스(예: 파일 전송)를 계속 실행할 수 있다는 것입니다. 로그아웃하면 이러한 프로세스가 중지됩니다.

4.1.3.1. vlock을 사용하여 가상 콘솔 잠금

사용자는 가상 콘솔을 잠금 해제해야 할 수도 있습니다. 이 작업은 **vlock** 이라는 유틸리티를 사용하여 수행할 수 있습니다. 이 유틸리티를 설치하려면 **root**로 다음 명령을 실행합니다.

```
~]# yum install vlock
```

설치 후 추가 매개변수 없이 **vlock** 명령을 사용하여 모든 콘솔 세션을 잠글 수 있습니다. 다른 사용자에게 계속 액세스할 수 있도록 허용하면서 현재 활성화된 가상 콘솔 세션이 잠깁니다. 워크스테이션의 모든 가상 콘솔에 액세스하지 못하도록 하려면 다음을 실행합니다.

```
vlock -a
```

이 경우 **vlock** 현재 활성화된 콘솔을 잠금 해제하고 **-a** 옵션을 사용하면 다른 가상 콘솔로 전환하지 않습니다.

자세한 내용은 **vlock(1)** 매뉴얼 페이지를 참조하십시오.

4.1.4. Removable Media의 읽기 전용 마운트 적용

이동식 미디어(예: USB 플래시 디스크)의 읽기 전용 마운트를 적용하기 위해 관리자는 **udev** 규칙을 사용하여 이동식 미디어를 감지하고 **blockdev** 유틸리티를 사용하여 읽기 전용으로 마운트하도록 구성할 수 있습니다. 이 경우 물리적 미디어의 읽기 전용 마운트를 강제 적용하는 데 충분합니다.

blockdev를 사용하여 **Removable** 미디어의 읽기 전용 마운트를 강제 사용

모든 이동식 미디어를 읽기 전용으로 마운트하려면 라는 새 **udev** 구성 파일(예: **/etc/udev/rules.d/** 디렉터리의 **80-readonly-removables.rules**)을 만듭니다.

```
SUBSYSTEM=="block",ATTRS{removable}=="1",RUN{program}="/sbin/blockdev --setro %N"
```

위의 **udev** 규칙을 사용하면 새로 연결된 이동식 블록**(storage)** 장치가 **blockdev** 유틸리티를 사용하여 자동으로 읽기 전용으로 구성됩니다**.**

새 **udev** 설정 적용

이러한 설정을 적용하려면 새 **udev** 규칙을 적용해야 합니다**. udev** 서비스는 구성 파일의 변경 사항을 자동으로 감지하지만 새 설정은 기존 장치에 적용되지 않습니다**.** 새로 연결된 장치만 새 설정의 영향을 받습니다**.** 따라서 다음에 연결할 때 새 설정이 적용되는지 확인하기 위해 연결된 모든 이동식 미디어를 마운트 해제하고 분리해야 합니다**.**

udev 가 이미 존재하는 장치에 모든 규칙을 다시 적용하도록 하려면 **root** 로 다음 명령을 입력합니다**.**

```
~# udevadm trigger
```

위 명령을 사용하여 모든 규칙을 다시 적용하도록 **udev** 를 강제 적용하면 이미 마운트된 스토리지 장치에는 영향을 미치지 않습니다**.**

udev 에서 모든 규칙을 다시 로드하도록 하려면**(**새 규칙이 자동으로 탐지되지 않는 경우**)** 다음 명령을 사용합니다**.**

```
~# udevadm control --reload
```

4.2. 루트 액세스 제어

홈 시스템을 관리할 때 사용자는 **root** 사용자로 일부 작업을 수행하거나 **sudo** 또는 **su** 와 같은 ***setuid*** 프로그램을 사용하여 유효한 루트 권한을 획득해야 합니다**. setuid** 프로그램은 프로그램을 실행하는 사용자가 아니라 프로그램 소유자의 사용자 **ID(*UID*)**로 작동하는 프로그램입니다**.** 이러한 프로그램은 다음 예와 같이 긴 형식 목록의 소유자 섹션에 있는 **s** 로 표시됩니다**.**

```
~]$ ls -l /bin/su
-rwsr-xr-x. 1 root root 34904 Mar 10  2011 /bin/su
```

참고

s 는 대문자 또는 소문자일 수 있습니다**.** 대문자로 표시되는 경우 기본 권한 비트가 설정되지 않았음을 의미합니다**.**

그러나 조직의 시스템 관리자의 경우 조직 내의 관리 액세스 사용자가 시스템에 있어야 하는 액세스 수에 따라 선택을 수행해야 합니다**. pam_console.so** 라는 **PAM** 모듈을 통해 일반적으로 이동식 미디어 재

부팅 및 마운트와 같은 루트 사용자에 대해서만 예약되는 일부 활동은 물리 콘솔에서 로그인하는 첫 번째 사용자에게 허용됩니다. 그러나 네트워크 설정 변경, 새 마우스 구성 또는 네트워크 장치 마운트와 같은 기타 중요한 시스템 관리 작업은 관리자 권한 없이는 불가능합니다. 따라서 시스템 관리자는 네트워크에서 사용자에게 얼마나 많은 액세스 권한을 받아야 하는지 결정해야 합니다.

4.2.1. 루트 액세스 허용

관리자가 이러한 이유 또는 다른 이유로 **root** 로 로그인할 수 없는 경우 루트 암호는 비밀로 유지되어야 하며, 부트 로더 암호 보호를 통해 실행 수준 하나 또는 단일 사용자 모드에 대한 액세스는 허용하지 않아야 합니다(이 항목에 대한 자세한 내용은 **4.2.5**절. **"Boot Loader** 보안**"** 참조).

다음은 관리자가 더 이상 루트 로그인을 허용하지 않도록 할 수 있는 네 가지 방법입니다.

루트 쉘 변경

사용자가 **root** 로 직접 로그인하지 못하도록 시스템 관리자는 **/etc/passwd** 파일에서 **root** 계정의 쉘을 **/sbin/nologin** 으로 설정할 수 있습니다.

표 **4.2.** 루트 쉘 비활성화

효과	영향을 받지 않음
root 쉘에 대한 액세스를 방지하고 이러한 시도를 기록합니다. 다음 프로그램이 **root** 계정에 액세스하지 못합니다. • **login** • **gdm** • **kdm** • **xdm** • **su** • **ssh** • **scp** • **sftp**	**FTP** 클라이언트, 메일 클라이언트 및 많은 **setuid** 프로그램과 같은 쉘이 필요하지 않은 프로그램입니다. 다음 프로그램은 **root** 계정에 액세스 ***하지 못합니다.*** • **sudo** • **FTP** 클라이언트 • 이메일 클라이언트

콘솔 장치**(tty)**를 사용하여 루트 액세스 비활성화

root 계정에 대한 액세스를 추가로 제한하기 위해 관리자는 **/etc/securetty** 파일을 편집하여 콘솔에서 **root** 로그인을 비활성화할 수 있습니다. 이 파일에는 **root** 사용자가 로그인할 수 있는 모든 장치가 나열됩니다. 파일이 전혀 없는 경우 루트 사용자는 콘솔 또는 원시 네트워크 인터페이스를 통해 시

스템상의 모든 통신 장치를 통해 로그인할 수 있습니다. 이는 사용자가 네트워크를 통해 일반 텍스트로 암호를 전송하는 **Telnet**을 사용하여 시스템에 **root** 로 로그인할 수 있기 때문에 위험합니다.

기본적으로 **Red Hat Enterprise Linux 7**의 **/etc/securetty** 파일은 **root** 사용자만 시스템에 물리적으로 연결된 콘솔에서만 로그인할 수 있습니다. 루트 사용자가 로그인하지 못하도록 하려면 쉘 프롬프트에서 **root** 로 다음 명령을 입력하여 이 파일의 내용을 제거합니다.

```
echo > /etc/securetty
```

KDM, GDM 및 **XDM** 로그인 관리자에서 **securetty** 지원을 활성화하려면 다음 줄을 추가하십시오.

```
auth [user_unknown=ignore success=ok ignore=ignore default=bad] pam_securetty.so
```

아래 나열된 파일에서 다음을 수행합니다.

- **/etc/pam.d/gdm**
- **/etc/pam.d/gdm-autologin**
- **/etc/pam.d/gdm-fingerprint**
- **/etc/pam.d/gdm-password**
- **/etc/pam.d/gdm-smartcard**
- **/etc/pam.d/kdm**
- **/etc/pam.d/kdm-np**
- **/etc/pam.d/xdm**

주의

인증이 끝날 때까지 콘솔이 열려 *있지 않기* 때문에 빈 **/etc/securetty** 파일이 루트 사용자가 **OpenSSH** 도구 모음을 사용하여 원격으로 로그인하지 못하도록 방지하지 않습니다.

표 **4.3.** 루트 로그인 비활성화

효과	영향을 받지 않음
콘솔 또는 네트워크를 사용하여 **root** 계정에 대한 액세스를 방지합니다. 다음 프로그램이 **root** 계정에 액세스하지 못합니다. • **login** • **gdm** • **kdm** • **xdm** • **tty**를 여는 기타 네트워크 서비스	**root**로 로그인하지 않고 **setuid** 또는 기타 메커니즘을 통해 관리 작업을 수행하는 프로그램입니다. 다음 프로그램은 **root** 계정에 액세스 *하지 못합니다*. • **su** • **sudo** • **ssh** • **scp** • **sftp**

루트 **SSH** 로그인 비활성화

SSH 프로토콜을 통한 **root** 로그인을 방지하려면 **SSH** 데몬의 구성 파일 **/etc/ssh/sshd_config** 를 편집하고 다음과 같은 행을 변경합니다**.**

```
#PermitRootLogin yes
```

다음과 같이 읽기**:**

```
PermitRootLogin no
```

표 **4.4.** 루트 **SSH** 로그인 비활성화

효과	영향을 받지 않음
OpenSSH 도구 모음을 사용하여 **root** 액세스를 방지합니다**.** 다음 프로그램이 **root** 계정에 액세스하지 못합니다**.** • **ssh** • **scp** • **sftp**	**OpenSSH** 도구 제품군에 포함되지 않은 프로그램**.**

PAM을 사용하여 서비스에 대한 **root** 액세스 제한

PAM은 **/lib/security/pam_listfile.so** 모듈을 통해 특정 계정을 거부할 수 있는 유연성을 제공합니다**.** 관리자는 이 모듈을 사용하여 로그인할 수 없는 사용자 목록을 참조할 수 있습니다**.** 시스템 서비스에 대한 루트 액세스를 제한하려면 **/etc/pam.d/** 디렉터리에서 타겟 서비스의 파일을 편집하고 인증에 필요한 **pam_listfile.so** 모듈이 필요합니다**.**

다음은 **/etc/pam.d/COMPLETE PAM** 설정 파일의 **vsftpd FTP** 서버에 모듈을 사용하는 방법의 예입니다**(**모듈이 한 줄에 있는 경우 첫 번째 줄의 \ 문자는 ***필요하지 않습니다*** **).**

```
auth   required   /lib/security/pam_listfile.so   item=user \
sense=deny file=/etc/vsftpd.ftpusers onerr=succeed
```

그러면 **PAM**이 **/etc/COMPLETE.ftpusers** 파일을 참조하고 나열된 모든 사용자에 대한 서비스에 대한 액세스를 거부하도록 지시합니다**.** 관리자는 이 파일의 이름을 변경할 수 있으며**,** 각 서비스에 대해 별도의 목록을 유지하거나 하나의 중앙 목록을 사용하여 여러 서비스에 대한 액세스를 거부할 수 있습니다**.**

관리자가 여러 서비스에 대한 액세스를 거부하려는 경우**,** 메일 클라이언트의 경우 **/etc/pam.d/pop** 및 **/etc/pam.d/imap** 과 같은 **PAM** 구성 파일에 유사한 행을 추가할 수 있습니다**. SSH** 클라이언트의 경우 **/etc/pam.d/ssh.**

PAM에 대한 자세한 내용은 **/usr/share/doc/pam *-<version>/html/ 디렉터리에 있는 Linux-*PAM** 시스템 관리자 가이드 를 참조하십시오**.**

표 **4.5. PAM**을 사용하여 루트 비활성화

효과	영향을 받지 않음
PAM이 인식하는 네트워크 서비스에 대한 루트 액세스를 방지합니다. 다음 서비스가 **root** 계정에 액세스하지 못합니다. • **login** • **gdm** • **kdm** • **xdm** • **ssh** • **scp** • **sftp** • **FTP** 클라이언트 • 이메일 클라이언트 • **PAM** 인식 서비스	**PAM**이 인식하지 못하는 프로그램 및 서비스

4.2.2. 루트 액세스 허용

조직 내 사용자가 신뢰할 수 있고 컴퓨터 복제가 있는 경우 루트 액세스를 허용하면 문제가 되지 않을 수 있습니다. 사용자의 루트 액세스를 허용하면 장치 추가 또는 네트워크 인터페이스 구성과 같은 사소한 활동이 개별 사용자가 처리할 수 있으므로 시스템 관리자가 네트워크 보안 및 기타 중요한 문제를 처리할 수 있습니다.

반면 개별 사용자에게 **root** 액세스 권한을 부여하면 다음과 같은 문제가 발생할 수 있습니다.

- ***머신 Misconfiguration* - root** 액세스 권한이 있는 사용자는 시스템을 잘못 구성할 수 있으며 문제를 해결하는 데 도움이 필요할 수 있습니다. 더 심각한 것은 모르는 상태에서 보안 허점을 열 수 있습니다.

- ***Insecure Services* - root** 액세스 권한을 가진 사용자는 **FTP** 또는 **Telnet**과 같이 시스템에서 안전하지 않은 서버를 실행하여 잠재적으로 사용자 이름과 암호를 위험에 빠뜨릴 수 있습니다. 이러한 서비스는 이러한 정보를 일반 텍스트로 네트워크를 통해 전송합니다.

- ***이메일 file file file as Root로 실행* - Linux**에 영향을 미치는 이메일 바이러스는 거의 존재하지 않습니다. 악의적인 프로그램은 루트 사용자가 실행할 때 가장 큰 위협을 초래합니다.

- ***감사 추적을 그대로 유지* -** 루트 계정이 여러 사용자가 공유하는 경우가 많기 때문에 여러 시스템 관리자가 시스템을 유지 관리할 수 있으므로 특정 시점에서 해당 사용자의 **root** 를 확인할 수 없습니다. 별도의 로그인을 사용하는 경우 사용자가 로그인한 계정과 세션 추적을 위해 고유 번호를 사용하여 로그인하면 사용자가 시작하는 모든 프로세스에서 상속되는 작업 구조에 배치됩니다. 동시 로그인을 사용하는 경우 고유 번호를 사용하여 특정 로그인에 대한 작업을 추적할 수 있습니다. 작업에서 감사 이벤트를 생성하면 로그인 계정 및 해당 고유 번호와 연결된 세션으로 기록됩니다. **aulast** 명령을 사용하여 이러한 로그인 및 세션을 확인합니다. **aulast** 명령의 **--proof** 옵션은 특정 세션으로 생성된 감사 가능한 이벤트를 격리하기 위해 특정 **ausearch** 쿼리를 제안하는 데 사용할 수 있습니다. 감사 시스템에 대한 자세한 내용은 **7**장. *시스템 감사*을 참조하십시오.

4.2.3. 루트 액세스 제한

관리자는 루트 사용자에 대한 액세스를 완전히 거부하는 대신 **su** 또는 **sudo** 와 같은 **setuid** 프로그램을 통해서만 액세스를 허용할 수 있습니다. **su** 및 **sudo** 에 대한 자세한 내용은 **Red Hat Enterprise Linux 7** 시스템 관리자 가이드의 권한 수집 장과 **su(1)** 및 **sudo(8)** 매뉴얼 페이지를 참조하십시오.

4.2.4. 자동 로그 제한 활성화

사용자가 **root** 로 로그인하면 무중단 로그인 세션이 심각한 보안 위험이 발생할 수 있습니다. 이 위험을 줄이기 위해 고정된 시간 후에 유휴 사용자가 자동으로 로그아웃되도록 시스템을 구성할 수 있습니다.

1. 루트 로서 **/etc/profile** 파일의 시작 부분에 다음 줄을 추가하여 이 파일의 처리가 중단될 수 없는지 확인합니다.

```
trap "" 1 2 3 15
```

2. 루트 로서 **/etc/profile** 파일에 다음 행을 삽입하여 **120**초 후에 자동으로 로그아웃합니다.

```
export TMOUT=120
readonly TMOUT
```

TMOUT 변수는 지정된 초 수에 대한 활동이 없는 경우 쉘을 종료합니다(위 예에서는 **120** 으로 설정). 특정 설치의 요구에 따라 제한을 변경할 수 있습니다.

4.2.5. Boot Loader 보안

Linux 부트 로더를 보호하는 비밀번호의 주요 이유는 다음과 같습니다.

1. ***단일 사용자 모드에 대한 액세스 방지*** - 공격자가 시스템을 단일 사용자 모드로 부팅할 수 있는 경우 루트 암호를 입력하라는 메시지가 표시되지 않고 자동으로 **root** 로 로그인됩니다.

주의

/etc/sysconfig/init 파일에서 ***SINGLE*** 매개 변수를 편집하여 단일 사용자 모드에 대한 액세스를 보호하는 것은 권장되지 않습니다. 공격자는 **GRUB 2**의 커널 명령줄에 사용자 지정 초기 명령(***init=*** 매개 변수 사용)을 지정하여 암호를 바이패스할 수 있습니다. **Red Hat Enterprise Linux 7** 시스템 관리자 가이드의 암호로 **GRUB 2** 부트 로더에 설명된 대로 **GRUB 2** 부트 로더를 암호로 보호하는 것이 좋습니다.

2. ***GRUB 2 콘솔에 대한 액세스 방지*** - 시스템에서 **GRUB 2**를 부트 로더로 사용하는 경우 공격자는 **GRUB 2** 편집기 인터페이스를 사용하여 설정을 변경하거나 **cat** 명령을 사용하여 정보를 수

집할 수 있습니다.

3. ***Insecure Operating Systems에 대한 액세스 방지 -*** 이중 부팅 시스템인 경우 공격자는 부팅 시 운영 체제를 선택할 수 있습니다(예: **access control** 및 파일 권한은 무시함).

Red Hat Enterprise Linux 7에는 **Intel 64** 및 **AMD64** 플랫폼에 **GRUB 2** 부트 로더가 포함되어 있습니다. **GRUB 2**에 대한 자세한 내용은 **Red Hat Enterprise Linux 7** 시스템 관리자 가이드의 **GRUB 2 Boot Loader** 장을 참조하십시오.

4.2.5.1. 대화형 시작 비활성화

부팅 시퀀스를 시작할 때 **I** 키를 누르면 시스템을 대화형으로 시작할 수 있습니다. 대화형 시작 중에 시스템에서 각 서비스를 하나씩 시작하라는 메시지를 표시합니다. 그러나 이를 통해 시스템에 대한 물리적 액세스 권한을 얻은 공격자는 보안 관련 서비스를 비활성화하고 시스템에 액세스할 수 있습니다.

사용자가 대화식으로 시스템을 시작하지 않도록 하려면 **/etc/sysconfig/init** 파일에서 ***PROMPT*** 매개변수를 비활성화합니다.

```
PROMPT=no
```

4.2.6. 하드 링크 및 심볼릭 링크 보호

악의적인 사용자가 보호되지 않은 하드 및 심볼릭 링크로 인해 발생하는 잠재적 취약점을 악용하지 못하도록 **Red Hat Enterprise Linux 7**에는 링크를 생성하거나 제공된 특정 조건만 충족할 수 있는 기능이 포함되어 있습니다.

하드 링크의 경우 다음 중 하나에 해당해야 합니다.

- 사용자는 자신이 연결되는 파일을 소유합니다.
- 사용자가 링크한 파일에 대한 읽기 및 쓰기 액세스 권한이 이미 있습니다.

심볼릭 링크의 경우, 프로세스는 고정 비트의 세계 쓰기 가능 디렉토리 외부에 있거나 다음 중 하나가 **true**이어야 하는 경우에만 링크를 따를 수 있습니다.

- 심볼릭 링크를 따르는 프로세스는 심볼릭 링크의 소유자입니다.

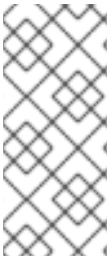
- 디렉터리의 소유자는 심볼릭 링크의 소유자와 동일합니다.

이 보호는 기본적으로 설정되어 있습니다. **/usr/lib/sysctl.d/50-default.conf** 파일의 다음 옵션으로 제어됩니다.

```
fs.protected_hardlinks = 1
fs.protected_symlinks = 1
```

기본 설정을 재정의하고 보호를 비활성화하려면 다음 콘텐츠를 사용하여 **/etc/sysctl.d/** 디렉터리에 라는 새 구성 파일 **(**예**: 51-no-protect-links.conf)**을 만듭니다.

```
fs.protected_hardlinks = 0
fs.protected_symlinks = 0
```



참고

기본 시스템 설정을 재정의하려면 새 구성 파일에 **.conf** 확장자가 있어야 하며 기본 시스템 파일**(**파일을 사전 순으로 읽고 있으므로 파일 이름이 시작 시 더 높은 수의 파일에 포함된 설정이 우선합니다**).**

***sysctl** 메커니즘을 사용하여 부팅 시 커널 매개변수 구성에 대한 자세한 내용은 **sysctl.d(5)** 매뉴얼 페이지를 참조하십시오.*

4.3. 서비스 보안

*관리 제어에 대한 사용자 액세스 문제는 조직 내의 시스템 관리자에게 중요한 문제이지만 어떤 네트워크 서비스가 활성 상태인지 모니터링하는 것이 **Linux** 시스템을 관리하고 운영하는 모든 사람에게 가장 중요한 문제입니다.*

***Red Hat Enterprise Linux 7**에 따른 많은 서비스는 네트워크 서버입니다. 네트워크 서비스가 시스템에서 실행 중인 경우 하나 이상의 네트워크 포트에서 연결을 수신 대기 중인 서버 애플리케이션(**daemon**) 입니다. 이러한 각 서비는 잠재적인 공격으로 취급되어야 합니다.*

4.3.1. 서비스에 대한 위험

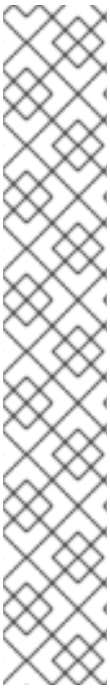
네트워크 서비스는 **Linux** 시스템에 많은 위험을 초래할 수 있습니다. 다음은 몇 가지 주요 문제의 목록입니다.

- **DoS(서비스 거부 공격) -** 요청으로 서비스 거부 공격으로 인해 서비스 거부 공격이 각 요청을 로그하고 응답하려고 하면 시스템을 사용할 수 없게 만들 수 있습니다.

- **DDoS(Distributed Denial of Service Attack) -** 여러 손상된 시스템을 사용하는 **DoS** 공격 유형으로 서비스에 대한 조정 공격을 지시하고 요청으로 플러딩하고 사용할 수 없게 만드는 **DoS** 공격 유형입니다.

- 스크립트 취약점 공격 **-** 서버가 스크립트를 사용하여 웹 서버에서 일반적으로 수행하는 것처럼 서버 측 작업을 실행하는 경우 공격자가 부적절하게 작성된 스크립트를 대상으로 할 수 있습니다. 이러한 스크립트 취약점 공격은 버퍼 오버플로 상태로 이어지거나 공격자가 시스템의 파일을 변경할 수 있도록 합니다.

- 버퍼 오버플로우 공격 **- 1023**에서 포트 **1**에서 수신 대기하려는 서비스는 관리 권한으로 시작되거나 **CAP_NET_BIND_SERVICE** 기능을 설정해야 합니다. 프로세스가 포트에 바인딩되어 있고 해당 프로세스가 수신 대기 중인 경우 권한 또는 기능이 종종 삭제됩니다. 권한 또는 기능이 삭제되지 않고 애플리케이션에 악용 가능한 버퍼 오버플로가 있는 경우 공격자가 데몬을 실행하는 사용자로 시스템에 액세스할 수 있습니다. 악용 가능한 버퍼 오버플로가 존재하기 때문에 크래커는 자동화된 도구를 사용하여 취약점을 가진 시스템을 식별하고 액세스 권한이 확보되면 자동화된 루트킷을 사용하여 시스템에 대한 액세스를 유지합니다.



참고

x86 호환 가능 메모리 분할 및 보호 기술인 **x86** 호환 유니 및 멀티 프로세서 커널에서 지원하는 실행 가능한 메모리 분할 및 보호 기술인 **Red Hat Enterprise Linux 7**에서 버퍼 오버플로 취약점의 위협을 완화합니다. **ExecShield**는 가상 메모리를 실행 파일과 실행 불가능한 세그먼트로 분리하여 버퍼 오버플로의 위험을 줄입니다. 실행 파일 세그먼트 외부에서 실행하려고 하는 프로그램 코드(예: 버퍼 오버플로 악용에서 삽입된 악성 코드)는 세그먼트 오류와 종료됩니다.**Any program code that tries to execute outside of the executable segment (such as malicious code injection from a buffer overflow exploit) triggers a segmentation fault and terminates.**

ExecShield에는 **AMD64** 플랫폼 및 **Intel® 64** 시스템의 **No eXecute (NX)** 기술 지원도 포함되어 있습니다. 이러한 기술은 **ExecShield**와 함께 작동하여 악성 코드가 실행 코드 **4KB**의 세분성으로 가상 메모리의 실행 가능한 부분에서 실행되지 않도록 하여 버퍼 오버플로 악용에 대한 공격 위험을 줄입니다.

중요

네트워크를 통한 공격에 대한 노출을 제한하려면 사용되지 않은 모든 서비스를 해제해야 합니다.

4.3.2. 서비스 식별 및 구성

보안을 강화하기 위해 **Red Hat Enterprise Linux 7**과 함께 설치된 대부분의 네트워크 서비스는 기본적으로 해제되어 있습니다. 그러나 몇 가지 주목할 만한 예외가 있습니다.

- **cups - Red Hat Enterprise Linux 7**의 기본 인쇄 서버.
- **cups-lpd -** 대체 인쇄 서버.
- **xinetd - gssftp** 및 **telnet** 과 같은 다양한 하위 서버에 대한 연결을 제어하는 슈퍼 서버입니다.
- **sshd - Telnet**을 안전하게 대체하는 **OpenSSH** 서버입니다.

이러한 서비스를 실행할지 여부를 결정할 때는 일반적인 방법을 사용하고 위험을 감수하지 않는 것이 가장 좋습니다. 예를 들어 프린터를 사용할 수 없는 경우 **cups** 를 실행 상태로 두지 마십시오. **portreserve** 에서도 마찬가지입니다. **NFSv3** 볼륨을 마운트하지 않거나 **NIS(ypbind** 서비스)를 사용하는 경우, **INPUT bind** 를 비활성화해야 합니다. 부팅 시 어떤 네트워크 서비스를 시작할 수 있는지 확인하는 것만으로는 부족합니다. 열려 있고 수신 대기 중인 포트도 확인하는 것이 좋습니다. 자세한 내용은 **4.4.2 절. "Which Ports Are Listening 확인"**을 참조하십시오.

4.3.3. 비보안 서비스

잠재적으로 모든 네트워크 서비스는 안전하지 않습니다. 따라서 사용되지 않는 서비스를 해제하는 것이 매우 중요합니다. 서비스에 대한 악용은 정기적으로 공개되고 패치되므로 모든 네트워크 서비스와 관련된 패키지를 정기적으로 업데이트하는 것이 매우 중요합니다. 자세한 내용은 **3장. 시스템을 최신 상태로 유지**를 참조하십시오.

일부 네트워크 프로토콜은 본질적으로 다른 프로토콜보다 안전하지 않습니다. 여기에는 다음과 같은 모든 서비스가 포함됩니다.

- 전송 사용자 이름 및 비밀번호를 네트워크 분리 **- Telnet** 및 **FTP**와 같은 많은 이전 프로토콜

에서는 인증 세션을 암호화하지 말고 가능한 경우 피해야 합니다.

- **Sensitive Data Over a Network Unencrypted -** 많은 프로토콜은 암호화되지 않은 네트워크를 통해 데이터를 전송합니다. 이러한 프로토콜에는 **Telnet, FTP, HTTP** 및 **SMTP**가 포함됩니다. **NFS** 및 **SMB**와 같은 많은 네트워크 파일 시스템도 암호화되지 않은 네트워크를 통해 정보를 전송합니다. 이러한 프로토콜을 사용하여 전송되는 데이터 유형을 제한할 때 사용자의 책임입니다.

본질적으로 안전하지 않은 서비스의 예로는 **rlogin,rsh,telnet, vsftpd** 가 있습니다.

SSH 를 사용하는 모든 원격 로그인 및 쉘 프로그램(**rlogin,rsh, telnet**)을 피해야 합니다. **sshd** 에 대한 자세한 내용은 **4.3.11절. "SSH 보안"** 을 참조하십시오.

FTP 는 원격 셸에서 시스템 보안에 본질적으로 위험하지는 않지만 문제를 피하기 위해 **FTP** 서버를 신중하게 구성하고 모니터링해야합니다. **FTP** 서버 보안에 대한 자세한 내용은 **4.3.9절. "FTP 보안"** 을 참조하십시오.

방화벽 뒤에서 신중하게 구현해야 하는 서비스는 다음과 같습니다.

- **auth**
- **nfs-server**
- **SMB** 및 **nbm (Samba)**
- **yppasswdd**
- **ypserv**
- **ypxfrd**

네트워크 서비스 보안에 대한 자세한 내용은 **4.4절. "네트워크 액세스 보안"** 에서 확인할 수 있습니다.

4.3.4. rpcbind 보안

custom bind 서비스는 **NIS** 및 **NFS**와 같은 **RPC** 서비스에 대한 동적 포트 할당 데몬입니다. 약한 인증 메커니즘을 가지고 있으며, 제어하는 서비스에 다양한 포트를 할당할 수 있습니다. 이러한 이유로 보안을 유지하기가 어렵습니다.

참고

NFSv4에는 더 이상 필요하지 않기 때문에 **Diffie bind** 는 **NFSv2** 및 **NFSv3** 구현에만 영향을 미칩니다. **NFSv2** 또는 **NFSv3** 서버를 구현하려는 경우 **INPUTbind** 가 필요하며 다음 섹션이 적용됩니다.

RPC 서비스를 실행하는 경우 다음 기본 규칙을 따릅니다.

4.3.4.1. TCP Wrappers를 사용하여 iLObind 보호

TCP Wrappers를 사용하여 인증의 기본 형태가 없기 때문에 네트워크 또는 호스트에 대한 액세스 권한이 있는 네트워크를 제한하는 것이 중요합니다.

또한 서비스에 대한 액세스를 제한할 때만 **IP** 주소를 사용합니다. 호스트 이름은 **DNS Poisoning** 및 기타 방법으로 위조될 수 있으므로 사용하지 마십시오.

4.3.4.2. firewalld를 사용하여 iLObind 보호

RuntimeConfig bind 서비스에 대한 액세스를 추가로 제한하려면 서버에 **firewalld** 규칙을 추가하고 특정 네트워크에 대한 액세스를 제한하는 것이 좋습니다.

다음은 **firewalld** 풍부한 언어 명령의 두 가지 예입니다. 첫 번째에서는 **192.168.0.0/24** 네트워크에서 사용 중인 포트 **111** 에 대한 **TCP** 연결을 허용합니다. 두 번째는 **localhost**에서 동일한 포트에 대한 **TCP** 연결을 허용합니다. 다른 모든 패킷은 삭제됩니다.

```
~]# firewall-cmd --add-rich-rule='rule family="ipv4" port port="111" protocol="tcp" source
address="192.168.0.0/24" invert="True" drop'
~]# firewall-cmd --add-rich-rule='rule family="ipv4" port port="111" protocol="tcp" source
address="127.0.0.1" accept'
```

UDP 트래픽을 동일하게 제한하려면 다음 명령을 사용하십시오.

```
~]# firewall-cmd --add-rich-rule='rule family="ipv4" port port="111" protocol="udp" source
address="192.168.0.0/24" invert="True" drop'
```

참고

firewalld 풍부한 언어 명령에 **--permanent** 를 추가하여 설정을 영구적으로 만듭니다. 방화벽 구현에 대한 자세한 내용은 **5장. 방화벽 사용** 을 참조하십시오.

4.3.5. rpc.mountd 보안

EgressIP .mountd 데몬은 **NFS** 버전 **2(RFC194) 및 NFS** 버전 **3(RFC1813)**에서 사용하는 프로토콜인 **NFS MOUNT** 프로토콜의 서버 측을 구현합니다.

RPC 서비스를 실행하는 경우 다음 기본 규칙을 따릅니다.

4.3.5.1. TCP Wrappers를 사용하여 iLO.mountd 보호

TCP Wrappers를 사용하여 인증의 기본 제공 형태가 없기 때문에 네트워크 또는 호스트에 대한 액세스 권한이 있는 네트워크를 제한하는 것이 중요합니다.

또한 서비스에 대한 액세스를 제한할 때만 **IP** 주소를 사용합니다. 호스트 이름은 **DNS Poisoning** 및 기타 방법으로 위조될 수 있으므로 사용하지 마십시오.

4.3.5.2. firewalld를 사용하여 InstallPlan.mountd 보호

KnativeServing .mountd 서비스에 대한 액세스를 추가로 제한하려면 서버에 **firewalld** 풍부한 언어 규칙을 추가하고 특정 네트워크에 대한 액세스를 제한합니다.

다음은 **firewalld** 풍부한 언어 명령의 두 가지 예입니다. 첫 번째는 **192.168.0.0/24** 네트워크에서 **mountd** 연결을 허용합니다. 두 번째는 로컬 호스트의 **mountd** 연결을 허용합니다. 다른 모든 패킷은 삭제됩니다.

```
~]# firewall-cmd --add-rich-rule 'rule family="ipv4" source NOT address="192.168.0.0/24" service
name="mountd" drop'
~]# firewall-cmd --add-rich-rule 'rule family="ipv4" source address="127.0.0.1" service
name="mountd" accept'
```

참고

firewalld 풍부한 언어 명령에 --permanent 를 추가하여 설정을 영구적으로 만듭니다. 방화벽 구현에 대한 자세한 내용은 5장. 방화벽 사용 을 참조하십시오.

4.3.6. NIS 보안

네트워크 정보 서비스 (NIS)는 ypserv 라는 RPC 서비스로, 사용자 이름, 암호 및 기타 관련 서비스의 맵을 해당 도메인 내에 있다고 주장하는 모든 컴퓨터에 대한 사용자 이름, 암호 및 기타 민감한 정보를 배포하는 데 사용됩니다.

NIS 서버는 여러 애플리케이션으로 구성됩니다. 여기에는 다음이 포함됩니다.

- /usr/sbin/rpc.yppasswdd - yppasswdd 서비스라고도 하는 이 데몬을 사용하면 사용자가 NIS 암호를 변경할 수 있습니다.
- /usr/sbin/rpc.ypxfrd - ypxfrd 서비스라고도 하며, 이 데몬은 네트워크를 통해 NIS 맵을 전송합니다.
- /usr/sbin/ypserv - NIS 서버 데몬입니다.

NIS는 오늘날의 표준에 따라 다소 안전하지 않습니다. 호스트 인증 메커니즘은 없으며 암호 해시를 포함하여 암호화되지 않은 네트워크를 통해 모든 정보를 전송합니다. 따라서 NIS를 사용하는 네트워크를 설정할 때 매우 주의해야 합니다. 이는 NIS의 기본 구성이 본질적으로 안전하지 않다는 사실로 더 복잡해집니다.

NIS 서버를 구현하려는 모든 사용자는 먼저 4.3.4절. “rpcbind 보안” 에 설명된 대로 openvswitch bind 서비스를 보안한 다음 네트워크 계획과 같은 다음 문제를 해결하는 것이 좋습니다.

4.3.6.1. 네트워크 계획을 주의 깊게 계획

NIS는 중요한 정보를 네트워크를 통해 암호화되지 않은 상태로 전송하므로 방화벽 뒤에서 서비스를 실행하고 세그먼트화된 보안 네트워크에서 서비스를 실행하는 것이 중요합니다. NIS 정보가 안전하지 않은 네트워크를 통해 전송될 때마다 가로채는 위험이 있습니다. 신중한 네트워크 설계는 심각한 보안 위반을 방지하는 데 도움이 될 수 있습니다.

4.3.6.2. 암호와 같은 NIS 도메인 이름 및 호스트 이름 사용

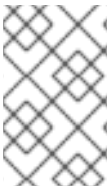
NIS 도메인 내 모든 시스템은 명령을 사용하여 사용자가 NIS 서버의 DNS 호스트 이름과 NIS 도메인 이름을 알고 있는 한 인증 없이 서버에서 정보를 추출할 수 있습니다.

예를 들어, 랩탑 컴퓨터를 네트워크에 연결하거나 외부 네트워크에서 (및 내부 IP 주소를 스푸핑하기 위해 관리) 다음 명령을 수행하면 /etc/passwd 맵이 표시됩니다.

```
ypcat -d <NIS_domain> -h <DNS_hostname> passwd
```

이 공격자가 root 사용자인 경우 다음 명령을 입력하여 /etc/shadow 파일을 얻을 수 있습니다.

```
ypcat -d <NIS_domain> -h <DNS_hostname> shadow
```



참고

Kerberos를 사용하는 경우 /etc/shadow 파일은 NIS 맵에 저장되지 않습니다.

NIS에 액세스하는 공격자가 더 어려워지도록 하려면 o7hfawtgmhwg.domain.com 과 같은 DNS 호스트 이름에 대해 임의의 문자열을 만듭니다. 마찬가지로, 다른 무작위로 NIS 도메인 이름을 생성합니다. 이로 인해 공격자가 NIS 서버에 액세스하기가 훨씬 더 어려워집니다.

4.3.6.3. /var/yp/securenets 파일 편집

/var/yp/securenets 파일이 비어 있거나 존재하지 않는 경우(기본 설치 이후의 경우) NIS는 모든 네트워크를 수신 대기합니다. 가장 먼저해야 할 일 중 하나는 ypserv 이 적절한 네트워크의 요청에만 응답하도록 파일에 넷마스크/네트워크 쌍을 배치하는 것입니다.

다음은 /var/yp/securenets 파일의 샘플 항목입니다.

```
255.255.255.0     192.168.0.0
```

주의

/var/yp/securenets 파일을 생성하지 않고 처음으로 NIS 서버를 시작하지 마십시오.

이 기술은 IP 스푸핑 공격으로부터 보호를 제공하지 않지만 NIS 서버 서비스의 네트워크에는 최소한 제한이 있습니다.

4.3.6.4. 정적 포트 할당 및 언어 규칙 사용

NIS와 관련된 모든 서버에는 INPUT .yppasswdd를 제외한 특정 포트에 할당할 수 있습니다. - 사용자가 로그인 암호를 변경할 수 있는 데몬입니다. 포트를 다른 두 NIS 서버 데몬인 INPUT .ypxfrd 및 ypserv에 지정하면 방화벽 규칙을 생성하여 NIS 서버 데몬을 침입자로부터 추가로 보호할 수 있습니다.

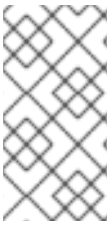
이렇게 하려면 /etc/sysconfig/network 에 다음 행을 추가합니다.

```
YPSERV_ARGS="-p 834"
YPXFRD_ARGS="-p 835"
```

그런 다음 다음과 같은 풍부한 언어 firewalld 규칙을 사용하여 서버가 이러한 포트를 수신 대기하는 네트워크를 적용할 수 있습니다.

```
~]# firewall-cmd --add-rich-rule='rule family="ipv4" source address="192.168.0.0/24" invert="True"
port port="834-835" protocol="tcp" drop'
~]# firewall-cmd --add-rich-rule='rule family="ipv4" source address="192.168.0.0/24" invert="True"
port port="834-835" protocol="udp" drop'
```

즉, 서버는 192.168.0.0/24 네트워크에서 요청이 제공된 경우에만 포트 834 및 835에 대한 연결만 허용합니다. 첫 번째 규칙은 TCP 및 UDP 용 두 번째 규칙입니다.



참고

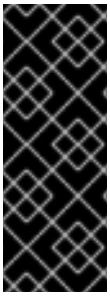
iptables 명령을 사용하여 방화벽 구현에 대한 자세한 내용은 5장. 방화벽 사용 을 참조하십시오.

4.3.6.5. Kerberos 인증 사용

NIS를 인증에 사용할 때 고려해야 할 문제 중 하나는 사용자가 시스템에 로그인할 때마다 /etc/shadow 맵의 암호 해시가 네트워크를 통해 전송된다는 것입니다. 침입자가 NIS 도메인에 액세스하고 네트워크 트래픽을 스니핑하는 경우 사용자 이름 및 암호 해시를 수집할 수 있습니다. 충분한 시간을 두고 암호 크래킹 프로그램은 취약한 암호를 추측할 수 있으며 공격자는 네트워크에서 유효한 계정에 액세스할 수 있습니다.

Kerberos는 비밀 키 암호화를 사용하므로 네트워크를 통해 암호 해시가 전송되지 않으므로 시스템의 보안을 훨씬 더 안전하게 유지할 수 있습니다. Kerberos에 대한 자세한 내용은 Linux 도메인 ID, 인증 및 정책 가이드의 Kerberos 사용을 위한 IdM로 로깅 섹션을 참조하십시오.

4.3.7. NFS 보안



중요

NFS 트래픽은 모든 버전에서 TCP를 사용하여 보낼 수 있으며, NFSv4를 사용하는 경우 NFSv3 대신 NFSv3와 함께 사용해야 합니다. 모든 버전의 NFS는 RPCSEC_GSS 커널 모듈의 일부로 Kerberos 사용자 및 그룹 인증을 지원합니다. Red Hat Enterprise Linux 7은 Diffie bind 를 사용하는 NFSv3를 지원하므로 사용자도 계속 포함되어 있습니다.

4.3.7.1. 네트워크 계획을 주의 깊게 계획

NFSv2 및 NFSv3는 일반적으로 데이터를 안전하지 않은 상태로 전달했습니다. 이제 모든 버전의 NFS에서는 Kerberos를 사용하여 일반 파일 시스템 작업을 인증(및 선택적으로 암호화)할 수 있습니다. NFSv4에서는 모든 작업에서 Kerberos를 사용할 수 있습니다. NFSv2 또는 NFSv3, 파일 잠금 및 마운트에서는 여전히 이를 사용하지 않습니다. NFSv4.0을 사용하는 경우 클라이언트가 NAT 또는 방화벽 뒤에 있으면 위임이 꺼질 수 있습니다. NFSv4.1을 사용하여 NAT 및 방화벽을 통해 위임을 처리하는 방법에 대한 자세한 내용은 Red Hat Enterprise Linux 7 Storage Administration Guide의 pNFS 섹션을 참조하십시오.

4.3.7.2. NFS 마운트 옵션 보안

/etc/fstab 파일에서 마운트 명령 사용은 Red Hat Enterprise Linux 7 스토리지 관리 가이드 의 마운트 명령 사용 장에 설명되어 있습니다. 보안 관리 관점에서는 사용자 지정 기본 옵션을 설정하는 데 사용할 수 있는 /etc/nfsmount.conf 에 NFS 마운트 옵션을 지정할 수도 있습니다.

4.3.7.2.1. NFS 서버 확인

주의

*전체 파일 시스템만 내보냅니다. 파일 시스템의 하위 디렉터리를 내보내는 것은 보안 문제가 될 수 있습니다. 경우에 따라 클라이언트가 파일 시스템의 내보낸 부분을 "진행"하여 내보내기 해제한 부분으로 이동할 수 있습니다(**exports(5)** 도움말 페이지의 하위 트리 섹션 참조).*

*마운트된 파일 시스템에 쓸 수 있는 사용자 수를 줄이기 위해 가능한 경우 항상 파일 시스템을 읽기 전용으로 내보내려면 **ro** 옵션을 사용합니다. 특별히 필요한 경우에만 **rw** 옵션만 사용하십시오. 자세한 내용은 **man exports(5)** 페이지를 참조하십시오. 쓰기 액세스를 허용하면 다음과 같은 심볼릭 링크 공격의 위험이 증가합니다. 여기에는 **/tmp** 및 **/usr /tmp** 와 같은 임시 디렉토리가 포함됩니다.*

***rw** 옵션을 사용하여 디렉터리를 마운트해야 하는 경우 위험을 줄이기 위해 가능한 경우 항상 디렉터리를 사용할 수 없도록 합니다. 일부 애플리케이션에서 홈 디렉터리를 내보내는 것은 일반 텍스트로 암호를 저장하거나 약하게 암호화되어 있기 때문에 위험도로 볼 수 있습니다. 애플리케이션 코드를 검토 및 개선할 때 이러한 위험이 감소되고 있습니다. 일부 사용자는 **SSH** 키에 암호를 설정하지 않으므로 홈 디렉터리도 위험이 있음을 의미합니다. 암호 사용을 강제하거나 **Kerberos**를 사용하면 해당 위험이 완화됩니다.*

*액세스 권한이 필요한 클라이언트에만 내보내기를 제한합니다. **NFS** 서버에서 **showmount -e** 명령을 사용하여 서버를 내보내는 작업을 검토합니다. 특히 필요하지 않은 항목을 내보내지 마십시오.*

***no_root_squash** 옵션을 사용하지 말고 기존 설치를 검토하여 사용되지 않는지 확인합니다. 자세한 내용은 **4.3.7.4**절. “**no_root_squash** 옵션을 사용하지 마십시오.”를 참조하십시오.*

***secure** 옵션은 “예약된” 포트로 내보내기를 제한하는 데 사용되는 서버 측 내보내기 옵션입니다. 기본적으로 서버는 기존 클라이언트(예: 커널 내 **NFS** 클라이언트)만 “신뢰할 수 있는” 코드(예: 커널 내 **NFS** 클라이언트)만 허용했기 때문에 “예약된” 포트(**1024** 미만의 포트)에서만 클라이언트 통신을 허용합니다. 그러나 많은 네트워크에서 일부 클라이언트에서 **root**가 되는 것은 어렵지 않으므로 서버에서 예약된 포트의 통신이 권한이 있다고 가정하는 것은 거의 안전하지 않습니다. 따라서 예약된 포트에 대한 제한은 제한된 값입니다. **Kerberos**, 방화벽 및 특정 클라이언트에 대한 내보내기 제한에 의존하는 것이 좋습니다.*

*대부분의 클라이언트는 가능한 경우 예약된 포트를 사용합니다. 그러나 예약된 포트는 제한된 리소스이므로 클라이언트(특히 많은 **NFS** 마운트가 있는 포트)도 높은 번호의 포트를 사용하도록 선택할 수 있습니다. **Linux** 클라이언트는 “**noresvport**” 마운트 옵션을 사용하여 이 작업을 수행할 수 있습니다. 내보내기에서 이를 허용하려면 “비보안” 내보내기 옵션을 사용하여 이 작업을 수행할 수 있습니다.*

*사용자가 서버에 로그인할 수 없도록 하는 것이 좋습니다. **NFS** 서버에서 위 설정을 검토하는 동안 서버와 액세스할 수 있는 사용자 및 항목을 검토합니다.*

4.3.7.2.2. NFS 클라이언트 확인

***nosuid** 옵션을 사용하여 **setuid** 프로그램의 사용을 허용하지 않습니다. **nosuid** 옵션은 **set-user-identifier** 또는 **set-group-identifier** 비트를 비활성화합니다. 이렇게 하면 원격 사용자가 **setuid** 프로그램을 실행하여 더 높은 권한을 얻을 수 없습니다. 클라이언트 및 서버 측에서 이 옵션을 사용합니다.*

***noexec** 옵션은 클라이언트의 모든 실행 파일을 비활성화합니다. 이를 사용하여 사용자가 공유 중인 파일 시스템에 저장된 파일을 실수로 실행하지 못하도록 합니다. **nosuid** 및 **noexec** 옵션은 대부분 파일 시스템이 아닌 경우 표준 옵션입니다.*

***nodev** 옵션을 사용하여 “**device-files**” 가 클라이언트에서 하드웨어 장치로 처리되지 않도록 합니다.*

***resvport** 옵션은 클라이언트 측 마운트 옵션이며 **secure** 는 해당 서버 측 내보내기 옵션입니다(위의 설명 참조). "예약 포트"와의 통신을 제한합니다. 예약 또는 "알려진" 포트는 **root** 사용자와 같은 권한 있는 사용자 및 프로세스를 위해 예약되어 있습니다. 이 옵션을 설정하면 클라이언트가 예약된 소스 포트를 사용하여 서버와 통신합니다.*

*모든 버전의 **NFS**에서는 이제 **Kerberos** 인증을 사용한 마운트를 지원합니다. 이를 활성화하는 마운트 옵션은 **sec=krb5** 입니다.*

***NFSv4**에서는 무결성을 위해 **krb5i** 를 사용한 **Kerberos**와 개인 정보 보호를 위해 **krb5p** 를 사용한 마운트를 지원합니다. 이는 **sec=krb5** 로 마운트할 때 사용되지만 **NFS** 서버에서 구성해야 합니다. 자세한 내용은 내보내기(내장**5** 내보내기)의 도움말 페이지를 참조하십시오.*

***NFS** 도움말 페이지(**man 5 nfs**)에는 **NFSv4**의 보안 개선 사항을 설명하고 모든 **NFS** 특정 마운트 옵션이 포함되어 있는 “**SECURITY CONSIDERATIONS**” 섹션이 있습니다.*

중요

***krb5-libs** 패키지에서 제공하는 **MIT Kerberos** 라이브러리는 새 배포에서 **DES(Data Encryption Standard)** 알고리즘을 사용하는 것을 지원하지 않습니다. 보안과 특정 호환성 상의 이유로 **DES**는 **Kerberos** 라이브러리에서 기본적으로 더 이상 사용되지 않으며 비활성화됩니다. 환경이 새로운 보안 알고리즘을 지원하지 않는 경우에만 **DES**를 사용하십시오.*

4.3.7.3. Syntax 오류 경고

NFS 서버는 내보낼 파일 시스템과 이러한 디렉터리를 /etc/exports 파일을 참조하여 내보낼 호스트를 결정합니다. 이 파일을 편집할 때 관련 공백을 추가하지 않도록 주의하십시오.

예를 들어 /etc/exports 파일의 다음 줄은 /tmp/nfs/ 디렉토리를 읽기/쓰기 권한을 사용하여 호스트 bob.example.com 에 공유합니다.

```
/tmp/nfs/     bob.example.com(rw)
```

반면 /etc/exports 파일의 다음 줄은 동일한 디렉토리를 호스트 bob.example.com 과 읽기 전용 권한으로 공유하고 호스트 이름의 단일 공간 문자로 인해 읽기 전용 권한으로 공유합니다.

```
/tmp/nfs/     bob.example.com (rw)
```

showmount 명령을 사용하여 공유 중인 내용을 확인하여 구성된 NFS 공유를 확인하는 것이 좋습니다.

```
showmount -e <hostname>
```

4.3.7.4. no_root_squash 옵션을 사용하지 마십시오.

기본적으로 NFS 공유는 root 사용자를 권한이 없는 사용자 계정인 nfsnobody 사용자로 변경합니다. 이렇게 하면 모든 루트가 생성한 파일의 소유자가 nfsnobody 로 변경되어 setuid bit가 설정된 프로그램 업로드를 방지할 수 있습니다.

no_root_squash 를 사용하는 경우 원격 루트 사용자는 공유 파일 시스템의 파일을 변경하고 다른 사용자가 의도하지 않게 실행할 수 있도록 Trojans에 의해 감염된 애플리케이션을 남겨 둘 수 있습니다.

4.3.7.5. NFS 방화벽 설정

NFSv4는 Red Hat Enterprise Linux 7의 기본 NFS 버전이며 TCP용으로만 포트 2049를 열어야 합니다. NFSv3를 사용하는 경우 아래에 설명된 대로 4개의 추가 포트가 필요합니다.

NFSv3의 포트 구성

NFS에 사용되는 포트는168 bind 서비스에 의해 동적으로 할당되므로 방화벽 규칙을 만들 때 문제가 발생할 수 있습니다. 이 프로세스를 단순화하려면 /etc/sysconfig/nfs 파일을 사용하여 사용할 포트를 지정합니다.

- *MOUNTD_PORT - 마운트를 위한 TCP 및 UDP 포트 (rpc.mountd)*
- *STATD_PORT - 상태용 TCP 및 UDP 포트 (rpc.statd)*

Red Hat Enterprise Linux 7에서 /etc/modprobe.d/lockd.conf 파일에 NFS 잠금 관리자(nlockmgr)의 TCP 및 UDP 포트를 설정합니다.

- *nlm_tcpport - nlockmgr의 TCP 포트 (rpc.lockd)*
- *nlm_udpport - UDP 포트 nlockmgr (rpc.lockd)*

지정된 포트 번호는 다른 서비스에서 사용해서는 안 됩니다. 포트 번호를 지정하고 TCP 및 UDP 포트 2049(NFS)를 허용하도록 방화벽을 구성합니다. 사용자 지정 가능한 NFS 잠금 관리자 매개변수에 대한 설명은 /etc/modprobe.d/lockd.conf 를 참조하십시오.

NFS 서버에서 gRPC info -p 명령을 실행하여 사용 중인 포트와 RPC 프로그램을 확인합니다.

4.3.7.6. Red Hat Identity Management로 NFS 보안

Kerberos 인식 NFS 설정은 Red Hat Enterprise Linux에 포함된 Red Hat Identity Management를 사용하는 환경에서 크게 단순화될 수 있습니다.

Red Hat Identity Management를 사용할 때 Kerberos를 사용하여 NFS 를 보호하는 방법을 알아보려면 Red Hat Enterprise Linux 7 Linux 도메인 ID, 인증 및 정책 가이드 를 참조하십시오.

4.3.8. HTTP 서버 보안

4.3.8.1. Apache HTTP Server 보안

Apache HTTP 서버는 Red Hat Enterprise Linux 7에서 가장 안정적이고 안전한 서비스 중 하나입니다. 많은 수의 옵션과 기술을 사용하여 Apache HTTP Server를 보호할 수 있습니다. 다음 섹션에서는 Apache HTTP 서버를 실행할 때 모범 사례를 간략하게 설명합니다.

프로덕션 환경에 배치하기 전에 시스템에서 실행 중인 스크립트가 의도한 대로 작동하는지 항상 확인

*하십시오. 또한 **root** 사용자만 스크립트 또는 **CGI**를 포함하는 디렉토리에 쓰기 권한이 있는지 확인합니다. 이렇게 하려면 **root** 사용자로 다음 명령을 입력합니다.*

```
chown root <directory_name>
```

```
chmod 755 <directory_name>
```

*시스템 관리자는 다음 구성 옵션을 사용할 때 주의해야 합니다(**/etc/httpd/conf/httpd.conf**에서 구성됨).*

FollowSymLinks

*이 지시문은 기본적으로 활성화되어 있으므로 웹 서버의 문서 루트에 대한 심볼릭 링크를 생성할 때 주의하십시오. 예를 들어 **/** 에 대한 심볼릭 링크를 제공하는 것은 좋지 않습니다.*

Indexes

이 지시문은 기본적으로 활성화되어 있지만 바람직하지 않을 수 있습니다. 사용자가 서버에서 파일을 검색하지 못하도록 하려면 이 지시문을 제거합니다.

UserDir

***UserDir** 지시문은 시스템에 사용자 계정이 있는지 확인할 수 있으므로 기본적으로 비활성화되어 있습니다. 서버에서 사용자 디렉터리 검색을 활성화하려면 다음 지시문을 사용합니다.*

```
UserDir enabled
    UserDir disabled root
```

*이러한 지시문은 **/root/** 가 아닌 모든 사용자 디렉터리를 검색하는 사용자 디렉토리를 활성화합니다. 비활성화된 계정 목록에 사용자를 추가하려면 **UserDir disabled** 줄에 공백으로 구분된 사용자 목록을 추가합니다.*

ServerTokens

***ServerTokens** 지시문은 클라이언트에 다시 전송되는 서버 응답 헤더 필드를 제어합니다. 여기에는 다음 매개변수를 사용하여 사용자 지정할 수 있는 다양한 정보가 포함됩니다.*

- ***ServerTokens Full** (기본 옵션) **-** 사용 가능한 모든 정보(**OS** 유형 및 사용된 모듈)를 제공합니다. 예를 들면 다음과 같습니다.*

```
Apache/2.0.41 (Unix) PHP/4.2.2 MyMod/1.2
```

- **ServerTokens Prod** 또는 **ServerTokens ProductOnly** - 다음 정보를 제공합니다.

```
Apache
```

- **ServerTokens major** - 다음 정보를 제공합니다.

```
Apache/2
```

- **ServerTokens Minor** - 다음 정보를 제공합니다.

```
Apache/2.0
```

- **ServerTokens Min** 또는 **ServerTokens Minimal** - 다음과 같은 정보를 제공합니다.

```
Apache/2.0.41
```

- **ServerTokens OS** - 다음 정보를 제공합니다.

```
Apache/2.0.41 (Unix)
```

가능한 공격자가 시스템에 대한 중요한 정보를 얻지 못하도록 **ServerTokens Prod** 옵션을 사용하는 것이 좋습니다.

중요

IncludesNoExec 지시문을 제거하지 마십시오. 기본적으로 **Server-Side Includes (SSI)** 모듈은 명령을 실행할 수 없습니다. 잠재적으로 공격자가 시스템에서 명령을 실행할 수 있으므로 반드시 필요한 경우가 아니면 이 설정을 변경하지 않는 것이 좋습니다.

httpd 모듈 제거

특정 시나리오에서는 특정 **httpd** 모듈을 제거하여 **HTTP** 서버의 기능을 제한하는 것이 좋습니다. 이를 위해 **/etc/httpd/conf.modules.d** 디렉터리에서 구성 파일을 편집합니다. 예를 들어 프록시 모듈을 제거하려면 다음을 수행합니다.

```
echo '# All proxy modules disabled' > /etc/httpd/conf.modules.d/00-proxy.conf
```

/etc/httpd/conf.d/ 디렉터리에는 모듈을 로드하는 데 사용되는 구성 파일이 포함되어 있습니다.

httpd 및 SELinux

자세한 내용은 Red Hat Enterprise Linux 7 SELinux 사용자 및 관리자 가이드의 Apache HTTP 서버 및 SELinux 장을 참조하십시오.

4.3.8.2. secrets 보안

NGINX는 고성능 **HTTP** 및 프록시 서버입니다. 이 섹션에서는 **wget** 구성을 강화시키는 추가 단계를 간략하게 설명합니다. **wget** 구성 파일의 **server** 섹션에서 다음 구성 변경 사항을 모두 수행합니다.

버전 문자열 비활성화

공격자가 서버에서 실행 중인**ngressController**의 버전을 학습하지 못하도록 하려면 다음 설정 옵션을 사용합니다.

```
server_tokens       off;
```

이는 버전 번호의 제거의 효과를 일으키고,부분에 의해 제공되는 모든 요청에서 **nginx** 문자열을 보고하기만 하면 됩니다:

```
$ curl -sI http://localhost | grep Server
Server: nginx
```

추가 보안 관련 헤더 포함

RuntimeClass가 제공하는 각 요청에는 알려진 특정 웹 애플리케이션 취약점을 완화하는 추가 **HTTP** 헤더가 포함될 수 있습니다.

- **add_header X-Frame-Options-02-EORIGIN; -** 이 옵션은 도메인 외부의 모든 페이지를 거부하여**Period**에서 제공하는 모든 콘텐츠를 프레임하고 효과적으로 클릭재킹 공격을 완화합니다.
- **add_header X-Content-Type-Options nosniff; -** 이 옵션은 이전 브라우저의 **MIME** 유형 스니핑을 방지합니다.
- **add_header X-XSS-Protection "1; mode=block"; -** 이 옵션을 사용하면 **clientSS(Cross-Site Scripting)** 필터링이 가능해지지 않게 했으며, 이 옵션을 사용하면 **browser**가 **rfc**에 응답에 포함된 잠재적으로 악의적인 콘텐츠를 렌더링하지 못하게 됩니다.

Potentially Harmful HTTP 방법 비활성화

활성화된 경우 일부 **HTTP** 방법에서는 공격자가 웹 애플리케이션을 테스트하도록 설계된 웹 서버에서 작업을 수행할 수 있습니다. 예를 들어 **TRACE** 방법은 **XST(Cross-Site Tracing)**를 허용하는 것으로 알려져 있습니다.

NetNamespace 서버에서는 이러한 유해한 **HTTP** 방법뿐만 아니라 허용되어야 하는 항목만 허용하여 임의의 방법을 허용할 수 있습니다. 예를 들어 다음과 같습니다.

```
# Allow GET, PUT, POST; return "405 Method Not Allowed" for all others.
if ( $request_method !~ ^(GET|PUT|POST)$ ) {
    return 405;
}
```

SSL 구성

NGINX 웹 서버에서 제공하는 데이터를 보호하려면 **HTTPS**를 통해서만 제공하는 데이터를 보호하는 것이 좋습니다. **8.9** 서버에서 **SSL**을 사용하도록 설정하는 데 필요한 보안 구성 프로필을 생성하려면 **Mozilla SSL** 구성 생성기 를 참조하십시오. 생성된 구성을 사용하면 알려진 취약한 프로토콜(예: **SSLv2** 또는 **SSLv3**), 암호 및 해싱 알고리즘(예: **3DES** 또는 **MD5**)이 비활성화됩니다.

SSL 서버 테스트를 사용하여 구성이 최신 보안 요구 사항을 충족하는지 확인할 수도 있습니다.

4.3.9. FTP 보안

FTP(File Transfer Protocol)는 네트워크를 통해 파일을 전송하도록 설계된 이전 **TCP** 프로토콜입니다. 사용자 인증을 포함한 서버의 모든 트랜잭션은 암호화되지 않으므로 안전하지 않은 프로토콜로 간주되므로 신중하게 구성해야 합니다.

Red Hat Enterprise Linux 7은 두 개의 **FTP** 서버를 제공합니다.

- **Red Hat Content Accelerator (tux) - FTP** 기능을 갖춘 커널 공간 웹 서버.
- **vsftpd - FTP** 서비스의 독립 실행형 보안 지향 구현입니다.

다음 보안 지침은 **vsftpd FTP** 서비스를 설정하기 위한 것입니다.

4.3.9.1. FTPRuntimeConfig Banner

사용자 이름과 암호를 제출하기 전에 모든 사용자에게 인사 배너가 표시됩니다. 기본적으로 이 배너에는 시스템의 약점을 확인하려고 하는 크래커에 유용한 버전 정보가 포함되어 있습니다.

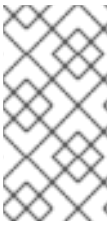
vsftpd 의 인사 배너를 변경하려면 다음 지시문을 **/etc/COMPLETE/vsftpd.conf** 파일에 추가하십시오.

```
ftpd_banner=<insert_greeting_here>
```

위의 지시문에서 **< insert_greeting_here >**를 인사 메시지 텍스트로 바꿉니다.

mutli-line banners의 경우 배너 파일을 사용하는 것이 가장 좋습니다. 여러 배너의 관리를 단순화하려면 모든 배너를 **/etc/banners/** 라는 새 디렉토리에 배치합니다. 이 예에서 **FTP** 연결의 배너 파일은 **/etc/banners/ftp.msg** 입니다. 다음은 이러한 파일이 어떻게 표시되는지 보여주는 예입니다.

```
########## Hello, all activity on ftp.example.com is logged. ##########
```



참고

4.4.1절. “TCP Wrappers 및 xinetd를 사용하여 서비스 보안” 에 지정된 대로 **220** 로 파일의 각 줄을 시작할 필요가 없습니다.

vsftpd 의 이 인사 배너 파일을 참조하려면 다음 지시문을 **/etc/COMPLETE/COMPLETE.conf** 파일에 추가합니다.

```
banner_file=/etc/banners/ftp.msg
```

또한 4.4.1.1절. “TCP 래퍼 및 연결 배너” 에 설명된 대로 **TCP Wrappers**를 사용하여 들어오는 연결에 추가 배너를 보낼 수 있습니다.

4.3.9.2. 익명 액세스

/var/ftp/ 디렉터리가 있으면 익명 계정이 활성화됩니다.

이 디렉터리를 생성하는 가장 쉬운 방법은 **vsftpd** 패키지를 설치하는 것입니다. 이 패키지는 익명 사용자의 디렉터리 트리를 설정하고 익명 사용자의 경우 읽기 전용으로 디렉토리에 대한 권한을 구성합니다.

기본적으로 익명 사용자는 디렉터리에 쓸 수 없습니다.

주의

FTP 서버에 대한 익명 액세스를 활성화하면 중요한 데이터가 저장된 위치를 유의하십시오.

4.3.9.2.1. 익명 업로드

익명의 사용자가 파일을 업로드할 수 있도록 하려면 **/var/ftp/pub/** 에 쓰기 전용 디렉토리를 만드는 것이 좋습니다. 이렇게 하려면 **root**로 다음 명령을 입력합니다.

```
~]# mkdir /var/ftp/pub/upload
```

다음으로 익명 사용자가 디렉터리의 내용을 볼 수 없도록 권한을 변경합니다.

```
~]# chmod 730 /var/ftp/pub/upload
```

디렉터리의 긴 형식 목록은 다음과 같아야 합니다.

```
~]# ls -ld /var/ftp/pub/upload
drwx-wx---. 2 root ftp 4096 Nov 14 22:57 /var/ftp/pub/upload
```

익명의 사용자가 디렉토리에 읽고 쓸 수 있도록 허용하는 관리자는 서버가 가려진 소프트웨어의 저장소가 되는 경우가 많습니다.

또한, **vsftpd** 에서 다음 행을 **/etc/COMPLETE/vsftpd.conf** 파일에 추가하십시오.

```
anon_upload_enable=YES
```

4.3.9.3. 사용자 계정

FTP는 인증을 위해 안전하지 않은 네트워크를 통해 암호화되지 않은 사용자 이름과 암호를 전송하므로 사용자 계정에서 서버에 대한 시스템 사용자 액세스를 거부하는 것이 좋습니다.

vsftpd 의 모든 사용자 계정을 비활성화하려면 다음 지시문을 /etc/COMPLETE/vsftpd.conf에 추가하십시오.

```
local_enable=NO
```

4.3.9.3.1. 사용자 계정 제한

root 사용자 및 sudo 권한이 있는 사용자와 같은 특정 계정 또는 특정 계정 그룹에 대해 FTP 액세스를 비활성화하는 가장 쉬운 방법은 4.2.1절. “루트 액세스 허용” 에 설명된 대로 PAM 목록 파일을 사용하는 것입니다. vsftpd 의 PAM 구성 파일은 /etc/pam.d/COMPLETE 입니다.

각 서비스 내에서 사용자 계정을 직접 비활성화할 수도 있습니다.

vsftpd 에서 특정 사용자 계정을 비활성화하려면 사용자 이름을 /etc/COMPLETE/ftpusers에 추가합니다.

4.3.9.4. TCP Wrappers를 사용하여 제어 액세스

TCP Wrappers를 사용하여 4.4.1절. “TCP Wrappers 및 xinetd를 사용하여 서비스 보안” 에 설명된 대로 FTP 데몬에 대한 액세스를 제어합니다.

4.3.10. Postfix 보안

Postfix는 SMTP(Simple Mail Transfer Protocol)를 사용하여 다른 MTA와 이메일 클라이언트 또는 전달 에이전트에 전자 메시지를 전달하는 MTA(메일 전송 에이전트)입니다. 많은 MTA가 서로 간에 트래픽을 암호화할 수 있지만 대부분은 그렇지 않으므로 공용 네트워크를 통해 이메일을 전송하는 것은 본질적으로 안전하지 않은 통신 형식으로 간주됩니다. Postfix는 Red Hat Enterprise Linux 7에서 Sendmail을 기본 MTA로 대체합니다.

Postfix 서버를 구현하려는 모든 사람이 다음 문제를 해결하는 것이 좋습니다.

4.3.10.1. 서비스 거부 공격 제한

이메일의 특성으로 인해 확인 된 공격자는 서버에서 매우 쉽게 메일로 범람하고 서비스 거부를 일으킬 수 있습니다. 이러한 공격의 효율성은 /etc/postfix/main.cf 파일에서 지시문의 제한을 설정하여 제한할 수 있습니다. 이미 있는 지시문의 값을 변경하거나 필요한 지시문을 다음 형식으로 추가할 수 있습니다.

<directive> = <value>

. 다음은 서비스 거부 공격을 제한하는 데 사용할 수 있는 지시문 목록입니다.

- *smtpd_client_connection_rate_limit - 클라이언트가 시간 단위마다 이 서비스를 수행할 수 있는 최대 연결 시도 수입니다(아래 설명됨). 기본값은 0이며, 이는 Postfix가 허용할 수 있으므로 클라이언트에서 시간 단위당 많은 연결을 할 수 있음을 의미합니다. 기본적으로 신뢰할 수 있는 네트워크의 클라이언트는 제외됩니다.*

- *anvil_rate_time_unit - 이 시간 단위는 속도 제한 계산에 사용됩니다. 기본값은 60초입니다.*

- *smtpd_client_event_limit_exceptions - 연결 및 속도 제한 명령에서 제외된 클라이언트입니다. 기본적으로 신뢰할 수 있는 네트워크의 클라이언트는 제외됩니다.*

- *nameserverd_client_message_rate_limit - 클라이언트가 시간 단위당 요청할 수 있는 최대 메시지 전송 수입니다(Postgr이 실제로 해당 메시지를 수락하는지 여부와 관계없이).*

- *default_process_limit - 지정된 서비스를 제공하는 Postfix 하위 프로세스의 기본 최대 수입니다. 이 제한은 master.cf 파일에 있는 특정 서비스에 대해 일반화될 수 있습니다. 기본값은 100입니다.*

- *queue_minfree - 메일 수신에 필요한 대기열 파일 시스템에서 사용 가능한 최소 공간(바이트)입니다. 이는 현재 Postfix SMTP 서버에서 사용하여 모든 메일을 허용할지 여부를 결정하는 데 사용됩니다. 기본적으로 사용 가능한 공간 크기가 message_size_limit보다 1.5배 미만인 경우 Postfix SMTP 서버는 MAIL FROM 명령을 거부합니다. 더 높은 사용 가능한 공간 제한을 지정하려면 message_size_limit를 1.5배 이상 포함하는 queue_minfree 값을 지정하십시오. 기본적으로 queue_minfree 값은 0입니다.*

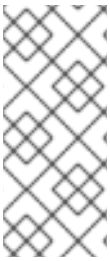
- *header_size_limit - 메시지 헤더를 저장하기 위한 바이트 단위의 최대 메모리 양입니다. 헤더가 클 경우 초과가 삭제됩니다. 기본값은 102400입니다.*

- *message_size_limit - 인코딩 정보를 포함하여 메시지의 최대 크기(바이트)입니다. 기본값은 10240000입니다.*

4.3.10.2. NFS 및 Postfix

NFS 공유 볼륨에 메일 스풀 디렉토리 /var/spool/postfix/ 을 넣지 마십시오. NFSv2와 NFSv3는 사용

자 및 그룹 ID에 대한 제어를 유지하지 않으므로 두 명 이상의 사용자는 동일한 UID를 가질 수 있으며 서로의 메일을 수신 및 읽을 수 있습니다.



참고

Kerberos를 사용하는 NFSv4에서는 SECRPC_GSS 커널 모듈이 UID 기반 인증을 사용하지 않으므로 이러한 문제가 발생하지 않습니다. 그러나 NFS 공유 볼륨에 메일 스풀 디렉터리를 배치 하지 않는 것이 좋습니다.

4.3.10.3. 메일 전용 사용자

Postfix 서버에서 로컬 사용자가 악용하는 것을 방지하기 위해 메일 사용자가 이메일 프로그램을 사용하여 Postfix 서버에만 액세스하는 것이 가장 좋습니다. 메일 서버의 쉘 계정은 허용되지 않아야 하며 /etc/passwd 파일의 모든 사용자 쉘은 /sbin/nologin 으로 설정해야 합니다(root 사용자를 제외하고).

4.3.10.4. Postfix 네트워크 수신 비활성화

기본적으로 Postfix는 로컬 루프백 주소만 수신 대기하도록 설정됩니다. /etc/postfix/main.cf 파일을 보고 이를 확인할 수 있습니다.

/etc/postfix/main.cf 파일을 보고 다음 inet_interfaces 행만 표시되는지 확인합니다.

```
inet_interfaces = localhost
```

이렇게 하면 Postfix가 네트워크가 아닌 로컬 시스템의 메일 메시지(예: cron 작업 보고서)만 수락하게 됩니다. 이는 기본 설정이며 Postfix가 네트워크 공격으로부터 보호합니다.

localhost 제한 제거 및 Postfix가 모든 인터페이스에서 수신 대기하도록 허용하려면 inet_interfaces = all 설정을 사용할 수 있습니다.

4.3.10.5. SASL을 사용하도록 Postfix 구성

Red Hat Enterprise Linux 7 버전의 Postfix 는 SMTP 인증(또는 SMTPAUTH)을 위해 Dovecot 또는 Cyrus SASL 구현을 사용할 수 있습니다. SMTP 인증은 단순 메일 전송 프로토콜 의 확장입니다. 활성화된 경우, 서버와 클라이언트 둘 다에서 지원 및 수락되는 인증 방법을 사용하여 SMTP 서버를 인증해야 합니다. 이 섹션에서는 Dovecot SASL 구현을 사용하도록 Postfix 를 구성하는 방법을 설명합니다.

Dovecot POP/IMAP 서버를 설치하면 **Dovecot SASL** 구현을 시스템에서 사용할 수 있게 하려면 **root** 사용자로 다음 명령을 실행합니다.

```
~]# yum install dovecot
```

Postfix SMTP 서버는 **UNIX** 도메인 소켓 또는 **TCP** 소켓 을 사용하여 **Dovecot SASL** 구현과 통신할 수 있습니다. 후자의 방법은 **Postfix** 및 **Dovecot** 애플리케이션이 별도의 시스템에서 실행되는 경우에만 필요합니다. 이 안내서는 **UNIX-domain socket** 방법에 우선하여 더 나은 개인 정보를 제공합니다.

Postfix 가 **Dovecot SASL** 구현을 사용하도록 지시하려면 두 애플리케이션 모두에 대해 여러 구성을 변경해야 합니다. 이러한 변경 사항을 적용하려면 아래 절차를 따르십시오.

Dovecot 설정

1. 다음 행을 포함하도록 기본 **Dovecot** 구성 파일인 **/etc/dovecot/conf.d/10-master.conf** 를 수정합니다(기본 설정 파일에는 이미 관련 섹션이 대부분 포함되어 있으며 행의 주석 처리를 해제해야 합니다).

```
service auth {
  unix_listener /var/spool/postfix/private/auth {
    mode = 0660
    user = postfix
    group = postfix
  }
}
```

위의 예에서는 **Postfix** 와 **Dovecot** 간 통신에 **UNIX** 도메인 소켓을 사용하는 것으로 가정합니다. 또한 **/var/spool/postfix/** 디렉터리에 있는 메일 대기열과 **postfix** 사용자 및 그룹 하에서 실행되는 애플리케이션을 포함하는 **Postfix SMTP** 서버의 기본 설정도 가정합니다. 이렇게 하면 읽기 및 쓰기 권한이 **postfix** 사용자 및 그룹으로 제한됩니다.

또는 다음 구성을 사용하여 **TCP** 를 통해 **Postfix** 인증 요청을 수신 대기하도록 **Dovecot** 를 설정할 수 있습니다.

```
service auth {
  inet_listener {
    port = 12345
  }
}
```

위의 예에서 **12345** 를 사용하려는 포트 수로 바꿉니다.

2.

/etc/dovecot/conf.d/10-auth.conf 구성 파일을 편집하여 Dovecot 에 일반 및 로그인 인증 메커니즘을 제공하도록 Dovecot에 지시합니다.

```
auth_mechanisms = plain login
```

Postfix 설정

Postfix 의 경우 기본 구성 파일인 /etc/postfix/main.cf 만 수정해야 합니다. 다음 설정 지시문을 추가하거나 편집합니다.

1.

Postfix SMTP 서버에서 SMTP 인증을 활성화합니다.

```
smtpd_sasl_auth_enable = yes
```

2.

Postfix 에 SMTP 인증을 위한 Dovecot SASL 구현을 사용하도록 지시합니다.

```
smtpd_sasl_type = dovecot
```

3.

Postfix 대기열 디렉토리와 관련된 인증 경로를 제공합니다(관련 경로를 사용하면 Postfix 서버가 chroot 에서 실행되는지 여부에 관계없이 구성이 작동합니다.).

```
smtpd_sasl_path = private/auth
```

이 단계에서는 Postfix 와 Dovecot 간 통신에 UNIX 도메인 소켓을 사용한다고 가정합니다. 통신에 TCP 소켓을 사용하는 경우 다른 시스템에서 Dovecot 를 찾도록 Postfix 를 구성하려면 다음과 유사한 구성 값을 사용합니다.

```
smtpd_sasl_path = inet:127.0.0.1:12345
```

위 예에서 127.0.0.1 은 Dovecot 머신의 IP 주소와 12345 를 Dovecot 의/etc/dovecot/conf.d/10-master.conf 설정 파일에 지정된 포트로 교체해야 합니다.

4.

Postfix SMTP 서버가 클라이언트에서 사용할 수 있도록 SASL 메커니즘을 지정합니다. 암호화되거나 암호화되지 않은 세션에 대해 다양한 메커니즘을 지정할 수 있습니다.

```
smtpd_sasl_security_options = noanonymous, noplaintext
smtpd_sasl_tls_security_options = noanonymous
```

위의 예제에서는 암호화되지 않은 세션 중에는 익명 인증이 허용되지 않으며 암호화되지 않은 사용자 이름 또는 암호를 전송하는 메커니즘이 허용되지 않도록 지정합니다. 암호화된 세션(TLS사용)의 경우 비익명 인증 메커니즘만 허용됩니다.

허용되는 SASL 메커니즘 제한을 위해 지원되는 모든 정책 목록은 http://www.postfix.org/SASL_README.html#smtpd_sasl_security_options 를 참조하십시오.

추가 리소스

다음 온라인 리소스는 SASL 을 통해 Postfix SMTP 인증을 구성하는 데 유용한 추가 정보를 제공합니다.

- http://wiki2.dovecot.org/HowTo/PostfixAndDovecotSASL - SMTP 인증에 대해 Dovecot SASL 구현을 사용하도록 Postfix 를 설정하는 방법에 대한 정보가 포함되어 있습니다.
- http://www.postfix.org/SASL_README.html#server_sasl - SMTP 인증을 위한 Dovecot 또는 Cyrus SASL 구현을 사용하도록 Postfix 를 설정하는 방법에 대한 정보가 포함되어 있습니다.

4.3.11. SSH 보안

SSH(Secure Shell)는 보안 채널을 통해 다른 시스템과 통신하는 데 사용되는 강력한 네트워크 프로토콜입니다. SSH 를 통한 전송은 암호화되고 가로채기로부터 보호됩니다. SSH 프로토콜에 대한 일반적인 정보와 Red Hat Enterprise Linux 7의 SSH 서비스 사용에 대한 자세한 내용은 Red Hat Enterprise Linux 7 시스템 관리자 가이드의 OpenSSH 장을 참조하십시오.

중요

이 섹션에서는 SSH 설정을 보호하는 가장 일반적인 방법에 대해 설명합니다. 이러한 제안된 조치 목록은 완전하거나 최종적인 것으로 간주해서는 안 됩니다. 기본 SSH 개념에 대한 설명은 sshd 데몬의 동작을 수정하는 데 사용할 수 있는 모든 구성 지시문에 대한 설명은 sshd_config (5) 를 참조하십시오.

4.3.11.1. 암호화 로그인

SSH 는 컴퓨터에 로그인하는 데 암호화 키를 사용할 수 있도록 지원합니다. 이는 암호만 사용하는 것보다 훨씬 안전합니다. 이 방법을 다른 인증 방법과 결합하면 다단계 인증으로 간주될 수 있습니다.If you combine this method with other authentication methods, it can be considered a multi-factor

authentication. 여러 인증 방법 사용에 대한 자세한 내용은 4.3.11.2절. "다중 인증 방법" 을 참조하십시오.

인증에 암호화 키를 사용하려면 **/etc/ssh/sshd_config** 파일의 **PubkeyAuthentication** 구성 지시문을 **yes** 로 설정해야 합니다. 이는 기본 설정입니다. 로그인에 암호를 사용할 수 없도록 **PasswordAuthentication** 지시문을 **no** 로 설정합니다.

ssh-keygen 명령을 사용하여 **SSH** 키를 생성할 수 있습니다. 추가 인수 없이 호출되는 경우 **2048**비트 **RSA** 키 세트를 생성합니다. 키는 기본적으로 **~/.ssh/** 디렉터리에 저장됩니다. **-b** 스위치를 사용하여 키의 **bit-strength**를 수정할 수 있습니다. **2048**비트 키를 사용하면 일반적으로 충분합니다. **Red Hat Enterprise Linux 7** 시스템 관리자 가이드의 **OpenSSH** 구성 장에는 키 쌍 생성에 대한 자세한 정보가 포함되어 있습니다.

~/.ssh/ 디렉터리에 두 개의 키가 표시되어야 합니다. **ssh-keygen** 명령을 실행할 때 기본값을 수락한 경우 생성된 파일의 이름은 **id_rsa** 및 **id_rsa.pub** 이고 개인 및 공개 키를 각각 포함합니다. 항상 개인 키를 노출에서 보호하여 다른 사람이 읽을 수 없도록 해야 하지만 파일의 소유자가 읽을 수 없도록 해야 합니다. 그러나 공개 키는 로그인할 시스템으로 전송되어야 합니다. **ssh-copy-id** 명령을 사용하여 서버에 키를 전송할 수 있습니다.

```
~]$ ssh-copy-id -i [user@]server
```

또한 이 명령은 공개 키를 서버의 **~/.ssh/authorized_keys** 파일에 자동으로 추가합니다. **sshd** 데몬은 서버에 로그인하려고 할 때 이 파일을 확인합니다.

암호 및 기타 인증 메커니즘과 마찬가지로 **SSH** 키를 정기적으로 변경해야 합니다. 이 작업을 수행하면 **authorized_keys** 파일에서 사용되지 않은 키를 제거해야 합니다.

4.3.11.2. 다중 인증 방법

다중 인증 방법 또는 다단계 인증을 사용하면 무단 액세스로부터 보호 수준이 높아지므로 시스템이 손상되지 않도록 강화할 때 고려해야 합니다. 다중 요소 인증을 사용하는 시스템에 로그인하려고 하는 사용자는 액세스 권한을 부여하기 위해 지정된 모든 인증 방법을 성공적으로 완료해야 합니다.

/etc/ssh/sshd_config 파일에서 **AuthenticationMethods configuration** 지시문을 사용하여 사용할 인증 방법을 지정합니다. 이 지시문을 사용하여 두 개 이상의 필수 인증 방법 목록을 정의할 수 있습니다. 이 경우 사용자는 목록 중 하나 이상에서 모든 메서드를 완료해야 합니다. 목록을 빈 공백으로 구분해야 하며 목록 내의 개별 인증 방법 이름은 쉼표로 구분해야 합니다. 예를 들어 다음과 같습니다.

```
AuthenticationMethods publickey,gssapi-with-mic publickey,keyboard-interactive
```

위의 **AuthenticationMethods** 지시문을 사용하여 구성된 **sshd** 데몬은 사용자가 공개 키 인증과 **gssapi-with-mic** 또는 키보드-대화형 인증을 통해 성공적으로 로그인을 시도하는 경우에만 액세스 권한을 부여합니다. **/etc/ssh/sshd_config** 파일에서 해당 구성 지시문(예: **PubkeyAuthentication**)을 사용하여 요청된 각 인증 방법을 명시적으로 활성화해야 합니다. 사용 가능한 인증 방법의 일반적인 목록은 **ssh(1)** 의 **AUTHENTICATION** 섹션을 참조하십시오.

4.3.11.3. SSH 보안의 다른 방법

프로토콜 버전

Red Hat Enterprise Linux 7과 함께 제공되는 **SSH** 프로토콜의 구현은 **SSH** 클라이언트에 대해 프로토콜의 **SSH-1** 및 **SSH-2** 버전을 계속 지원하지만 가능한 경우 후자만 사용해야 합니다. **SSH-2** 버전에는 이전 **SSH-1**보다 많은 개선 사항이 포함되어 있으며 대부분의 고급 구성 옵션은 **SSH-2**를 사용하는 경우에만 사용할 수 있습니다.

SSH -2를 사용하여 **SSH** 프로토콜이 사용되는 인증 및 통신을 보호하는 범위를 최대화하는 것이 좋습니다. **sshd** 데몬에서 지원하는 프로토콜 버전 또는 버전은 **/etc/ssh/sshd_config** 파일의 프로토콜 구성 지시문을 사용하여 지정할 수 있습니다. 기본 설정은 **2** 입니다. **SSH-2** 버전은 **Red Hat Enterprise Linux 7 SSH** 서버에서 지원하는 유일한 버전입니다.

키 유형

ssh-keygen 명령은 기본적으로 **SSH-2 RSA** 키 쌍을 생성하는 반면, **-t** 옵션을 사용하여 **DSA** 또는 **ECDSA** 키도 생성하도록 지시할 수 있습니다. **ECDSA (Elliptic Curve Digital Signature Algorithm)**는 동일한 대칭 키 길이에서 더 나은 성능을 제공합니다. 또한 더 짧은 키를 생성합니다.

기본이 아닌 포트

기본적으로 **sshd** 데몬은 **TCP** 포트 **22** 에서 수신 대기합니다. 포트를 변경하면 자동화된 네트워크 검사를 기반으로 하는 공격에 시스템의 노출이 줄어들어 모호성을 통한 보안이 향상됩니다. 포트는 **/etc/ssh/sshd_config** 구성 파일에서 **Port** 지시문을 사용하여 지정할 수 있습니다. 기본이 아닌 포트 사용을 허용하도록 기본 **SELinux** 정책을 변경해야 합니다. **root** 로 다음 명령을 입력하여 **ssh_port_t SELinux** 유형을 수정할 수 있습니다.

```
~]# semanage -a -t ssh_port_t -p tcp port_number
```

위의 명령에서 **port_number** 를 **Port** 지시문을 사용하여 지정된 새 포트 번호로 바꿉니다.

루트 로그인 없음

특정 사용 사례에 **root** 사용자로 로그인할 가능성이 필요하지 않은 경우 **PermitRootLogin** 설정 지시문을 **/etc/ssh/sshd_config** 파일에서 **no** 로 설정하는 것이 좋습니다. 관리자는 **root** 사용자로 로그인할 가능성을 비활성화하여 일반 사용자로 로그인한 후 권한 있는 명령을 실행한 다음 루트 권한을 얻을 수 있는 사용자를 감사할 수 있습니다.

X Security 확장 사용

Red Hat Enterprise Linux 7 클라이언트의 **X** 서버는 **X** 보안 확장을 제공하지 않습니다. 따라서 클라이언트는 **X11** 전달을 통해 신뢰할 수 없는 **SSH** 서버에 연결할 때 다른 보안 계층을 요청할 수 없습니다. 대부분의 애플리케이션은 이 확장 기능이 활성화된 상태에서 실행할 수 없었습니다. 기본적으로 **/etc/ssh/ssh_config** 파일의 **ForwardX11Trusted** 옵션은 **yes** 로 설정되어 있으며 **ssh -X remote_machine** (신뢰할 수 없는 호스트)과 **ssh -Y remote_machine (trusted host)** 명령 사이에 차이가 없습니다.

주의

신뢰할 수 없는 호스트에 연결하는 동안 **X11** 전달을 사용하지 않는 것이 좋습니다.

4.3.12. PostgreSQL 보안

PostgreSQL 은 **DBMS(Object-Relational database Management System)**입니다. **Red Hat Enterprise Linux 7**에서 **postgresql-server** 패키지는 **PostgreSQL** 을 제공합니다. 설치되지 않은 경우 **root** 사용자로 다음 명령을 입력하여 설치합니다.

```
~]# yum install postgresql-server
```

PostgreSQL 을 사용하려면 먼저 디스크에서 데이터베이스 스토리지 영역을 초기화해야 합니다. 이를 데이터베이스 클러스터라고 합니다. 데이터베이스 클러스터를 초기화하려면 **PostgreSQL** 과 함께 설치된 **initdb** 명령을 사용합니다. 데이터베이스 클러스터의 원하는 파일 시스템 위치는 **-D** 옵션으로 표시됩니다. 예를 들어 다음과 같습니다.

```
~]$ initdb -D /home/postgresql/db1
```

initdb 명령이 아직 없는 경우 지정한 디렉터리를 생성하려고 합니다. 이 예에서는 **/home/postgresql/db1** 이라는 이름을 사용합니다. **/home/postgresql/db1** 디렉터리에는 데이터베이스에 저장된 모든 데이터와 클라이언트 인증 구성 파일이 포함되어 있습니다.

```
~]$ cat pg_hba.conf
# PostgreSQL Client Authentication Configuration File
# This file controls: which hosts are allowed to connect, how clients
# are authenticated, which PostgreSQL user names they can use, which
# databases they can access.  Records take one of these forms:
#
# local      DATABASE  USER  METHOD  [OPTIONS]
```

```
# host      DATABASE  USER  ADDRESS  METHOD  [OPTIONS]
# hostssl   DATABASE  USER  ADDRESS  METHOD  [OPTIONS]
# hostnossl  DATABASE  USER  ADDRESS  METHOD  [OPTIONS]
```

pg_hba.conf 파일의 다음 줄을 사용하면 인증된 모든 로컬 사용자가 사용자 이름이 있는 데이터베이스에 액세스할 수 있습니다.

```
local   all             all                                     trust
```

이 문제는 데이터베이스 사용자를 생성하고 로컬 사용자가 없는 계층화된 애플리케이션을 사용할 때 문제가 될 수 있습니다. 시스템의 모든 사용자 이름을 명시적으로 제어하지 않으려면 **pg_hba.conf** 파일에서 이 행을 제거합니다.

4.3.13. Docker 보안

Docker 는 **Linux Containers** 내에서 애플리케이션 배포를 자동화하고 런타임 종속 항목과 함께 애플리케이션을 컨테이너로 패키징하는 기능을 제공하는 오픈 소스 프로젝트입니다. **Docker** 워크플로를 보다 안전하게 유지하려면 **Red Hat Enterprise Linux Atomic Host 7** 컨테이너 보안 가이드 의 절차를 따르십시오.

4.3.14. DDoS 공격에 대해 memcached 보안

Memcached 는 오픈 소스 고성능 분산 메모리 개체 캐싱 시스템입니다. 대부분의 경우 데이터베이스 로드를 줄여 동적 웹 애플리케이션의 성능을 개선하는 데 주로 사용됩니다.

Memcached는 데이터베이스 호출, **API** 호출 또는 페이지 렌더링 결과에서 문자열 및 오브젝트와 같은 임의의 데이터의 작은 청크를 위한 메모리 내 키-값 저장소입니다. **Memcached**를 사용하면 애플리케이션이 필요한 것보다 더 많은 시스템이 있는 시스템의 일부에서 메모리를 가져와 애플리케이션이 필요한 것보다 적은 영역에서 액세스할 수 있도록 합니다.

Memcached Vulnerabilities

2018년, 공개 인터넷에 노출된 **memcached** 서버를 악용하여 **DDoS** 개정 공격 취약점을 발견했습니다. 이러한 공격은 전송에 **UDP** 프로토콜을 사용하여 **memcached** 통신을 활용합니다. 이 공격은 높은 진폭 비율로 인해 효과가 있습니다 - 몇 백 바이트 크기의 요청이 몇 메가바이트 또는 수백 메가바이트 크기의 응답을 생성할 수 있습니다. 이 문제는 **CVE-2018-1000115** 로 지정되어 있습니다.

대부분의 경우 **memcached** 서비스를 공용 인터넷에 노출할 필요가 없습니다. 이러한 노출에는 원격 공격자가 **memcached**에 저장된 정보를 유출하거나 수정할 수 있는 자체 보안 문제가 있을 수 있습니다.

memcached 강화

보안 위험을 완화하려면 구성에 적용 가능한 다음 단계의 수를 수행합니다.

- LAN에서 방화벽을 구성합니다. 로컬 네트워크 내에서만 **memcached** 서버에 액세스할 수 있어야 하는 경우 **memcached**에서 사용하는 포트에 대한 외부 트래픽을 허용하지 마십시오. 예를 들어 **memcached**에서 기본적으로 사용되는 포트 **11211**을 허용된 포트 목록에서 제거합니다.

```
~]# firewall-cmd --remove-port=11211/udp
~]# firewall-cmd --runtime-to-permanent
```

  특정 IP 범위가 포트 **11211**을 사용하도록 허용하는 **firewalld** 명령은 5.8절. “Zone을 사용하여 소스에 따라 트래픽 관리” 를 참조하십시오.

- 클라이언트에 이 프로토콜이 필요하지 않는 한 **/etc/sysconfig/memcached** 파일의 **OPTIONS** 변수에 **-U 0 -p 11211** 값을 추가하여 **UDP**를 비활성화합니다.

```
OPTIONS="-U 0 -p 11211"
```

- 애플리케이션과 동일한 시스템에서 단일 **memcached** 서버만 사용하는 경우 **memcached**를 설정하여 **localhost** 트래픽만 수신 대기합니다. **-l 127.0.0.1,::1** 값을 **/etc/sysconfig/memcached** 의 **OPTIONS** 에 추가합니다.

```
OPTIONS="-l 127.0.0.1,::1"
```

- 인증을 변경할 수 있는 경우 **SASL(Simple Authentication and Security Layer)** 인증을 활성화합니다.

1. **/etc/sasl2/memcached.conf** 파일에 수정하거나 추가합니다.

```
sasldb_path: /path.to/memcached.sasldb
```

2. **SASL** 데이터베이스에 계정을 추가합니다.

```
~]# saslpasswd2 -a memcached -c cacheuser -f /path.to/memcached.sasldb
```

3. **memcached** 사용자 및 그룹에 대한 데이터베이스에 액세스할 수 있는지 확인합니다.

```
~]# chown memcached:memcached /path.to/memcached.sasldb
```

4. **OPTIONS** 에 **-S** 값을 **/etc/sysconfig/memcached** 에 추가하여 **memcached**에서 **SASL** 지원을 활성화합니다.

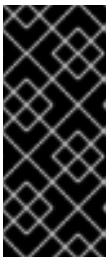
```
OPTIONS="-S"
```

5. **memcached** 서버를 다시 시작하여 변경 사항을 적용합니다.

6. **SASL** 데이터베이스에서 생성된 사용자 이름과 암호를 애플리케이션의 **memcached** 클라이언트 구성에 추가합니다.

- **stunnel** 을 사용하여 **memcached** 클라이언트와 서버 간 통신을 암호화합니다. **memcached**에서 **TLS**를 지원하지 않으므로 해결방법은 **memcached** 프로토콜 상단에 **TLS**를 제공하는 **stunnel** 과 같은 프록시를 사용하는 것입니다.

  PSK (Pre Shared Keys)를 사용하도록 **stunnel** 을 구성하거나 사용자 인증서를 사용하는 것이 더 좋습니다. 인증서를 사용하는 경우 인증된 사용자만 **memcached** 서버에 연결할 수 있으며 트래픽이 암호화됩니다.



중요

터널을 사용하여 **memcached**에 액세스하는 경우 서비스가 **localhost**에서만 수신 대기 중이거나 방화벽이 네트워크에서 **memcached** 포트에 대한 액세스를 허용하지 않는지 확인합니다.

자세한 내용은 4.8절. “stunnel 사용”를 참조하십시오.

4.4. 네트워크 액세스 보안

4.4.1. TCP Wrappers 및 xinetd를 사용하여 서비스 보안

TCP Wrappers는 서비스에 대한 액세스를 거부하는 것 이상으로 많은 것을 할 수 있습니다. 이 섹션에서는 연결 배너를 전송하고 특정 호스트의 공격을 경고하고 로깅 기능을 향상시키는 데 사용할 수 있는 방법을 보여줍니다. **TCP Wrapper** 기능 및 제어 언어에 대한 자세한 내용은 **hosts_options(5)** 도움말 페이지를 참조하십시오. 서비스에 적용할 수 있는 옵션 역할을 하는 사용 가능한 플래그의 **xinetd.conf(5)** 도움말 페이지를 참조하십시오.

4.4.1.1. TCP 래퍼 및 연결 배너

사용자가 서비스에 연결할 때 적합한 배너를 표시하는 것은 잠재적인 공격자가 시스템 관리자가 감시하고 있음을 알리는 좋은 방법입니다. 또한 사용자에게 제공되는 시스템에 대한 정보를 제어할 수 있습니다. 서비스에 대한 **TCP Wrappers** 배너를 구현하려면 **banner** 옵션을 사용합니다.

이 예제에서는 **vsftpd** 에 대한 배너를 구현합니다. 시작하려면 배너 파일을 만듭니다. 시스템의 모든 위치에 있을 수 있지만 데몬과 동일한 이름이 있어야 합니다. 이 예제에서는 파일을 **/etc/banners/COMPLETE**라고 하며 다음 행을 포함합니다.

```
220-Hello, %c
220-All activity on ftp.example.com is logged.
220-Inappropriate use will result in your access privileges being removed.
```

%c 토큰은 사용자 이름 및 호스트 이름과 같은 다양한 클라이언트 정보 또는 사용자 이름과 **IP** 주소를 제공하여 연결을 훨씬 더 위협적으로 만듭니다.

이 배너를 들어오는 연결에 표시하려면 **/etc/hosts.allow** 파일에 다음 행을 추가합니다.

```
vsftpd : ALL : banners /etc/banners/
```

4.4.1.2. TCP Wrappers 및 공격 경고

특정 호스트 또는 네트워크가 서버 공격으로 탐지된 경우 **TCP Wrappers**를 사용하여 **generate** 지시문을 사용하여 해당 호스트 또는 네트워크의 후속 공격 관리자에게 경고를 표시할 수 있습니다.

이 예에서 **206.182.68.0/24** 네트워크의 크래커가 서버 공격을 시도하는 것으로 탐지되었습니다. **/etc/hosts.deny** 파일에 다음 행을 추가하여 해당 네트워크의 연결 시도를 거부하고 특수 파일에 대한 시도를 기록하십시오.

```
ALL : 206.182.68.0 : spawn /bin/echo `date` %c %d >> /var/log/intruder_alert
```

%d 토큰은 공격자가 액세스하려고 하는 서비스의 이름을 제공합니다.

연결을 허용하고 로깅하려면 **/etc/hosts.allow** 파일에 **spawn** 지시문을 배치합니다.

참고

spawn 지시문은 쉘 명령을 실행하기 때문에 특수 스크립트를 생성하여 관리자에게 알리거나 특정 클라이언트가 서버에 연결을 시도하는 경우 명령 체인을 실행하는 것이 좋습니다.

4.4.1.3. TCP Wrappers 및 향상된 로깅

특정 유형의 연결이 다른 연결보다 우려되는 경우 심각도 옵션을 사용하여 해당 서비스의 로그 수준을 높일 수 있습니다.

이 예에서는 **FTP** 서버의 포트 **23(Telnet** 포트)에 연결을 시도하는 모든 사람이 크래커라고 가정합니다. 이를 표시하려면 기본 플래그, 정보, 연결 거부 대신 로그 파일에 **emerg** 플래그를 배치합니다.

이렇게 하려면 **/etc/hosts.deny** 에 다음 행을 배치합니다.

```
in.telnetd : ALL : severity emerg
```

이는 기본 **authpriv** 로깅 기능을 사용하지만 기본 값에서 **info** 의 기본 값에서 **emerg** 로 우선 순위를 승격합니다. 이 기능은 콘솔에 로그 메시지를 직접 게시합니다.

4.4.2. Which Ports Are Listening 확인

가능한 공격을 피하기 위해 사용되지 않는 포트를 닫는 것이 중요합니다. 수신 대기 상태의 예기치 않은 포트의 경우 발생할 수 있는 침입에 대해 조사해야 합니다.

Open Ports Scan에 netstat 사용

*다음 명령을 **root** 로 입력하여 네트워크에서 연결을 수신 대기 중인 포트를 확인합니다.*

```
~]# netstat -pan -A inet,inet6 | grep -v ESTABLISHED
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address       Foreign Address    State     PID/Program name
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address          Foreign Address        State       PID/Program name
tcp        0      0 0.0.0.0:111            0.0.0.0:*              LISTEN      1/systemd
tcp        0      0 192.168.124.1:53       0.0.0.0:*              LISTEN      1829/dnsmasq
tcp        0      0 0.0.0.0:22             0.0.0.0:*              LISTEN      1176/sshd
tcp        0      0 127.0.0.1:631          0.0.0.0:*              LISTEN      1177/cupsd
tcp6       0      0 :::111                 :::*                   LISTEN      1/systemd
tcp6       0      0 ::1:25                 :::*                   LISTEN      1664/master
sctp               0.0.0.0:2500                                    LISTEN   20985/sctp_darn
udp        0      0 192.168.124.1:53       0.0.0.0:*                       1829/dnsmasq
udp        0      0 0.0.0.0:67             0.0.0.0:*                       977/dhclient
...
```

***netstat** 명령의 **-l** 옵션을 사용하여 수신 대기 중인 서버 소켓만 표시합니다.*

```
~]# netstat -tlnw
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp        0      0 0.0.0.0:111            0.0.0.0:*              LISTEN
tcp        0      0 192.168.124.1:53       0.0.0.0:*              LISTEN
tcp        0      0 0.0.0.0:22             0.0.0.0:*              LISTEN
tcp        0      0 127.0.0.1:631          0.0.0.0:*              LISTEN
tcp        0      0 127.0.0.1:25           0.0.0.0:*              LISTEN
tcp6       0      0 :::111                 :::*                   LISTEN
tcp6       0      0 :::22                  :::*                   LISTEN
tcp6       0      0 ::1:631                :::*                   LISTEN
tcp6       0      0 ::1:25                 :::*                   LISTEN
raw6       0      0 :::58                  :::*                   7
```

Open Ports Scan에 ss 사용

*또는 **ss** 유틸리티를 사용하여 수신 대기 상태에 열려 있는 포트를 나열합니다. **netstat** 보다 많은 **TCP** 및 상태 정보를 표시할 수 있습니다.*

```
~]# ss -tlw
etid State      Recv-Q Send-Q     Local Address:Port                 Peer Address:Port
udp   UNCONN     0      0                 :::ipv6-icmp                       :::*
tcp   LISTEN     0      128               *:sunrpc                          *:*
tcp   LISTEN     0      5         192.168.124.1:domain                          *:*
tcp   LISTEN     0      128               *:ssh                             *:*
tcp   LISTEN     0      128         127.0.0.1:ipp                             *:*
tcp   LISTEN     0      100         127.0.0.1:smtp                            *:*
tcp   LISTEN     0      128               :::sunrpc                         :::*
```

```
tcp   LISTEN     0      128                :::ssh                     :::*
tcp   LISTEN     0      128               ::1:ipp                     :::*
tcp   LISTEN     0      100               ::1:smtp                     :::*
```

```
~]# ss -plno -A tcp,udp,sctp
Netid State      Recv-Q Send-Q     Local Address:Port                 Peer Address:Port
udp   UNCONN     0      0          192.168.124.1:53                        *:*                 users:
(("dnsmasq",pid=1829,fd=5))
udp   UNCONN     0      0              *%virbr0:67                         *:*                 users:
(("dnsmasq",pid=1829,fd=3))
udp   UNCONN     0      0                   *:68                       *:*                 users:
(("dhclient",pid=977,fd=6))
...
tcp   LISTEN     0      5          192.168.124.1:53                        *:*                 users:
(("dnsmasq",pid=1829,fd=6))
tcp   LISTEN     0      128                 *:22                       *:*                 users:
(("sshd",pid=1176,fd=3))
tcp   LISTEN     0      128          127.0.0.1:631                         *:*                 users:
(("cupsd",pid=1177,fd=12))
tcp   LISTEN     0      100          127.0.0.1:25                          *:*                 users:
(("master",pid=1664,fd=13))
...
sctp  LISTEN     0      5                   *:2500                     *:*                 users:
(("sctp_darn",pid=20985,fd=3))
```

***UNCONN** 상태는 **UDP** 수신 대기 모드로 포트를 표시합니다.*

*외부 시스템에서 **ss** 출력에 표시된 모든 **IP** 주소(**localhost 127.0.0.0** 또는 **::1** 범위를 제외)에 대해 검사를 수행합니다. **IPv6** 주소를 스캔하는 데 **-6** 옵션을 사용합니다.*

*그런 다음 네트워크를 통해 연결된 다른 원격 시스템의 **nmap** 도구를 첫 번째 시스템으로 사용하여 외부 검사를 수행합니다. **firewalld** 에서 규칙을 확인하는 데 사용할 수 있습니다. 다음은 **TCP** 연결을 수신 대기 중인 포트를 결정하는 예입니다.*

```
~]# nmap -sT -O 192.168.122.65
   Starting Nmap 6.40 ( http://nmap.org ) at 2017-03-27 09:30 CEST
   Nmap scan report for 192.168.122.65
   Host is up (0.00032s latency).
   Not shown: 998 closed ports
   PORT    STATE SERVICE
   22/tcp  open  ssh
   111/tcp open  rpcbind
   Device type: general purpose
   Running: Linux 3.X
   OS CPE: cpe:/o:linux:linux_kernel:3
   OS details: Linux 3.7 - 3.9
   Network Distance: 0 hops
```

```
OS detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.79 seconds
```

TCP 연결 검사 **(-sT)** 는 **TCP SYN** 검사 **(-sS)** 가 옵션이 아닌 경우 기본 **TCP** 검사 유형입니다. **O** 옵션은 호스트의 운영 체제를 감지합니다.

netstat 및 **ss** 를 사용하여 **Open SCTP** 포트 검사

netstat 유틸리티는 **Linux** 네트워킹 하위 시스템에 대한 정보를 출력합니다. **Open Stream Control Transmission Protocol (SCTP)** 포트에 대한 프로토콜 통계를 표시하려면 **root** 로 다음 명령을 입력합니다.

```
~]# netstat -plnS
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address   Foreign Address  State    PID/Program name
sctp              127.0.0.1:250                 LISTEN   4125/sctp_darn
sctp       0      0 127.0.0.1:260   127.0.0.1:250    CLOSE    4250/sctp_darn
sctp       0      0 127.0.0.1:250   127.0.0.1:260    LISTEN   4125/sctp_darn
```

```
~]# netstat -nl -A inet,inet6 | grep 2500
sctp              0.0.0.0:2500                          LISTEN
```

ss 유틸리티는 **SCTP** 오픈 포트를 표시할 수도 있습니다.

```
~]# ss -an | grep 2500
sctp   LISTEN     0      5         *:2500                *:*
```

자세한 내용은 **ss(8), netstat(8), nmap(1)** 및 **services(5)** 매뉴얼 페이지를 참조하십시오.

4.4.3. 소스 라우팅 비활성화

소스 라우팅은 **IP** 패킷이 패킷에 취해야 하는 경로를 라우터에 알리는 정보, 주소 목록을 전송할 수 있도록 하는 인터넷 프로토콜 메커니즘입니다. 경로가 트래버스될 때 홉을 기록하는 옵션도 있습니다. **"route** 레코드**"**인 홉 목록은 대상에 소스에 대한 반환 경로를 제공합니다. 이를 통해 소스(전송 호스트)에서 일부 또는 모든 라우터의 라우팅 테이블을 무시하고 경로를 느슨하게 또는 엄격하게 지정할 수 있습니다. 사용자가 악의적인 목적으로 네트워크 트래픽을 리디렉션할 수 있습니다. 따라서 소스 기반 라우팅을 비활성화해야 합니다.

accept_source_route 옵션을 사용하면 네트워크 인터페이스가 **Strict Source Routing (SSR)** 또는 **Loose Source Routing (LSR)** 옵션이 설정된 패킷을 허용합니다. 소스 라우팅 패킷의 수락은 **sysctl** 설정에 의해 제어됩니다. 다음 명령을 **root**로 실행하여 **SSR** 또는 **LSR** 옵션이 설정된 패킷을 삭제합니다.

```
~]# /sbin/sysctl -w net.ipv4.conf.all.accept_source_route=0
```

*패킷 전달을 비활성화하는 것도 가능한 경우 위의 항목과 함께 수행해야 합니다(가상화를 방해할 수 있음). **root**로 아래 명령을 실행합니다.*

*이러한 명령은 모든 인터페이스에서 **IPv4** 및 **IPv6** 패킷 전달을 비활성화합니다.*

```
~]# /sbin/sysctl -w net.ipv4.conf.all.forwarding=0
```

```
~]# /sbin/sysctl -w net.ipv6.conf.all.forwarding=0
```

이러한 명령은 모든 인터페이스에서 모든 멀티 캐스트 패킷의 전달을 비활성화합니다.

```
~]# /sbin/sysctl -w net.ipv4.conf.all.mc_forwarding=0
```

```
~]# /sbin/sysctl -w net.ipv6.conf.all.mc_forwarding=0
```

__ICMP__ 리디렉션 수락에는 합법적인 사용이 거의 없습니다. 특별히 필요하지 않은 경우 __ICMP__ 리디렉션 패킷의 수락 및 전송을 비활성화합니다.

이러한 명령은 모든 인터페이스에서 모든 __ICMP__ 리디렉션 패킷을 수락하지 않도록 비활성화합니다.

```
~]# /sbin/sysctl -w net.ipv4.conf.all.accept_redirects=0
```

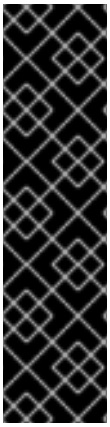
```
~]# /sbin/sysctl -w net.ipv6.conf.all.accept_redirects=0
```

이 명령은 모든 인터페이스에서 보안 __ICMP__ 리디렉션 패킷 수락을 비활성화합니다.

```
~]# /sbin/sysctl -w net.ipv4.conf.all.secure_redirects=0
```

이 명령은 모든 인터페이스에서 모든 __IPv4 ICMP__ 리디렉션 패킷의 수락을 비활성화합니다.

```
~]# /sbin/sysctl -w net.ipv4.conf.all.send_redirects=0
```

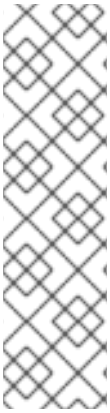


중요

ICMP 리디렉션 전송은 net.ipv4.conf.all.send_redirects 또는 net.ipv4.conf 중 하나 이상이 활성 상태로 유지됩니다.인터페이스.send_redirects 옵션이 enabled로 설정됩니다. net.ipv4.conf.인터페이스.send_redirects 옵션을 모든 인터페이스 의 0 값으로 설정해야 합니다. 새 인터페이스를 추가할 때마다 ICMP 요청 전송을 자동으로 비활성화하려면 다음 명령을 입력합니다.

```
~]# /sbin/sysctl -w net.ipv4.conf.default.send_redirects=0
```

IPv4 리디렉션 패킷 전송을 비활성화하는 지시문만 있습니다. IPv4와 IPv6의 차이로 인해 발생하는 "IPv6 노드 요구" 사항에 대한 설명은 RFC4294 를 참조하십시오.



참고

재부팅 후에도 이러한 설정을 지속하려면 /etc/sysctl.conf 파일을 수정하십시오. 예를 들어 모든 인터페이스에서 모든 IPv4 ICMP 리디렉션 패킷을 수락하려면 root 사용자로 실행되는 편집기로 /etc/sysctl.conf 파일을 열고 다음과 같이 행을 추가합니다.

```
net.ipv4.conf.all.send_redirects=0
```

자세한 내용은 sysctl 도움말 페이지 sysctl(8) 를 참조하십시오. 소스 기반 라우팅 및 해당 변형과 관련된 인터넷 옵션에 대한 설명은 RFC791 을 참조하십시오.

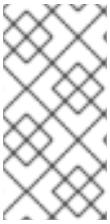
주의

이더넷 네트워크는 ARP 또는 MAC 주소 스푸핑, 무단 DHCP 서버, IPv6 라우터 또는 주변 광고와 같은 트래픽을 리디렉션하는 추가 방법을 제공합니다. 또한 유니캐스트 트래픽은 종종 브로드캐스트를 통해 정보 유출이 발생합니다. 이러한 약점은 네트워크 운영자에 의해 구현되는 특정 카운터에서만 해결할 수 있습니다. 호스트 기반 카운터는 완전히 작동하지 않습니다.

4.4.3.1. 역방향 경로 전달

역방향 경로 전달은 하나의 인터페이스를 통해 들어오는 패킷이 다른 인터페이스를 통해 나가는 것을 방지하는 데 사용됩니다. 나가는 경로와 들어오는 경로가 다른 경우 종종 symmetric 라우팅 이라고 합니

다. 라우터는 종종 이러한 방식으로 패킷을 라우팅하지만 대부분의 호스트는 이 작업을 수행할 필요가 없습니다. 예외는 하나의 링크를 통해 트래픽을 보내고 다른 서비스 공급자의 다른 링크를 통해 트래픽을 수신하는 것을 포함하는 애플리케이션입니다. 예를 들어, **X DSL** 또는 위성 링크와 **3G** 모뎀과 함께 리스된 라인 사용. 이러한 시나리오가 사용자에게 적용 가능한 경우 들어오는 인터페이스에서 역방향 경로 전달을 해제해야 합니다. 즉, 필요하지 않은 경우 사용자가 로컬 서브넷에서 **IP** 주소를 스푸핑하고 **DDoS** 공격의 가능성을 줄일 수 있으므로 이 기능이 가장 적합합니다.



참고

Red Hat Enterprise Linux 7의 기본값은 **RFC 3704**의 엄격한 **Reverse Path** 권장 사항에 따라 **Strict Reverse Path Forwarding** 을 사용합니다.

주의

전달이 활성화된 경우 **source-address** 검증 방법(예: **iptables** 규칙 등)이 있는 경우에만 **Reverse Path Forwarding**을 비활성화해야 합니다.

rp_filter

역방향 경로 전달은 **rp_filter** 지시문을 통해 활성화됩니다. **sysctl** 유틸리티를 사용하여 실행 중인 시스템을 변경할 수 있으며 **/etc/sysctl.conf** 파일에 행을 추가하여 영구적으로 변경할 수 있습니다. **rp_filter** 옵션은 커널에 세 가지 모드 중 하나를 선택하도록 지시하는 데 사용됩니다.

임시 전역 변경을 수행하려면 **root** 로 다음 명령을 입력합니다.

```
sysctl -w  net.ipv4.conf.default.rp_filter=integer
sysctl -w net.ipv4.conf.all.rp_filter=integer
```

여기서 **integer** 는 다음 중 하나입니다.

- **0 -** 소스 검증 없음
- **1 - RFC 3704**에 정의된 엄격한 모드.

- 2 - RFC 3704에 정의된 느슨한 모드입니다.

설정은 net.ipv4.conf를 사용하여 네트워크 인터페이스별로 재정의할 수 있습니다.인터페이스.rp_filter 명령은 다음과 같습니다.

```
sysctl -w net.ipv4.conf.interface.rp_filter=integer
```

참고

재부팅 후에도 이러한 설정을 지속하려면 /etc/sysctl.conf 파일을 수정하십시오. 예를 들어 모든 인터페이스의 모드를 변경하려면 root 사용자로 실행되는 편집기로 /etc/sysctl.conf 파일을 열고 다음과 같이 행을 추가합니다.

```
net.ipv4.conf.all.rp_filter=2
```

IPv6_rpfilter

IPv6 프로토콜의 경우 firewalld 데몬은 기본적으로 Reverse Path Forwarding에 적용됩니다. 설정은 /etc/firewalld/firewalld.conf 파일에서 확인할 수 있습니다. IPv6_rpfilter 옵션을 설정하여 firewalld 동작을 변경할 수 있습니다.

Reverse Path Forwarding의 사용자 지정 구성이 필요한 경우 다음과 같이 ip6tables 명령을 사용하여 firewalld 데몬 없이 이를 수행할 수 있습니다.

```
ip6tables -t raw -I PREROUTING -m rpfilter --invert -j DROP
```

이 규칙은 원시/PREROUTING 체인의 시작 부분에 삽입해야 하므로 특히 상태 저장 일치 규칙보다 먼저 모든 트래픽에 적용됩니다. iptables 및 ip6tables 서비스에 대한 자세한 내용은 5.13절. “iptables를 사용하여 IP 세트 설정 및 제어” 을 참조하십시오.

패킷 전달 활성화

시스템 외부에서 도착한 패킷을 다른 외부 호스트로 전달하려면 커널에서 IP 전달을 활성화해야 합니다. root 로 로그인하고 /etc/sysctl.conf 파일에서 net.ipv4.ip_forward = 0 을 읽는 행을 다음으로 변경합니다.

```
net.ipv4.ip_forward = 1
```

/etc/sysctl.conf 파일에서 변경 사항을 로드하려면 다음 명령을 입력합니다.

```
/sbin/sysctl -p
```

IP 전달이 설정되어 있는지 확인하려면 root 로 다음 명령을 실행합니다.

```
/sbin/sysctl net.ipv4.ip_forward
```

위의 명령에서 1 을 반환하면 IP 전달이 활성화됩니다. 0 을 반환하는 경우 다음 명령을 사용하여 수동으로 설정할 수 있습니다.

```
/sbin/sysctl -w net.ipv4.ip_forward=1
```

4.4.3.2. 추가 리소스

다음은 Reverse Path Forwarding에 대해 자세히 설명하는 리소스입니다.

- *설치된 문서*

  /usr/share/doc/kernel-doc-버전/Documentation/networking/ip-sysctl.txt - 이 파일에는 디렉터리에서 사용할 수 있는 전체 파일 및 옵션이 포함되어 있습니다. 커널 설명서에 처음 액세스하기 전에 root 로 다음 명령을 입력합니다.

  ```
  ~]# yum install kernel-doc
  ```

- *온라인 문서*

  다중 홈 네트워크의 Ingress 필터링에 대한 설명은 RFC 3704 를 참조하십시오.

4.5. DNSSEC로 DNS 트래픽 보안

4.5.1. DNSSEC 소개

DNSSEC는 DNS 클라이언트가 DNS 이름 서버에서 응답의 무결성을 인증하고 점검하여 원래 상태를 확인하고 전송 중에 변조되었는지 확인할 수 있는 일련의DNSSEC(Domain Name System Security Extensions)입니다.

4.5.2. DNSSEC 이해

인터넷 연결을 위해 점점 더 많은 웹 사이트에서 HTTPS 를 사용하여 안전하게 연결할 수 있는 기능을 제공합니다. 그러나 HTTPS 웹 서버에 연결하기 전에 IP 주소를 직접 입력하지 않는 한 DNS 조회를 수행해야 합니다. 이러한 DNS 조회는 안전하지 않게 수행되며 인증 부족으로 인해 중간자 공격을 받습니다. 즉, DNS 클라이언트는 지정된 DNS 이름 서버에서 제공하는 응답이 정품이며 로 변경되지 않았음을 확신할 수 없습니다. 더 중요한 것은 재귀적 이름 서버가 다른 이름 서버에서 얻는 레코드가 정품인지 확인할 수 없다는 것입니다. DNS 프로토콜은 클라이언트의 메시지 가로채기(man-in-the-middle) 공격의 대상이 되지 않도록 하는 메커니즘을 제공하지 않았습니다. DNSSEC는 DNS 를 사용하여 도메인 이름을 확인할 때 인증 및 무결성 검사 부족을 해결하기 위해 도입되었습니다. 이는 기밀성의 문제를 다루지 않습니다.

DNSSEC 정보를 게시하려면 DNS 확인자가 계층적 신뢰 체인을 구축할 수 있도록 하는 방법과 같이 DNS 리소스 레코드에 디지털 서명과 공개 키를 배포해야 합니다. 모든 DNS 리소스 레코드에 대한 디지털 서명이 생성되어 영역에 디지털 서명 리소스 레코드(RRSIG)로 추가됩니다. 영역의 공개 키는 DNSKEY 리소스 레코드로 추가됩니다. 계층적 체인을 빌드하기 위해 DNSKEY의 해시는 상위 영역에 서명(DS) 리소스 레코드 Delegation 으로 게시됩니다. 존재하지 않는 것의 증거를 용이하게 하기 위해, NextSEC (NSEC) 및 NSEC3 리소스 레코드가 사용됩니다. DNSSEC 서명된 영역에서 각 리소스 레코드 세트(RRset)에 해당 RRSIG 리소스 레코드가 있습니다. 하위 영역(NS 및 글루 레코드)에 대한 위임에 사용되는 레코드는 서명되지 않습니다. 이러한 레코드는 하위 영역에 표시되고 여기에 서명됩니다.

DNSSEC 정보를 처리하는 작업은 루트 영역 공개 키로 구성된 확인자에 의해 수행됩니다. 이 키를 사용하여 확인자에서 루트 영역에 사용된 서명을 확인할 수 있습니다. 예를 들어 루트 영역은 .com 의 DS 레코드에 서명했습니다. 루트 영역은 .com 이름 서버에 대한 NS 및 glue 레코드도 제공합니다. 확인자는 이 위임을 따르고 이러한 위임된 이름 서버를 사용하여 .com 의 DNSKEY 레코드에 대한 쿼리를 따릅니다. 가져온 DNSKEY 레코드의 해시는 루트 영역의 DS 레코드와 일치해야 합니다. 그렇다면 확인자는 .com 에 대해 가져온 DNSKEY를 신뢰합니다. .com 영역에서 RRSIG 레코드는 .com DNSKEY에 의해 생성됩니다. 이 프로세스는 .com 내의 위임(예: redhat .com)에 대해 비슷하게 반복됩니다. 이 방법을 사용하면 정상적인 작업 중에 전 세계에서 많은 DNS KEY를 수집하는 동안 하나의 루트 키로 DNS 확인자만 구성하면 됩니다. 암호화 검사에 실패하면 확인자는 SERVFAIL을 애플리케이션에 반환합니다.

DNSSEC는 DNSSEC를 지원하지 않는 애플리케이션과 완전히 보이지 않는 방식으로 설계되었습니다. 비DNSSEC 애플리케이션이 DNSSEC 가능 리졸버를 쿼리하는 경우 RRSIG와 같은 이러한 새로운 리소스 레코드 유형 없이 응답을 받습니다. 그러나 DNSSEC 가능 확인자는 여전히 모든 암호화 검사를 수행하고 악성 DNS 응답을 탐지하는 경우 애플리케이션에 SERVFAIL 오류를 반환합니다. DNSSEC는 DNS 서버(인증 및 재귀) 간 데이터의 무결성을 보호하며, 애플리케이션과 확인자 간에 보안을 제공하지 않습니다. 따라서 애플리케이션에 해당 확인자에게 안전한 전송이 제공되는 것이 중요합니다. 가장 쉬운 방법은 localhost 에서 DNSSEC 가능 확인자를 실행하고 /etc/resolv.conf 에서 127.0.0.1 을 사용하는 것입니다. 또는 원격 DNS 서버에 대한 VPN 연결을 사용할 수 있습니다.

핫스팟 문제 이해

Wi-Fi 핫스팟 또는 VPN “은 DNS에 의존하는 경우”: 유용한 포털은 Wi-Fi 서비스에 대한 인증(또는 결제)에 필요한 페이지로 사용자를 리디렉션하기 위해 DNS 를 납치하는 경향이 있습니다. VPN에 연결하는

사용자는 회사 네트워크 외부에 존재하지 않는 리소스를 찾기 위해 "내부 전용" DNS 서버를 사용해야 하는 경우가 많습니다. 이를 위해서는 소프트웨어의 추가 처리가 필요합니다. 예를 들어 dnssec-trigger 를 사용하여 Hotspot이 DNS 쿼리를 가로채고 unbound 가 프록시 이름 서버로 작동하여 DNSSEC 쿼리를 처리할 수 있는지 감지할 수 있습니다.

DNSSEC 복구 복구 선택

DNSSEC 가능 재귀 확인을 배포하려면 BIND 또는 unbound 를 사용할 수 있습니다. 기본적으로 DNSSEC를 활성화하고 DNSSEC 루트 키를 사용하여 구성됩니다. 서버에서 DNSSEC를 활성화하려면 로컬 사용자가 dnssec-trigger 를 사용할 때 Hotspots에 필요한 DNSSEC 재정의를 동적으로 재구성할 수 있고 Libreswan 을 사용할 때 VPN과 같은 모바일 장치에서 unbound 를 사용하는 것이 좋습니다. 바인딩되지 않은 데몬에서는 서버와 모바일 장치 모두에 유용할 수 있는 etc/ EgressIP/*.d/ 디렉터리에 나열된 DNSSEC 예외의 배포를 추가로 지원합니다.

4.5.3. Dnssec-trigger 이해

unbound 가 /etc/resolv.conf 에 설치되고 구성되면 애플리케이션의 모든 DNS 쿼리는 바인딩되지 않은 .dnssec-trigger 는 트리거된 경우에만 unbound resolver를 재구성합니다. 이는 주로 다른 Wi-Fi 네트워크에 연결되는 로밍 클라이언트 시스템에 적용됩니다. 프로세스는 다음과 같습니다.

- NetworkManager 는 DHCP "를" 통해 새 DNS 서버를 가져올 때 dnssec-trigger 를 트리거합니다.
- 그런 다음 DNSSEC -trigger 는 서버에 대해 여러 테스트를 수행하고 적절하게 DNSSEC를 지원하는지 여부를 결정합니다.
- 이 경우 dnssec-trigger 는 해당 DNS 서버를 모든 쿼리에 대해 전달자로 사용하도록 unbound 를 재구성합니다.
- 테스트에 실패하면 dnssec-trigger 는 새 DNS 서버를 무시하고 몇 가지 사용 가능한 fall-back 방법을 시도합니다.
- 무제한 포트 53(UDP 및 TCP)을 사용할 수 있다고 결정하는 경우, 전달자를 사용하지 않고 unbound 에 전체 재귀 DNS 서버가 되도록 지시합니다.
- 예를 들어 네트워크의 DNS 서버 자체에 도달하기를 제외하고 방화벽에서 포트 53을 차단하기 때문에 DNS를 포트 80에 사용하거나 TLS 캡슐화 DNS 를 포트 443에 사용합니다. 포트 80 및 443에서 DNS 를 실행하는 서버는 /etc/dnssec-trigger/dnssec-trigger.conf 에서 구성할 수 있습니다. 주석 처리된 예제를 기본 구성 파일에서 사용할 수 있어야 합니다.

- 이러한 장애 조치 방법도 실패하면 dnssec-trigger 가 비보안으로 작동하여 DNSSEC를 완전히 무시하거나 새 DNS 쿼리를 시도하지 않고 “캐시에 이미 있는 모든 항목에 대해 응답하는 캐시 전용” 모드에서 실행할 수 있습니다.

Wi-Fi 핫스팟은 인터넷에 대한 액세스 권한을 부여하기 전에 사용자를 사인온 페이지로 리디렉션합니다. 위에 설명된 프로빙 시퀀스 중에 리디렉션이 감지되면 사용자는 인터넷에 액세스해야 하는지 묻는 메시지가 표시됩니다. dnssec-trigger 데몬은 10초마다 DNSSEC 확인자를 계속 검사합니다. dnssec-trigger 그래픽 유틸리티 사용에 대한 자세한 내용은 4.5.8절. “Dnssec-trigger 사용” 을 참조하십시오.

4.5.4. VPN 제공 도메인 및 이름 서버

일부 유형의 VPN 연결은 해당 도메인에 사용할 도메인 및 이름 서버 목록을 VPN 터널 설정의 일부로 전달할 수 있습니다. Red Hat Enterprise Linux 에서는 NetworkManager 가 지원합니다. 즉, unbound,dnssec-trigger 및 NetworkManager 의 조합은 VPN 소프트웨어에서 제공하는 도메인과 이름 서버를 적절하게 지원할 수 있습니다. VPN 터널이 시작되면 로컬 바인딩 되지 않은 캐시가 수신된 도메인 이름의 모든 항목에 대해 플러시되므로 도메인 이름 내의 이름에 대한 쿼리가 VPN을 사용하여 도달한 내부 이름 서버에서 새로 가져옵니다. VPN 터널이 종료되면 도메인에 대한 쿼리가 공용 IP 주소가 반환되고 이전에 가져온 개인 IP 주소가 반환되지 않도록 바인딩 되지 않은 캐시가 다시 플러시됩니다. 4.5.11절. “연결 제공 도메인에 대한 DNSSEC 유효성 검사 구성”을 참조하십시오.

4.5.5. 권장되는 이름 지정 사례

정적 및 일시적인 이름은 모두 host.example.com 과 같이 DNS 의 시스템에 사용되는FQDN(정규화된 도메인 이름)과 일치하도록 권장합니다.

ICANN(Internet Corporation for Assigned Names) 및 번호(ICANN)는 이전에 등록되지 않은 최상위 도메인(예: .yourcompany)을 공개 레지스터에 추가하는 경우가 있습니다. 따라서 Red Hat은 사설 네트워크에서도 위임되지 않은 도메인 이름을 사용하지 않을 것을 강력히 권장합니다. 네트워크 구성에 따라 다르게 확인되는 도메인 이름이 발생할 수 있습니다. 결과적으로 네트워크 리소스를 사용할 수 없게 됩니다. 또한 위임되지 않은 도메인 이름을 사용하면 도메인 이름 충돌에 DNSSEC 검증을 활성화하기 위해 수동으로 구성이 필요하므로 DNSSEC를 배포하고 유지 관리하기가 더 어려워집니다. 이 문제에 대한 자세한 내용은 도메인 이름 충돌에 대한 ICANN FAQ 를 참조하십시오.

4.5.6. Trust Anchors 이해

계층적 암호화 시스템에서 신뢰 앵커 는 신뢰할 수 있는 신뢰할 수 있는 엔티티입니다. 예를 들어 X.509 아키텍처에서 루트 인증서는 신뢰 체인이 파생되는 신뢰 앵커입니다. 신뢰 앵커는 경로 검증을 가능하게하기 위해 사전에 신뢰 당사자를 보유하고 있어야합니다.

DNSSEC 컨텍스트에서 신뢰 앵커는 해당 이름과 연결된 DNS 이름 및 공개 키(또는 공개 키의 해시)로 구성됩니다. 이 키는 기본 64로 인코딩된 키로 표현됩니다. 인증서는 DNS 레코드를 확인하고 인증하는 데 사용할 수 있는 공개 키를 포함한 정보를 교환하는 수단입니다. RFC 4033 은 DNSKEY RR의 구성된 DNSKEY RR 또는 DS RR 해시로 신뢰 앵커를 정의합니다. 보안 인식 확인 확인자는 이 공개 키 또는 해시를 서명된 DNS 응답에 대한 인증 체인을 구축하기 위한 시작점으로 사용합니다. 일반적으로 검증 확인 확인자는 DNS 프로토콜 외부의 보안 또는 신뢰할 수 있는 수단을 통해 신뢰 앵커의 초기 값을 얻어야 합니다. 신뢰 앵커의 존재는 확인자는 신뢰 앵커가 서명 될 영역을 예상해야 함을 의미합니다.

4.5.7. DNSSEC 설치

4.5.7.1. unbound 설치

시스템에서 DNSSEC를 로컬로 사용하여 DNS 를 확인하려면 DNS 확인자 바인딩(또는 바인딩)을 설치해야 합니다. mobile 장치에 dnssec-trigger 를 설치해야 합니다. 서버의 경우 서버가 있는 위치(LAN 또는 인터넷)에 따라 로컬 도메인의 전달 구성이 필요할 수 있지만 unbound 로 충분해야 합니다. dnssec-trigger 는 현재 글로벌 퍼블릭 DNS 영역에만 도움이 됩니다. NetworkManager,dhclient 및 VPN 애플리케이션은 종종 도메인 목록(및 이름 서버 목록)을 자동으로 수집할 수 있지만 dnssec-trigger 또는 unbound 는 수집할 수 없습니다.

unbound 를 설치하려면 root 사용자로 다음 명령을 입력합니다.

```
~]# yum install unbound
```

4.5.7.2. unbound가 실행 중인지 확인

바인딩 되지 않은 데몬이 실행 중인지 확인하려면 다음 명령을 입력합니다.

```
~]$ systemctl status unbound
unbound.service - Unbound recursive Domain Name Server
  Loaded: loaded (/usr/lib/systemd/system/unbound.service; disabled)
  Active: active (running) since Wed 2013-03-13 01:19:30 CET; 6h ago
```

systemctl status 명령은 unbound 서비스가 실행되지 않는 경우 unbound 를 Active: inactive(dead) 로 보고합니다.

4.5.7.3. unbound 시작

현재 세션에 대해 바인딩 되지 않은 데몬을 시작하려면 **root** 사용자로 다음 명령을 입력합니다.

```
~]# systemctl start unbound
```

systemctl enable 명령을 실행하여 시스템이 부팅될 때마다 **unbound** 가 시작되는지 확인합니다.

```
~]# systemctl enable unbound
```

바인딩되지 않은 데몬을 사용하면 로컬 데이터를 구성하거나 다음 디렉터리를 사용하여 재정의할 수 있습니다.

- **/etc/ EgressIP/conf.d** 디렉터리는 특정 도메인 이름에 대한 구성을 추가하는 데 사용됩니다. 도메인 이름에 대한 쿼리를 특정 **DNS** 서버로 리디렉션하는 데 사용됩니다. 이는 종종 기업 **WAN** 내에만 존재하는 하위 도메인에 사용됩니다.

- **/etc/ EgressIP/keys.d** 디렉터리는 특정 도메인 이름에 대한 신뢰 앵커를 추가하는 데 사용됩니다. 이는 내부 전용 이름이 **DNSSEC** 서명되는 경우 필요하지만 신뢰 경로를 구축하기 위한 공개적으로 존재하는 **DS** 레코드가 없습니다. 또 다른 사용 사례는 회사 **WAN** 외부에서 공개적으로 사용 가능한 이름과 다른 **DNSKEY**를 사용하여 도메인의 내부 버전이 서명되는 경우입니다.

- **/etc/cnv/local.d** 디렉터리는 특정 **DNS** 데이터를 로컬 오버라이드로 추가하는 데 사용됩니다. 이는 블랙리스트를 빌드하거나 수동 재정의를 생성하는 데 사용할 수 있습니다. 이 데이터는 **unbound** 를 통해 클라이언트에 반환되지만 **DNSSEC** 서명으로 표시되지 않습니다.

NetworkManager 및 일부 **VPN** 소프트웨어는 구성을 동적으로 변경할 수 있습니다. 이러한 구성 디렉터리에는 주석 처리된 예제 항목이 포함되어 있습니다. 자세한 내용은 **unbound.conf(5)** 매뉴얼 페이지를 참조하십시오.

4.5.7.4. Dnssec-trigger 설치

dnssec-trigger 애플리케이션은 데몬인 **dnssec-triggerd** 로 실행됩니다. **dnssec-trigger** 를 설치하려면 **root** 사용자로 다음 명령을 입력합니다.

```
~]# yum install dnssec-trigger
```

4.5.7.5. Dnssec-trigger Daemon이 실행 중인지 확인

dnssec-triggerd 가 실행 중인지 확인하려면 다음 명령을 입력합니다.

```
~]$ systemctl status dnssec-triggerd
systemctl status dnssec-triggerd.service
dnssec-triggerd.service - Reconfigure local DNS(SEC) resolver on network change
  Loaded: loaded (/usr/lib/systemd/system/dnssec-triggerd.service; enabled)
  Active: active (running) since Wed 2013-03-13 06:10:44 CET; 1h 41min ago
```

dnssec-triggerd 데몬이 실행되지 않는 경우 **systemctl status** 명령은 **dnssec-triggerd** 를 **Active: inactive(dead)** 로 보고합니다. 현재 세션에 대해 시작하려면 **root** 사용자로 다음 명령을 입력합니다.

```
~]# systemctl start dnssec-triggerd
```

systemctl enable 명령을 실행하여 시스템을 부팅할 때마다 **dnssec-triggerd** 가 시작되는지 확인합니다.

```
~]# systemctl enable dnssec-triggerd
```

4.5.8. Dnssec-trigger 사용

dnssec-trigger 애플리케이션에는 **DNSSEC** 프로브 결과를 표시하고 필요에 따라 **DNSSEC** 프로브 요청을 수행하기 위한 **GNOME** 패널 유틸리티가 있습니다. 유틸리티를 시작하려면 **Super** 키를 눌러 **Activities Overview(활동 개요)**를 입력하고 **DNSSEC** 를 입력한 다음 **Enter** 키를 누릅니다. 배 앵커를 다시 조립하는 아이콘은 화면 하단에 있는 메시지 트레이에 추가됩니다. 화면 오른쪽 하단에 있는 라운드 파란 알림 아이콘을 눌러 표시합니다. 앵커 아이콘을 마우스 오른쪽 버튼으로 클릭하여 팝업 메뉴를 표시합니다.

일반 작업에서 **unbound** 는 이름 서버로 로컬로 사용되며 **resolv.conf** 는 **127.0.0.1** 을 가리킵니다. **Hotspot Sign-On** 패널에서 **OK** 를 클릭하면 이 내용이 변경됩니다. **DNS** 서버는 **NetworkManager** 에서 쿼리하고 **resolv.conf** 에 배치됩니다. 이제 **Hotspot**의 **Sign-On** 페이지에서 인증할 수 있습니다. 앵커 아이콘은 **DNS** 쿼리를 안전하지 않게 경고하는 큰 빨간색 느낌표를 표시합니다. 인증되면 **dnssec-trigger**

*는 이를 자동으로 탐지하고 보안 모드로 다시 전환해야 하지만 경우에 따라 **Reprobe** 을 선택하여 수동으로 이 작업을 수행해야 합니다.*

DNSSEC-trigger 는 일반적으로 사용자 개입이 필요하지 않습니다. 시작된 후에는 백그라운드에서 작동하며 문제가 발생하면 팝업 텍스트 상자를 통해 사용자에게 알립니다. 또한 resolv.conf 파일의 변경 사항에 대해 unbound 에게 알립니다.

4.5.9. DNSSEC로 dig 사용

DNSSEC가 작동하는지 여부를 확인하려면 다양한 명령줄 툴을 사용할 수 있습니다. 사용할 가장 좋은 도구는 bind-utils 패키지의 dig 명령입니다. 유용한 기타 툴은 unbound 패키지의 ldns 패키지 및 unbound-host 의 드릴 입니다. 이전 DNS 유틸리티 nslookup 및 host 는 더 이상 사용되지 않으므로 사용해서는 안 됩니다.

dig 를 사용하여 DNSSEC 데이터를 요청하는 쿼리를 보내려면 +dnssec 옵션이 명령에 추가됩니다. 예를 들면 다음과 같습니다.

```
~]$ dig +dnssec whitehouse.gov
; <<>> DiG 9.9.3-rl.13207.22-P2-RedHat-9.9.3-4.P2.el7 <<>> +dnssec whitehouse.gov
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21388
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;whitehouse.gov.   IN A

;; ANSWER SECTION:
whitehouse.gov.  20 IN A 72.246.36.110
whitehouse.gov.  20 IN RRSIG A 7 2 20 20130825124016 20130822114016 8399
whitehouse.gov. BB8VHWEklaKpaLprt3hq1GkjDROvkmjYTBxiGhuki/BJn3PolGyrftxR
HH0377l0Lsybj/uZv5hL4UwWd/lw6Gn8GPikqhztAkgMxddMQ2lARP6p
wbMOKbSUuV6NGUT1WWwpbi+LelFMqQcAq3Se66iyH0Jem7HtgPEUE1Zc 3ol=

;; Query time: 227 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Aug 22 22:01:52 EDT 2013
;; MSG SIZE  rcvd: 233
```

A 레코드 외에도 DNSSEC 서명을 포함하는 RRSIG 레코드와 서명의 만료 시간 및 만료 시간이 반환됩니다. 바인딩되지 않은 서버에서는 상단에 있는 flags: 섹션에 ad bit를 반환하여 데이터가 DNSSEC 인증되었음을 표시했습니다.

DNSSEC 검증에 실패하면 dig 명령에서 SERVFAIL 오류를 반환합니다.

```
~]$ dig badsign-a.test.dnssec-tools.org
; <<>> DiG 9.9.3-rl.156.01-P1-RedHat-9.9.3-3.P1.el7 <<>> badsign-a.test.dnssec-tools.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 1010
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;badsign-a.test.dnssec-tools.org. IN A

;; Query time: 1284 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Aug 22 22:04:52 EDT 2013
;; MSG SIZE  rcvd: 60]
```

오류에 대한 자세한 정보를 요청하려면 **dig** 명령에 **+cd** 옵션을 지정하여 **DNSSEC** 검사를 비활성화할 수 있습니다.

```
~]$ dig +cd +dnssec badsign-a.test.dnssec-tools.org
; <<>> DiG 9.9.3-rl.156.01-P1-RedHat-9.9.3-3.P1.el7 <<>> +cd +dnssec badsign-a.test.dnssec-
tools.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 26065
;; flags: qr rd ra cd; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;badsign-a.test.dnssec-tools.org. IN A

;; ANSWER SECTION:
badsign-a.test.dnssec-tools.org. 49 IN A 75.119.216.33
badsign-a.test.dnssec-tools.org. 49 IN RRSIG A 5 4 86400 20130919183720 20130820173720
19442 test.dnssec-tools.org.
E572dLKMvYB4cgTRyAHIKKEvdOP7tockQb7hXFNZKVbfXbZJOlDREJrr
zCgAfJ2hykfY0yJHAlnuQvM0s6xOnNBSvc2xLlybJdfTaN6kSR0YFdYZ
n2NpPctn2kUBn5UR1BJRin3Gqy20LZlZx2KD7cZBtieMsU/lunyhCSc0 kYw=

;; Query time: 1 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Aug 22 22:06:31 EDT 2013
;; MSG SIZE  rcvd: 257
```

DNSSEC 실수는 종종 열화되거나 만료 시간이 발생하지만, 이 예에서 www.dnssec-tools.org 의 사람들은 이 RRSIG 서명을 수동으로 보고 탐지할 수 없었습니다. 이 오류는 systemctl status unbound 의 출력에 표시되고 unbound 데몬은 다음과 같이 이러한 오류를 syslog 에 기록합니다.

> Aug 22 22:04:52 laptop unbound: [3065:0] info: validation failure badsign-a.test.dnssec-tools.org. A IN

unbound-host 를 사용하는 예:

```
~]$ unbound-host -C /etc/unbound/unbound.conf -v whitehouse.gov
whitehouse.gov has address 184.25.196.110 (secure)
whitehouse.gov has IPv6 address 2600:1417:11:2:8800::fc4 (secure)
whitehouse.gov has IPv6 address 2600:1417:11:2:8000::fc4 (secure)
whitehouse.gov mail is handled by 105 mail1.eop.gov. (secure)
whitehouse.gov mail is handled by 110 mail5.eop.gov. (secure)
whitehouse.gov mail is handled by 105 mail4.eop.gov. (secure)
whitehouse.gov mail is handled by 110 mail6.eop.gov. (secure)
whitehouse.gov mail is handled by 105 mail2.eop.gov. (secure)
whitehouse.gov mail is handled by 105 mail3.eop.gov. (secure)
```

4.5.10. Dnssec-trigger에 대한 Hotspot Detection Infrastructure 설정

네트워크에 연결할 때 dnssec-trigger 는 핫스팟을 탐지합니다. 핫스팟은 일반적으로 네트워크 리소스를 사용하기 전에 웹 페이지와의 사용자 상호 작용을 강제하는 장치입니다. 탐지는 알려진 콘텐츠로 특정 고정 웹 페이지를 다운로드하여 수행됩니다. 핫스팟이 있는 경우 수신한 콘텐츠가 예상대로 표시되지 않습니다.

dnssec-trigger 에서 Hotspot을 감지하는 데 사용할 수 있는 알려진 콘텐츠가 있는 고정 웹 페이지를 설정하려면 다음과 같이 진행하십시오.

1. 인터넷에서 공개적으로 연결할 수 있는 일부 시스템에서 웹 서버를 설정합니다. Red Hat Enterprise Linux 7 시스템 관리자 가이드의 웹 서버 장을 참조하십시오.

2. 서버가 실행되면 알려진 콘텐츠가 있는 정적 페이지를 게시합니다. 페이지가 유효한 HTML 페이지일 필요는 없습니다. 예를 들어 OK 문자열만 포함된 hotspot.txt 라는 일반 텍스트 파일을 사용할 수 있습니다. 서버가 example.com 에 있고 웹 서버 document_root/static/ 하위 디렉터리에 hotspot.txt 파일을 게시했다고 가정하면 정적 웹 페이지에 대한 주소는 example.com/static/hotspot.txt 입니다. Red Hat Enterprise Linux 7 시스템 관리자 가이드의 웹 서버 장에 있는 DocumentRoot 지시문을 참조하십시오.

3. */etc/dnssec-trigger/dnssec-trigger.conf* 파일에 다음 행을 추가합니다.

```
url: "http://example.com/static/hotspot.txt OK"
```

이 명령은 **HTTP (포트 80)**를 사용하여 프로브된 **URL**을 추가합니다. 첫 번째 부분은 해결 될 **URL**과 다운로드 페이지입니다. 명령의 두 번째 부분은 다운로드한 웹 페이지에 포함될 것으로 예상되는 텍스트 문자열입니다.

설정 옵션에 대한 자세한 내용은 **man** 페이지 **dnssec-trigger.conf(8)** 을 참조하십시오.

4.5.11. 연결 제공 도메인에 대한 DNSSEC 유효성 검사 구성

기본적으로 적절한 이름 서버가 있는 전달 영역은 **NetworkManager** 를 통한 **Wi-Fi** 연결을 제외한 모든 연결에서 제공하는 모든 도메인에 대해 **dnssec-trigger** 에 의해 바인딩 되지 않습니다. 기본적으로 **unbound** 에 추가된 모든 전달 영역은 **DNSSEC**의 유효성을 검사합니다.

전달 영역의 유효성을 확인하는 기본 동작을 변경할 수 있으므로 모든 전달 영역을 기본적으로 검증할 수 없습니다. 이렇게 하려면 **dnssec-trigger** 구성 파일 **/etc/dnssec.conf** 에서 **validate_connection_provided_zones** 변수를 변경합니다. **root** 사용자로 다음과 같이 행을 열고 편집합니다.

```
validate_connection_provided_zones=no
```

기존 전달 영역에는 변경 사항이 적용되지 않지만 향후 영역의 경우만 적용됩니다. 따라서 현재 제공된 도메인에 대해 **DNSSEC**를 비활성화하려면 다시 연결해야 합니다.

4.5.11.1. Wi-Fi Supplied 도메인에 대한 DNSSEC 유효성 검사 구성

Wi-Fi 제공 영역에 대한 정방향 영역을 추가할 수 있습니다. 이렇게 하려면 **dnssec-trigger** 구성 파일 **/etc/dnssec.conf** 에서 **add_wifi_provided_zones** 변수를 변경합니다. **root** 사용자로 다음과 같이 행을 열고 편집합니다.

```
add_wifi_provided_zones=yes
```

기존 전달 영역에는 변경 사항이 적용되지 않지만 향후 영역의 경우만 적용됩니다. 따라서 현재 **Wi-Fi** 제공 도메인에 대해 **DNSSEC**를 활성화하려면 **Wi-Fi** 연결을 다시 연결(다시 시작)해야 합니다.

주의

unbound 로의 정방향 영역으로 **Wi-Fi** 제공 도메인을 추가하는 경우 다음과 같은 보안 영향을 미칠 수 있습니다.

1. **Wi-Fi** 액세스 포인트는 권한이 없으며 모든 **DNS** 쿼리를 해당 **DNS** 서버로 라우팅할 수 없는 **DHCP** 를 통해 도메인을 의도적으로 제공할 수 있습니다.

2. 전달 영역에 대한 **DNSSEC** 검증이 멈춘 경우 **Wi-Fi** 제공 **DNS** 서버는 이를 인식하지 않고 제공된 도메인에서 도메인 이름에 대한 **IP** 주소를 스푸핑할 수 있습니다.

4.5.12. 추가 리소스

다음은 **DNSSEC**에 대해 자세히 설명하는 리소스입니다.

4.5.12.1. 설치된 문서

- **DNSSEC-trigger(8)** 매뉴얼 페이지 - **dnssec -triggerd,dnssec-trigger-control** 및 **dnssec-trigger-panel** 에 대한 명령 옵션에 대해 설명합니다.
- **DNSSEC-trigger.conf(8)** 매뉴얼 페이지 - **dnssec-triggerd** 의 설정 옵션에 대해 설명합니다.
- **unbound(8)** 도움말 페이지 - **DNS** 검증 확인 확인자인 **unbound** 에 대한 명령 옵션에 대해 설명합니다.
- **unbound.conf(5)** 매뉴얼 페이지 - **unbound** 를 구성하는 방법에 대한 정보가 포함되어 있습니다.
- **resolv.conf(5)** 도움말 페이지 - 리졸버 루틴이 읽은 정보를 포함합니다.

4.5.12.2. 온라인 문서

http://tools.ietf.org/html/rfc4033

RFC 4033 DNS 보안 소개 및 요구 사항.

http://www.dnssec.net/

많은 DNSSEC 리소스에 대한 링크가 있는 웹 사이트.

http://www.dnssec-deployment.org/

Homeland Security 부서가 후원하는 DNSSEC 배포 이니셔티브에는 많은 DNSSEC 정보가 포함되어 있으며 DNSSEC 배포 문제를 설명하는 메일링 목록이 있습니다.

http://www.internetsociety.org/deploy360/dnssec/community/

DNSSEC 배포를 촉진하고 조정하는 Internet Society의 "Deploy360" 이니셔티브는 전 세계 커뮤니티와 DNSSEC 활동을 찾는 데 유용한 리소스입니다.

http://www.unbound.net/

이 문서에는 바인딩되지 않은 DNS 서비스에 대한 일반 정보가 포함되어 있습니다.

http://www.nlnetlabs.nl/projects/dnssec-trigger/

이 문서에는 dnssec-trigger 에 대한 일반 정보가 포함되어 있습니다.

4.6. LIBRESWAN을 사용하여 VPN(VIRTUAL PRIVATE NETWORK) 보안

Red Hat Enterprise Linux 7에서는 Libreswan 애플리케이션에서 지원하는 IPsec 프로토콜을 사용하여VPN(Virtual Private Network)을 구성할 수 있습니다. Libreswan 은 Openswan 애플리케이션에 대한 연속이며, Openswan 문서의 많은 예제는 Libreswan 과 교환 가능합니다. NetworkManager IPsec 플러그인을 NetworkManager-libreswan 이라고 합니다. GNOME Shell 사용자는 NetworkManager-libreswan-gnome 패키지를 설치해야 하며 NetworkManager-libreswan 을 종속성으로 사용해야 합니다. NetworkManager-libreswan-gnome 패키지는 선택적 채널에서만 사용할 수 있습니다. 추가 리포지토리 *활성화 및 선택적 리포지토리*를 참조하십시오.

VPN용 IPsec 프로토콜은 자체적으로 인터넷 키 교환 (IKE) 프로토콜을 사용하여 구성됩니다. IPsec과 IKE라는 용어는 서로 바꿔 사용할 수 있습니다. IPsec VPN은 IKE VPN, IKEv2 VPN, XAUTH VPN, Cisco VPN 또는 IKE/IPsec VPN이라고도 합니다. L2TP(Level 2 Tunneling Protocol)을 사용하는 IPsec VPN 변형은 일반적으로 선택적 채널 xl2tpd 애플리케이션이 필요한 L2TP/IPsec VPN이라고 합니다.

Libreswan 은 Red Hat Enterprise Linux 7에서 사용 가능한 오픈 소스 사용자 공간 IKE 구현입니다. IKE 버전 1 및 2는 사용자 수준 데몬으로 구현됩니다. 또한IKE 프로토콜 자체도 암호화되어 있습니다. IPsec 프로토콜은 Linux 커널에 의해 구현되며 Libreswan 은 VPN 터널 구성을 추가 및 제거하도록 커널을 설정합니다.

IKE 프로토콜은 UDP 포트 500 및 4500을 사용합니다. IPsec 프로토콜은 프로토콜 번호 50이 있고, 프로토콜 번호 51인 인증된 헤더 (AH)와 프로토콜 번호 51인 Encapsulated Security Payload (ESP)라는 두 가지 프로토콜로 구성됩니다. AH 프로토콜은 사용하지 않는 것이 좋습니다. AH 사용자는 null 암호화를 사용하여 ESP 로 마이그레이션하는 것이 좋습니다.

IPsec 프로토콜에는 터널 모드(기본값) 및 전송 모드 의 두 가지 작동 모드가 있습니다. IKE 없이 IPsec을 사용하여 커널을 구성할 수 있습니다. 이를 Manual Keying이라고 합니다. ip xfrm 명령을 사용하여 수동 키 처리를 설정할 수 있지만 보안상의 이유로 강력히 권장되지 않습니다. netlink를 사용하여 Linux 커널과 Libreswan 인터페이스. 패킷 암호화 및 암호 해독은 Linux 커널에서 수행됩니다.

Libreswan 은NSS(Network Security Services) 암호화 라이브러리를 사용합니다. libreswan 및 NSS는 모두 Federal Information Processing Standard (FIPS)PublicationRuntimeConfig와 함께 사용하도록 인증되었습니다.

중요

Libreswan 및 Linux 커널에서 구현되는 IKE/IPsec VPN은 Red Hat Enterprise Linux 7에서 사용할 수 있는 유일한 VPN 기술입니다. 이러한 위험에 대해 이해하지 않고 다른 VPN 기술을 사용하지 마십시오.

4.6.1. Libreswan 설치

Libreswan 을 설치하려면 root 로 다음 명령을 입력합니다.

```
~]# yum install libreswan
```

Libreswan 이 설치되어 있는지 확인하려면 다음을 수행하십시오.

```
~]$ yum info libreswan
```

Libreswan 의 새 설치 후 설치 프로세스의 일부로 **NSS** 데이터베이스를 초기화해야 합니다. 새 데이터베이스를 시작하기 전에 다음과 같이 이전 데이터베이스를 제거합니다.**Before you start a new database, remove the old database as follows:**

```
~]# systemctl stop ipsec
~]# rm /etc/ipsec.d/*db
```

그런 다음 새 **NSS** 데이터베이스를 초기화하려면 **root** 로 다음 명령을 입력합니다.

```
~]# ipsec initnss
Initializing NSS database
```

FIPS 모드에서만 작동하는 경우에만 **NSS** 데이터베이스를 암호로 보호해야 합니다. 이전 명령 대신 **FIPS** 모드의 데이터베이스를 초기화하려면 다음을 사용합니다.

```
~]# certutil -N -d sql:/etc/ipsec.d
Enter a password which will be used to encrypt your keys.
The password should be at least 8 characters long,
and should contain at least one non-alphabetic character.

Enter new password:
Re-enter password:
```

Libreswan 에서 제공하는 **ipsec** 데몬을 시작하려면 **root** 로 다음 명령을 실행합니다.

```
~]# systemctl start ipsec
```

데몬이 현재 실행 중인지 확인하려면 다음을 수행하십시오.

```
~]$ systemctl status ipsec
* ipsec.service - Internet Key Exchange (IKE) Protocol Daemon for IPsec
   Loaded: loaded (/usr/lib/systemd/system/ipsec.service; disabled; vendor preset: disabled)
   Active: active (running) since Sun 2018-03-18 18:44:43 EDT; 3s ago
     Docs: man:ipsec(8)
           man:pluto(8)
           man:ipsec.conf(5)
  Process: 20358 ExecStopPost=/usr/sbin/ipsec --stopnflog (code=exited, status=0/SUCCESS)
  Process: 20355 ExecStopPost=/sbin/ip xfrm state flush (code=exited, status=0/SUCCESS)
  Process: 20352 ExecStopPost=/sbin/ip xfrm policy flush (code=exited, status=0/SUCCESS)
  Process: 20347 ExecStop=/usr/libexec/ipsec/whack --shutdown (code=exited, status=0/SUCCESS)
  Process: 20634 ExecStartPre=/usr/sbin/ipsec --checknflog (code=exited, status=0/SUCCESS)
  Process: 20631 ExecStartPre=/usr/sbin/ipsec --checknss (code=exited, status=0/SUCCESS)
```

```
  Process: 20369 ExecStartPre=/usr/libexec/ipsec/_stackmanager start (code=exited,
status=0/SUCCESS)
  Process: 20366 ExecStartPre=/usr/libexec/ipsec/addconn --config /etc/ipsec.conf --checkconfig
(code=exited, status=0/SUCCESS)
 Main PID: 20646 (pluto)
   Status: "Startup completed."
   CGroup: /system.slice/ipsec.service
           └─20646 /usr/libexec/ipsec/pluto --leak-detective --config /etc/ipsec.conf --nofork
```

Libreswan 이 시스템을 시작할 때 시작되도록 하려면 **root** 로 다음 명령을 실행합니다.

```
~]# systemctl enable ipsec
```

ipsec 서비스를 허용하도록 모든 중간 및 호스트 기반 방화벽을 구성합니다. 방화벽에 대한 정보는 *5장. 방화벽 사용* 를 참조하십시오. 특정 서비스를 통과할 수 있도록 허용하십시오. **Libreswan** 에서 다음 패킷을 허용하도록 방화벽이 필요합니다.

- 인터넷 키 교환 **(IKE)** 프로토콜용 **UDP** 포트 **500** 및 **4500**
- 프로토콜 **50: Encapsulated Security Payload (ESP) IPsec** 패킷
- 인증된 헤더 **(AH) IPsec** 패킷의 경우 프로토콜 **51**

Libreswan 을 사용하여 **IPsec VPN**을 설정하는 방법에 대한 세 가지 예를 제공합니다. 첫 번째 예는 두 개의 호스트를 함께 연결하여 안전하게 통신할 수 있도록 하는 것입니다. 두 번째 예는 두 사이트를 함께 연결하여 하나의 네트워크를 구성하는 것입니다. 세 번째 예는이 컨텍스트에서 대행사로 알려진 원격 사용자를 지원하는 것입니다.

4.6.2. Libreswan을 사용하여 VPN 구성 생성

Libreswan 은 **IKE/IPsec**이 피어 프로토콜로 피어링되기 때문에 “소스” 및 “대상” 또는 “서버” 및 “클라이언트” 라는 용어를 사용하지 않습니다. 대신 “왼쪽” 및 “오른쪽” 용어를 사용하여 엔드 포인트(호스트)를 참조합니다. 이렇게 하면 대부분의 경우 두 끝점에서 동일한 구성을 사용할 수 있지만 많은 관리자가 항상 로컬 호스트에 “**left**” 를 사용하고 원격 호스트에 대해 “오른쪽” 을 사용합니다.

일반적으로 끝점 인증에 사용되는 네 가지 방법이 있습니다.

- 공유 키 **(PSK)**는 가장 간단한 인증 방법입니다. **PSK**는 임의의 문자로 구성되어야 하며 길이가 **20**자 이상이어야 합니다. **FIPS** 모드에서 **PSK**는 사용된 무결성 알고리즘에 따라 최소 강도 요

구 사항을 준수해야 합니다. 64 랜덤 문자보다 짧은 PSK를 사용하지 않는 것이 좋습니다.

- 원시 RSA 키는 일반적으로 정적 호스트-호스트 또는 서브넷-to-subnet IPsec 구성에 사용됩니다. 호스트는 서로의 공개 RSA 키로 수동으로 구성됩니다. 이 방법은 수십 개 이상의 호스트가 모두 IPsec 터널을 서로 설정해야 하는 경우 잘 확장되지 않습니다.

- X.509 인증서는 일반적으로 일반적인 IPsec 게이트웨이에 연결해야 하는 여러 호스트가 있는 대규모 배포에 사용됩니다. 중앙 인증 기관 (CA)은 호스트 또는 사용자를 위해 RSA 인증서에 서명하는 데 사용됩니다. 이 중앙 CA는 개별 호스트 또는 사용자의 취소를 포함하여 신뢰를 중계해야 합니다.

- NULL 인증은 인증 없이 메시 암호화를 얻는 데 사용됩니다. 이는 수동 공격으로부터 보호하지만 적극적인 공격으로부터 보호하지는 않습니다. 그러나 IKEv2 에서는 symmetric 인증 방법을 허용하므로 클라이언트가 서버를 인증하지만 서버는 클라이언트를 인증하지 않는 인터넷 스케일 Opportunistic IPsec에도 NULL 인증을 사용할 수 있습니다. 이 모델은 TLS (https:// 웹 사이트라고도 함)를 사용하는 보안 웹 사이트와 유사합니다.

이러한 인증 방법 외에도 양자 컴퓨터에 의해 가능한 공격으로부터 보호하기 위해 추가 인증을 추가할 수 있습니다. 이러한 추가 인증 방법을 Postquantum Preshared Keys (PPK)라고 합니다. 개별 클라이언트 또는 클라이언트 그룹은 대역 외 구성된 PreShared 키에 해당하는 PPK(PPKID)를 지정하여 자체 PPK를 사용할 수 있습니다. 4.6.9절. “Quantum Computers에 대한 보호 사용”을 참조하십시오.

4.6.3. Libreswan을 사용하여 호스트-호스트 VPN 만들기

“왼쪽” 과 “오른쪽” 이라는 두 호스트 간에 호스트 간 IPsec VPN을 생성하도록 Libreswan 을 구성하려면 새로운 원시 RSA 키 쌍을 생성하기 위해 두 호스트 모두에 root 로 다음 명령을 입력합니다.“” “”

```
~]# ipsec newhostkey --output /etc/ipsec.d/hostkey.secrets
Generated RSA key pair with CKAID 14936e48e756eb107fa1438e25a345b46d80433f was stored in
the NSS database
```

이를 통해 호스트에 대한 RSA 키 쌍이 생성됩니다. RSA 키를 생성하는 프로세스는 특히 엔트로피가 낮은 가상 머신에서 몇 분이 걸릴 수 있습니다.

호스트에 공개 키를 “왼쪽으로” 지정할 수 있도록 호스트 공개 키를 보려면 “newhostkey” 명령에서 반환한 CKAID를 사용하여 새 hostkey가 추가된 호스트에서 root 로 다음 명령을 실행합니다.

```
~]# ipsec showhostkey --left --ckaid 14936e48e756eb107fa1438e25a345b46d80433f
 # rsakey AQPFKElpV
```

```
leftrsasigkey=0sAQPFKElpV2GdCF0Ux9Kqhcap53Kaa+uCgduoT2I3x6LkRK8N+GiVGkRH4Xg+WMrz
Rb94kDDD8m/BO/Md+A30u0NjDk724jWuUU215rnpwvbdAob8pxYc4ReSgjQ/DkqQvsemoeF4kimMU1
OBPNU7lBw4hTBFzu+iVUYMELwQSXpremLXHBNIamUbe5R1+ibgxO19l/PAbZwxyGX/ueBMBvSQ+H
0UqdGKbq7UgSEQTFa4/gqdYZDDzx55tpZk2Z3es+EWdURwJOgGiiiIFuBagasHFpeu9Teb1VzRyytny
NiJCBVhWVqsB4h6eaQ9RpAMmqBdBeNHfXwb6/hg+JIKJgjidXvGtgWBYNDpG40fEFh9USaFlSdiHO+
dmGyZQ74Rg9sWLtiVdlH1YEBUtQb8f8FVry9wSn6AZqPlpGgUdtkTYUCaaifsYH4hoIA0nku4Fy/Ugej8
9ZdrSN7Lt+igns4FysMmBOl9Wi9+LWnfl+dm4Nc6UNgLE8kZc+8vMJGkLi4SYjk2/MFYgqGX/COxSCPE
FUZFiNK7Wda0kWea/FqE1heem7rvKAPIiqMymjSmytZl9hhkCD16pCdgrO3fJXsfAUChYYSPyPQClkav
vBL/wNK9zlaOwssTaKTj4Xn90SrZaxTEjpqUeQ==
```

*아래에 설명된 대로 두 호스트의 구성 파일에 추가하려면 이 키가 필요합니다. **CKAID**를 잊어 버린 경우 다음을 사용하여 시스템의 모든 호스트 키 목록을 얻을 수 있습니다.*

```
~]# ipsec showhostkey --list
< 1 >  RSA keyid: AQPFKElpV ckaid: 14936e48e756eb107fa1438e25a345b46d80433f
```

*키 쌍의 시크릿 부분은 **/etc/ipsec.d/*.db** 에 있는 "**NSS** 데이터베이스에" 저장됩니다.*

*이 호스트 간 터널에 대한 구성 파일을 만들기 위해 왼쪽**rsasigkey=** 및 **rightrsasigkey= from** 위의 줄은 **/etc/ipsec.d/** 디렉터리에 배치된 사용자 지정 구성 파일에 추가됩니다.*

***root** 로 실행되는 편집기를 사용하여 다음 형식으로 적절한 이름으로 파일을 생성합니다.*

```
/etc/ipsec.d/my_host-to-host.conf
```

다음과 같이 파일을 편집합니다.

```
conn mytunnel
    leftid=@west.example.com
    left=192.1.2.23
    leftrsasigkey=0sAQOrlo+hOafUZDlCQmXFrje/oZm [...] W2n417C/4urYHQkCvuIQ==
    rightid=@east.example.com
    right=192.1.2.45
    rightrsasigkey=0sAQO3fwC6nSSGgt64DWiYZzuHbc4 [...] D/v8t5YTQ==
    authby=rsasig
    # load and initiate automatically
    auto=start
```

*공개 키는 **RSAID** 대신 **CKAID**로 구성할 수도 있습니다. 이 경우 "**leftrsasigkey=**"대신 "**leftckaid=**" 을 사용합니다.*

*왼쪽 및 오른쪽 호스트 모두에서 동일한 구성 파일을 사용할 수 있습니다. **Libreswan**이 지정된 **IP** 주소*

또는 호스트 이름을 기반으로 "왼쪽" 인지 "오른쪽" 인지 자동으로 감지합니다. 호스트 중 하나가 모바일 호스트인 경우 **IP** 주소를 미리 알 수 없는 경우 모바일 클라이언트에서 **%defaultroute** 를 **IP** 주소로 사용합니다. 그러면 동적 **IP** 주소가 자동으로 선택됩니다. 들어오는 모바일 호스트의 연결을 수락하는 정적 서버 호스트에서 **IP** 주소에 **%any** 을 사용하여 모바일 호스트를 지정합니다.

leftrsasigkey 값이 "왼쪽" 호스트에서 가져와 **rightrsasigkey** 값을 "올바른" 호스트에서 가져왔는지 확인합니다. **right ckaid**와 **rightckaid** 를 사용할 때도 적용됩니다.

ipsec 을 재시작하여 새 구성을 읽고 부팅 시 시작되도록 구성된 경우 터널이 설정되었는지 확인합니다.

```
~]# systemctl restart ipsec
```

auto=start 옵션을 사용하면 **IPsec** 터널을 몇 초 내에 설정해야 합니다. 다음 명령을 루트로 입력하여 수동으로 터널을 로드하고 시작할 수 있습니다.

```
~]# ipsec auto --add mytunnel
~]# ipsec auto --up mytunnel
```

4.6.3.1. Libreswan을 사용하여 호스트-호스트 VPN 확인

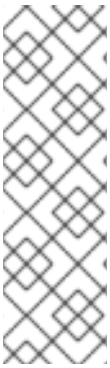
IKE 협상은 **UDP** 포트 **500** 및 **4500**에서 이루어집니다. **IPsec** 패킷은 **Encapsulated Security Payload (ESP)** 패킷으로 표시됩니다. **ESP** 프로토콜에는 포트가 없습니다. **VPN** 연결이 **NAT** 라우터를 통과해야 하는 경우 **ESP** 패킷은 포트 **4500**의 **UDP** 패킷으로 캡슐화됩니다.

패킷이 **VPN** 터널을 통해 전송되는지 확인하려면 다음 형식으로 **root** 로 명령을 실행합니다.

```
~]# tcpdump -n -i interface esp or udp port 500 or udp port 4500
00:32:32.632165 IP 192.1.2.45 > 192.1.2.23: ESP(spi=0x63ad7e17,seq=0x1a), length 132
00:32:32.632592 IP 192.1.2.23 > 192.1.2.45: ESP(spi=0x4841b647,seq=0x1a), length 132
00:32:32.632592 IP 192.0.2.254 > 192.0.1.254: ICMP echo reply, id 2489, seq 7, length 64
00:32:33.632221 IP 192.1.2.45 > 192.1.2.23: ESP(spi=0x63ad7e17,seq=0x1b), length 132
00:32:33.632731 IP 192.1.2.23 > 192.1.2.45: ESP(spi=0x4841b647,seq=0x1b), length 132
00:32:33.632731 IP 192.0.2.254 > 192.0.1.254: ICMP echo reply, id 2489, seq 8, length 64
00:32:34.632183 IP 192.1.2.45 > 192.1.2.23: ESP(spi=0x63ad7e17,seq=0x1c), length 132
00:32:34.632607 IP 192.1.2.23 > 192.1.2.45: ESP(spi=0x4841b647,seq=0x1c), length 132
00:32:34.632607 IP 192.0.2.254 > 192.0.1.254: ICMP echo reply, id 2489, seq 9, length 64
00:32:35.632233 IP 192.1.2.45 > 192.1.2.23: ESP(spi=0x63ad7e17,seq=0x1d), length 132
00:32:35.632685 IP 192.1.2.23 > 192.1.2.45: ESP(spi=0x4841b647,seq=0x1d), length 132
00:32:35.632685 IP 192.0.2.254 > 192.0.1.254: ICMP echo reply, id 2489, seq 10, length 64
```

인터페이스는 트래픽을 전달하는 것으로 알려진 인터페이스입니다. **tcpdump** 로 캡처를 종료하려면

Ctrl+C 누릅니다.



참고

tcpdump 명령은 **IPsec** 과 약간의 예기치 않게 상호 작용합니다. 발신된 일반 텍스트 패킷이 아닌 발신 암호화된 패킷만 볼 수 있습니다. 암호화된 수신 패킷과 암호가 해독된 수신 패킷을 확인할 수 있습니다. 가능한 경우 두 시스템 간의 라우터에서 **tcpdump** 를 실행하고 엔드포인트 자체에서 실행되지 않습니다. **VTI(Virtual Tunnel Interface)**를 사용하는 경우 물리적 인터페이스의 **tcpdump**는 **ESP** 패킷을 표시하는 반면, **VTI** 인터페이스의 **tcpdump**에는 일반 텍스트 트래픽이 표시됩니다.

터널이 비정상적으로 설정되었는지 확인하고 터널을 통해 트래픽이 얼마나 사라졌는지 추가로 확인하려면 **root** 로 다음 명령을 입력합니다.

```
~]# ipsec whack --trafficstatus
006 #2: "mytunnel", type=ESP, add_time=1234567890, inBytes=336, outBytes=336, id='@east'
```

4.6.4. Libreswan을 사용하여 사이트 간 VPN 구성

Libreswan 에서 사이트 간 **IPsec VPN**을 만들기 위해 두 개의 호스트 사이에 **IPsec** 터널이 생성됩니다. 이 터널은 하나 이상의 서브넷에서 통과할 트래픽을 허용하도록 구성된 끝점입니다. 따라서 네트워크의 원격 부분에 대한 게이트웨이라고 간주할 수 있습니다. 사이트 간 **VPN** 구성은 하나 이상의 네트워크 또는 서브넷을 구성 파일에 지정해야 한다는 점에서 호스트 간 **VPN**과만 다릅니다.

사이트 간 **IPsec VPN**을 생성하도록 **Libreswan** 을 구성하려면 먼저 **4.6.3절. “Libreswan을 사용하여 호스트-호스트 VPN 만들기”** 에 설명된 대로 호스트 간 **IPsec VPN**을 구성한 다음 해당 파일을 적절한 이름으로 복사하거나 **/etc/ipsec.d/my_site-to-site.conf** 와 같은 파일로 복사합니다. **root** 로 실행되는 편집기를 사용하여 다음과 같이 사용자 지정 설정 파일 **/etc/ipsec.d/my_site-to-site.conf** 를 편집합니다.

```
conn mysubnet
    also=mytunnel
    leftsubnet=192.0.1.0/24
    rightsubnet=192.0.2.0/24
    auto=start

conn mysubnet6
    also=mytunnel
    connaddrfamily=ipv6
    leftsubnet=2001:db8:0:1::/64
    rightsubnet=2001:db8:0:2::/64
    auto=start

conn mytunnel
    leftid=@west.example.com
    left=192.1.2.23
```

```
    leftrsasigkey=0sAQOrlo+hOafUZDlCQmXFrje/oZm [...] W2n417C/4urYHQkCvuIQ==
    rightid=@east.example.com
    right=192.1.2.45
    rightrsasigkey=0sAQO3fwC6nSSGgt64DWiYZzuHbc4 [...] D/v8t5YTQ==
    authby=rsasig
```

*터널을 작동시키려면 **Libreswan** 을 다시 시작하거나 수동으로 로드하고 다음 명령을 사용하여 **root** 로 모든 연결을 시작합니다.*

```
~]# ipsec auto --add mysubnet
```

```
~]# ipsec auto --add mysubnet6
```

```
~]# ipsec auto --up mysubnet
104 "mysubnet" #1: STATE_MAIN_I1: initiate
003 "mysubnet" #1: received Vendor ID payload [Dead Peer Detection]
003 "mytunnel" #1: received Vendor ID payload [FRAGMENTATION]
106 "mysubnet" #1: STATE_MAIN_I2: sent MI2, expecting MR2
108 "mysubnet" #1: STATE_MAIN_I3: sent MI3, expecting MR3
003 "mysubnet" #1: received Vendor ID payload [CAN-IKEv2]
004 "mysubnet" #1: STATE_MAIN_I4: ISAKMP SA established {auth=OAKLEY_RSA_SIG
cipher=aes_128 prf=oakley_sha group=modp2048}
117 "mysubnet" #2: STATE_QUICK_I1: initiate
004 "mysubnet" #2: STATE_QUICK_I2: sent QI2, IPsec SA established tunnel mode
{ESP=>0x9414a615 <0x1a8eb4ef xfrm=AES_128-HMAC_SHA1 NATOA=none NATD=none
DPD=none}
```

```
~]# ipsec auto --up mysubnet6
003 "mytunnel" #1: received Vendor ID payload [FRAGMENTATION]
117 "mysubnet" #2: STATE_QUICK_I1: initiate
004 "mysubnet" #2: STATE_QUICK_I2: sent QI2, IPsec SA established tunnel mode
{ESP=>0x06fe2099 <0x75eaa862 xfrm=AES_128-HMAC_SHA1 NATOA=none NATD=none
DPD=none}
```

4.6.4.1. Libreswan을 사용하여 사이트 간 VPN 확인

VPN 터널을 통해 패킷이 전송되는지 확인하는 작업은 4.6.3.1절. “Libreswan을 사용하여 호스트-호스트 VPN 확인” 에 설명된 절차와 동일합니다.

4.6.5. Libreswan을 사용하여 사이트 간 단일 터널 VPN 구성

*종종 사이트 간 터널이 구축되면 게이트웨이는 공용 IP 주소 대신 내부 IP 주소를 사용하여 서로 통신해야 합니다. 이 작업은 단일 터널을 사용하여 수행할 수 있습니다. 호스트 이름이 **west** 인 왼쪽 호스트에 내부 IP 주소 **192.0.1.254** 가 있고 호스트 이름이 **east** 인 오른쪽 호스트가 있는 경우 단일 터널을 사용하여 두 서버의 **/etc/ipsec.d/myvpn.conf** 파일에 다음 구성을 저장하십시오.*

```
conn mysubnet
```

```
leftid=@west.example.com
leftrsasigkey=0sAQOrlo+hOafUZDlCQmXFrje/oZm [...] W2n417C/4urYHQkCvuIQ==
left=192.1.2.23
leftsourceip=192.0.1.254
leftsubnet=192.0.1.0/24
rightid=@east.example.com
rightrsasigkey=0sAQO3fwC6nSSGgt64DWiYZzuHbc4 [...] D/v8t5YTQ==
right=192.1.2.45
rightsourceip=192.0.2.254
rightsubnet=192.0.2.0/24
auto=start
authby=rsasig
```

4.6.6. Libreswan을 사용하여 서브넷 확장 구성

IPsec 은 종종 허브 및 사용자 지정 아키텍처에 배포됩니다. 각 리프 노드에는 더 큰 범위의 일부인 **IP** 범위가 있습니다. 잎은 허브를 통해 서로 통신합니다. 이를 서브넷 **UTrusion**이라고 합니다.

예 4.2. 간단한 서브넷 확장 설정 구성

다음 예제에서는 **10.0.0.0/8** 및 더 작은 **/24** 서브넷을 사용하는 두 개의 분기로 헤드 사무실을 구성합니다.

헤드 사무실에서:

```
conn branch1
    left=1.2.3.4
    leftid=@headoffice
    leftsubnet=0.0.0.0/0
    leftrsasigkey=0sA[...]
    #
    right=5.6.7.8
    rightid=@branch1
    rightsubnet=10.0.1.0/24
    rightrsasigkey=0sAXXXX[...]
    #
    auto=start
    authby=rsasig

conn branch2
    left=1.2.3.4
    leftid=@headoffice
    leftsubnet=0.0.0.0/0
    leftrsasigkey=0sA[...]
    #
    right=10.11.12.13
    rightid=@branch2
    rightsubnet=10.0.2.0/24
    rightrsasigkey=0sAYYYY[...]
```

```
    #
    auto=start
    authby=rsasig
```

"branch1" 사무실에서는 동일한 연결을 사용합니다. 또한 통과 연결을 사용하여 로컬 LAN 트래픽이 터널을 통해 전송되는 것을 제외합니다.

```
conn branch1
    left=1.2.3.4
    leftid=@headoffice
    leftsubnet=0.0.0.0/0
    leftrsasigkey=0sA[...]
    #
    right=10.11.12.13
    rightid=@branch2
    rightsubnet=10.0.1.0/24
    rightrsasigkey=0sAYYYY[...]
    #
    auto=start
    authby=rsasig

conn passthrough
    left=1.2.3.4
    right=0.0.0.0
    leftsubnet=10.0.1.0/24
    rightsubnet=10.0.1.0/24
    authby=never
    type=passthrough
    auto=route
```

4.6.7. IKEv2 원격 액세스 VPN Libreswan 구성

Horizon 전사에서는 랩탑과 같은 동적으로 할당된 IP 주소를 사용하는 모바일 클라이언트를 대상으로 사용자를 이동할 수 있습니다. 인증서는 인증서를 사용하여 인증됩니다. 이전 IKEv1 XAUTH 프로토콜을 사용하지 않도록 하려면 다음 예제에서 IKEv2를 사용합니다.

서버에서 다음을 수행합니다.

```
conn roadwarriors
    ikev2=insist
    # Support (roaming) MOBIKE clients (RFC 4555)
    mobike=yes
    fragmentation=yes
    left=1.2.3.4
    # if access to the LAN is given, enable this, otherwise use 0.0.0.0/0
    # leftsubnet=10.10.0.0/16
    leftsubnet=0.0.0.0/0
```

```
    leftcert=vpn-server.example.com
    leftid=%fromcert
    leftxauthserver=yes
    leftmodecfgserver=yes
    right=%any
    # trust our own Certificate Agency
    rightca=%same
    # pick an IP address pool to assign to remote users
    # 100.64.0.0/16 prevents RFC1918 clashes when remote users are behind NAT
    rightaddresspool=100.64.13.100-100.64.13.254
    # if you want remote clients to use some local DNS zones and servers
    modecfgdns="1.2.3.4, 5.6.7.8"
    modecfgdomains="internal.company.com, corp"
    rightxauthclient=yes
    rightmodecfgclient=yes
    authby=rsasig
    # optionally, run the client X.509 ID through pam to allow/deny client
    # pam-authorize=yes
    # load connection, don't initiate
    auto=add
    # kill vanished roadwarriors
    dpddelay=1m
    dpdtimeout=5m
    dpdaction=%clear
```

다음과 같습니다.

left=1.2.3.4

***1.2.3.4** 값은 서버의 실제 **IP** 주소 또는 호스트 이름을 지정합니다.*

leftcert=vpn-server.example.com

*이 옵션은 인증서를 가져오는 데 사용된 친숙한 이름 또는 닉네임을 참조하는 인증서를 지정합니다. 일반적으로 이름은 **.p12** 파일 형식으로 **PKCS #12** 인증서 번들의 일부로 생성됩니다. 자세한 내용은 **pkcs12(1)** 및 **pk12util(1)** 매뉴얼 페이지를 참조하십시오.*

모바일 클라이언트에서 도로 전사의 장치는 이전 설정의 약간의 변형을 사용합니다.

```
conn to-vpn-server
    ikev2=insist
    # pick up our dynamic IP
    left=%defaultroute
    leftsubnet=0.0.0.0/0
    leftcert=myname.example.com
    leftid=%fromcert
    leftmodecfgclient=yes
    # right can also be a DNS hostname
```

```
right=1.2.3.4
# if access to the remote LAN is required, enable this, otherwise use 0.0.0.0/0
# rightsubnet=10.10.0.0/16
rightsubnet=0.0.0.0/0
# trust our own Certificate Agency
rightca=%same
authby=rsasig
# allow narrowing to the server's suggested assigned IP and remote subnet
narrowing=yes
# Support (roaming) MOBIKE clients (RFC 4555)
mobike=yes
# Initiate connection
auto=start
```

다음과 같습니다.

auto=start

이 옵션을 사용하면 **ipsec** 시스템 서비스가 시작될 때마다 **VPN**에 연결할 수 있습니다. 나중에 연결을 설정하려는 경우 **auto=add** 로 바꿉니다.

4.6.8. X.509를 사용하여 IKEv1 원격 액세스 VPN Libreswan 및 XAUTH 구성

Libreswan 은 **XAUTH IPsec** 확장을 사용하여 연결을 설정하므로 **VPN** 클라이언트에 **IP** 주소 및 **DNS** 정보를 기본적으로 할당하는 방법을 제공합니다. 확장 인증**(XAUTH)**은 **PSK** 또는 **X.509** 인증서를 사용하여 배포할 수 있습니다. **X.509**를 사용하여 배포하는 것이 더 안전합니다. 클라이언트 인증서는 인증서 해지 목록 또는**OCSP(Online Certificate Status Protocol)**로 취소할 수 있습니다. **X.509** 인증서를 사용하면 개별 클라이언트는 서버를 가장할 수 없습니다. **PSK**를 사용하면 그룹 암호라고도 하며 이론적으로 가능합니다.

XAUTH에는 **VPN** 클라이언트가 사용자 이름과 암호로 자신을 추가로 식별해야 합니다. **Google Authenticator** 또는 **RSA SecureID** 토큰과 같은 **One Time Passwords(OTP)**의 경우 일회성 토큰이 사용자 암호에 추가됩니다.

XAUTH에는 다음 세 가지 백엔드가 있습니다.

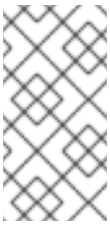
xauthby=pam

이렇게 하면 **/etc/pam.d/pluto** 의 설정을 사용하여 사용자를 인증합니다. **PAM(Pluggable Authentication Modules)**은 다양한 백엔드를 자체적으로 사용하도록 구성할 수 있습니다. 시스템 계정 사용자 암호 스키마, **LDAP** 디렉터리, **RADIUS** 서버 또는 사용자 지정 암호 인증 모듈을 사용할 수 있습니다. 자세한 내용은 **PAM(Pluggable Authentication Modules)** 사용 장을 참조하십시오.

xauthby=file

*이렇게 하면 **/etc/ipsec.d/passwd** 구성 파일을 사용합니다(**/etc/ipsec.d/nsspassword** 파일과 혼동해서는 안 됩니다). 이 파일의 형식은 **Apache .htpasswd** 파일과 유사하며 **Apache htpasswd** 명령을 사용하여 이 파일에 항목을 생성할 수 있습니다. 그러나 사용자 이름과 암호가 끝나면 예를 들어 **conn remoteusers** 를 사용하여 사용자를 제거하도록 **VPN**을 제공하는 경우 암호 파일 항목이 다음과 같이 표시됩니다.*

```
user1:$apr1$MIwQ3DHb$1l69LzTnZhnCT2DPQmAOK.:remoteusers
```



참고

*__htpasswd__ 명령을 사용하는 경우 각 행의 **user:password** 부분 뒤에 연결 이름을 수동으로 추가해야 합니다.*

xauthby=alwaysok

*서버는 항상 **XAUTH** 사용자와 암호 조합을 대체합니다. 서버에서 이러한 이름을 무시하더라도 클라이언트는 사용자 이름과 암호를 지정해야 합니다. 이는 사용자가 이미 **X.509** 인증서로 식별하거나 **XAUTH** 백엔드를 필요로 하지 않고 **VPN**을 테스트할 때만 사용해야 합니다.*

***X.509** 인증서를 사용한 서버 구성 예:*

```
conn xauth-rsa
   ikev2=never
   auto=add
   authby=rsasig
   pfs=no
   rekey=no
   left=ServerIP
   leftcert=vpn.example.com
   #leftid=%fromcert
   leftid=vpn.example.com
   leftsendcert=always
   leftsubnet=0.0.0.0/0
   rightaddresspool=10.234.123.2-10.234.123.254
   right=%any
   rightrsasigkey=%cert
   modecfgdns="1.2.3.4,8.8.8.8"
   modecfgdomains=example.com
   modecfgbanner="Authorized access is allowed"
   leftxauthserver=yes
   rightxauthclient=yes
   leftmodecfgserver=yes
   rightmodecfgclient=yes
   modecfgpull=yes
   xauthby=pam
   dpddelay=30
```

```
dpdtimeout=120
dpdaction=clear
ike_frag=yes
# for walled-garden on xauth failure
# xauthfail=soft
# leftupdown=/custom/_updown
```

xauthfail 를 소프트로 설정하면 하드 대신 인증 실패가 무시되고 사용자가 올바르게 인증된 것처럼 **VPN**이 설정됩니다. 사용자 지정 **updown** 스크립트는 환경 변수 **XAUTH_FAILED** 를 확인하는 데 사용할 수 있습니다. 그런 다음 이러한 사용자는 예를 들어 **iptables DNAT**를 사용하여 관리자에게 연락하거나 유료 서브스크립션을 갱신할 수 있는 월든 "으로" 리디렉션될 수 있습니다.

VPN 클라이언트는 **modecfgdomain** 값과 **DNS** 항목을 사용하여 지정된 도메인의 쿼리를 지정된 이름 서버로 리디렉션합니다. 이를 통해 사용자는 내부 **DNS** 이름을 사용하여 내부 전용 리소스에 액세스할 수 있습니다. **IKEv2**는 **modecfgdomains** 및 **modecfgdns** 를 사용하여 쉼표로 구분된 도메인 이름 및 이름 서버 **IP** 주소 목록을 지원하지만 **IKEv1** 프로토콜은 하나의 도메인 이름만 지원하며 **libreswan**은 두 개의 이름 서버 **IP** 주소만 지원합니다. 선택적으로 배너 텍스트를 **VPN cliens**에 보내려면 **modecfgbanner** 옵션을 사용하십시오.

leftsubnet 이 **0.0.0.0/0** 이 아닌 경우 분할 터널링 구성 요청이 자동으로 클라이언트로 전송됩니다. 예를 들어 **leftsubnet=10.0.0.0/8** 을 사용하는 경우 **VPN** 클라이언트는 **VPN**을 통해 **10.0.0.0/8** 에 대한 트래픽만 보냅니다.

클라이언트에서 사용자는 사용된 백엔드에 따라 사용자 암호를 입력해야 합니다. 예를 들어 다음과 같습니다.

xauthby=file

관리자가 암호를 생성하여 **/etc/ipsec.d/passwd** 파일에 저장했습니다.

xauthby=pam

암호는 **/etc/pam.d/pluto** 파일의 **PAM** 구성에 지정된 위치에서 가져옵니다.

xauthby=alwaysok

비밀번호가 확인되지 않고 항상 승인됩니다. 테스트 목적으로 이 옵션을 사용하거나 **xauth** 전용 클라이언트의 호환성을 보장하려면 이 옵션을 사용합니다.

추가 리소스

XAUTH에 대한 자세한 내용은 **ISAKMP/Oakley(XAUTH) Internet-Draft** 문서의 확장 인증을 참조하십시오.

4.6.9. Quantum Computers에 대한 보호 사용

IKEv1을 PreShared Keys와 함께 사용하여 앰프링 공격자에 대한 보호 기능이 제공되었습니다. IKEv2의 재 설계는 기본적으로 이러한 보호를 제공하지 않습니다. Libreswan 은 뮤지션 공격에 대해 IKEv2 연결을 보호하기 위해 Postquantum Preshared Keys (PPK)의 사용을 제공합니다.

선택적 PPK 지원을 활성화하려면 연결 정의에 ppk=yes 를 추가합니다. PPK를 요구하려면 ppk=insist 를 추가합니다. 그런 다음 각 클라이언트에는 대역 외로 전달되는 비밀 값이 있는 PPK ID를 제공할 수 있습니다(및 가급적 언더 안전한 경우). PPK는 랜덤스에서 매우 강력해야 하며 사전 단어를 기반으로 하지 않아야 합니다. PPK ID와 PPK 데이터 자체는 ipsec.secrets 에 저장됩니다. 예를 들면 다음과 같습니다.

```
@west @east : PPKS "user1" "thestringismeanttobearandomstr"
```

PPKS 옵션은 정적 PPK를 나타냅니다. 한 번에패드 기반 동적 PPK를 사용하는 실험 기능이 있습니다. 각각의 연결에서 1회장의 새로운 부분이 PPK로 사용됩니다. 사용할 경우 다시 사용하지 않도록 파일 내의 동적 PPK의 해당 부분을 0으로 덮어씁니다. 더 이상 타임스탬프가 남아 있지 않으면 연결이 실패합니다. 자세한 내용은 ipsec.secrets(5) 도움말 페이지를 참조하십시오.

주의

동적 PPK 구현은 기술 프리뷰로 제공되며 이 기능을 주의해서 사용해야 합니다. 자세한 내용은 Red Hat Enterprise Linux 7.5 릴리스 노트 를 참조하십시오.

4.6.10. 추가 리소스

다음 정보 소스는 Libreswan 및 ipsec 데몬과 관련된 추가 리소스를 제공합니다.

4.6.10.1. 설치된 문서

- IPsec (8) 매뉴얼 페이지 - ipsec 에 대한 명령 옵션에 대해 설명합니다.
- IPsec.conf(5) 도움말 페이지 - ipsec 구성에 대한 정보가 포함되어 있습니다.

- *IPsec.secrets(5) 도움말 페이지 - ipsec.secrets 파일의 형식에 대해 설명합니다.*

- *ipsec_auto(8) 도움말 페이지 - 자동 키 교환을 사용하여 설정된 Libreswan IPsec 연결을 조작하기 위한 자동 명령행 클라이언트를 사용하는 방법에 대해 설명합니다.*

- *ipsec_rsasigkey(8) 도움말 페이지 - RSA 서명 키를 생성하는 데 사용되는 툴에 대해 설명합니다.*

- */usr/share/doc/libreswan-version/*

4.6.10.2. 온라인 문서

https://libreswan.org

업스트림 프로젝트의 웹 사이트.

https://libreswan.org/wiki

Libreswan 프로젝트 위키입니다.

https://libreswan.org/man/

모든 Libreswan 매뉴얼 페이지.

NIST 특수 게시 800-77: IPsec VPN에 대한 가이드

IPsec을 기반으로 보안 서비스 구현 조직에 대한 실질적인 지침입니다.

4.7. OPENSSL 사용

OpenSSL 은 애플리케이션에 암호화 프로토콜을 제공하는 라이브러리입니다. openssl 명령줄 유틸리티를 사용하면 쉘의 암호화 함수를 사용할 수 있습니다. 여기에는 대화형 모드가 포함됩니다.

openssl 명령행 유틸리티에는 시스템에 설치된 openssl 의 버전에 대한 정보를 제공하는 여러 의사 명령이 있습니다. pseudo-commands list-standard-commands,list-message-digest-commands, list-

cipher-commands 는 모든 표준 명령, 메시지 다이제스트 명령 또는 암호화 명령의 목록을 각각 출력하고, 현재 **openssl** 유틸리티에서 사용할 수 있는 명령을 각각 출력합니다.

pseudo-commands list-cipher-algorithms 및 **list-message-digest-algorithms** 는 모든 암호 및 메시지 다이제스트 이름을 나열합니다. **pseudo-command list-public-key-algorithms** 에는 지원되는 모든 공개 키 알고리즘이 나열됩니다. 예를 들어 지원되는 공개 키 알고리즘을 나열하려면 다음 명령을 실행합니다.

```
~]$ openssl list-public-key-algorithms
```

pseudo-command no- command-name 은 지정된 이름의 명령 이름을 사용할 수 있는지 여부를 테스트합니다. 쉘 스크립트에서 사용하기 위한 용도입니다. 자세한 내용은 **man openssl(1)** 을 참조하십시오.

4.7.1. 암호화 키 생성 및 관리

OpenSSL 을 사용하면 해당 개인 키에서 공개 키가 파생됩니다. 따라서 알고리즘을 결정한 첫 번째 단계는 개인 키를 생성하는 것입니다. 이 예에서 개인 키는 **privkey.pem** 이라고 합니다. 예를 들어 기본 매개 변수를 사용하여 **RSA** 개인 키를 만들려면 다음 명령을 실행합니다.

```
~]$ openssl genpkey -algorithm RSA -out privkey.pem
```

RSA 알고리즘은 다음 옵션을 지원합니다.

- **rsa_keygen_bits:numbits -** 생성된 키의 비트 수입니다. 지정하지 않으면 **1024** 가 사용됩니다.
- **rsa_keygen_pubexp:value - RSA** 공용 지수 값. 큰 **10**진수 값이거나 앞에 **0x** 가 있는 경우 **16**진수 값일 수 있습니다. 기본값은 **65537** 입니다.

예를 들어 **3** 을 공용 지수로 사용하여 **2048**비트 **RSA** 개인 키를 만들려면 다음 명령을 실행합니다.

```
~]$ openssl genpkey -algorithm RSA -out privkey.pem -pkeyopt rsa_keygen_bits:2048 \ -pkeyopt
rsa_keygen_pubexp:3
```

128비트 **AES** 및 암호 "**hello**" 를 사용하여 출력하므로 개인 키를 암호화하려면 다음 명령을 실행합니다.

```
~]$ openssl genpkey -algorithm RSA -out privkey.pem -aes-128-cbc -pass pass:hello
```

개인 키 생성에 대한 자세한 내용은 **man genpkey(1)** 을 참조하십시오.

4.7.2. 인증서 생성

OpenSSL 을 사용하여 인증서를 생성하려면 개인 키를 사용할 수 있어야 합니다. 이 예에서 개인 키는 **privkey.pem** 이라고 합니다. 아직 개인 키를 생성하지 않은 경우 참조하십시오. **4.7.1절. "암호화 키 생성 및 관리"**

인증서를**CA(** 인증 기관 **)**에서 서명하려면 인증서를 생성한 다음 서명을 위해 **CA**로 보내야 합니다. 이를 인증서 서명 요청이라고 합니다. 자세한 내용은 **4.7.2.1절. "인증서 서명 요청 생성"**를 참조하십시오. 대체 방법은 자체 서명된 인증서를 만드는 것입니다. 자세한 내용은 **4.7.2.2절. "자체 서명된 인증서 만들기Create a self-signed certificate"**를 참조하십시오.

4.7.2.1. 인증서 서명 요청 생성

CA에 제출하기 위해 인증서를 생성하려면 다음 형식의 명령을 실행합니다.

```
~]$ openssl req -new -key privkey.pem -out cert.csr
```

이렇게 하면 기본 개인 정보 보호 전자 메일 **(PEM)** 형식으로 인코딩된 **cert.csr** 이라는 **X.509** 인증서가 생성됩니다. **PEM**이라는 이름은 **RFC 1424** "에 설명된 인터넷 전자 메일의 개인 정보 보호 기능 개선" 에서 파생됩니다. 대체 **DER** 형식으로 인증서 파일을 생성하려면 **-outform DER** 명령 옵션을 사용합니다.

위의 명령을 실행한 후에는 인증서의 고유 이름 **(DN)**을 생성하기 위해 귀하와 조직에 대한 정보를 입력하라는 메시지가 표시됩니다. 다음 정보가 필요합니다.

- 해당 국가의 두 가지 국가 코드
- 주 또는 지역의 전체 이름
- 도시 또는 타운
- 조직의 이름

- 조직 내 단위 이름
- 시스템의 이름 또는 호스트 이름
- 이메일 주소

req(1) 도움말 페이지는 **PKCS# 10** 인증서 요청 및 유틸리티 생성을 설명합니다. 인증서 생성 프로세스에 사용되는 기본 설정은 **/etc/pki/tls/openssl.cnf** 파일에 포함되어 있습니다. 자세한 내용은 **man openssl.cnf(5)** 를 참조하십시오.

4.7.2.2. 자체 서명된 인증서 만들기Create a self-signed certificate

366 일 동안 유효한 자체 서명 인증서를 생성하려면 다음 형식의 명령을 실행합니다.

```
~]$ openssl req -new -x509 -key privkey.pem -out selfcert.pem -days 366
```

4.7.2.3. Makefile을 사용하여 인증서 생성

/etc/pki/tls/certs/ 디렉터리에는 **make** 명령을 사용하여 인증서를 생성하는 데 사용할 수 있는 **Makefile** 이 포함되어 있습니다. 사용법 지침을 보려면 다음과 같이 명령을 실행합니다.

```
~]$ make -f /etc/pki/tls/certs/Makefile
```

또는 디렉터리로 변경하고 다음과 같이 **make** 명령을 실행합니다.

```
~]$ cd /etc/pki/tls/certs/
~]$ make
```

자세한 내용은 **make(1)** 도움말 페이지를 참조하십시오.

4.7.3. 인증서 확인

CA에서 서명한 인증서를 신뢰할 수 있는 인증서라고 합니다. 따라서 자체 서명된 인증서는 신뢰할 수 없는 인증서입니다. **verify** 유틸리티는 **SSL** 및 **S/MIME** 함수를 사용하여 표준 작업에서 **OpenSSL** 이 사용하는 것과 동일한 인증서를 확인합니다. 오류가 발견되면 보고되고 다른 오류를 보고하기 위해 테스트를 계속하려고 시도합니다.

PEM 형식으로 여러 개의 개별 X.509 인증서를 확인하려면 다음 형식으로 명령을 실행합니다.

```
~]$ openssl verify cert1.pem cert2.pem
```

인증서 체인을 확인하려면 리프 인증서가 cert.pem 에 있어야 하며 신뢰하지 않는 중간 인증서는 untrusted.pem 에서 직접 연결되어 있어야 합니다. 신뢰할 수 있는 루트 CA 인증서는 /etc/pki/tls/certs/ca-bundle.crt 또는 cacert.pem 파일에 나열된 기본 CA 중 하나여야 합니다. 그런 다음 체인을 확인하려면 다음 형식으로 명령을 실행합니다.

```
~]$ openssl verify -untrusted untrusted.pem -CAfile cacert.pem cert.pem
```

자세한 내용은 man verify(1) 을 참조하십시오.

중요

이 알고리즘의 부족으로 인해 Red Hat Enterprise Linux 7에서는 MD5 해시 알고리즘을 사용한 서명 확인이 비활성화됩니다. 항상 SHA256과 같은 강력한 알고리즘을 사용하십시오.

4.7.4. 파일 암호화 및 암호 해독

OpenSSL 로 파일을 암호화(및 암호 해독)하기 위해 pkeyutl 또는 enc 내장 명령을 사용할 수 있습니다. pkeyutl 을 사용하면RSA 키는 암호화 및 암호 해독을 수행하는 데 사용되지만 enc, 대칭 알고리즘으로 사용됩니다.

RSA 키 사용

일반 텍스트 라는 파일을 암호화하려면 다음과 같이 명령을 실행합니다.

```
~]$ openssl pkeyutl -in plaintext -out cyphertext -inkey privkey.pem
```

키와 인증서의 기본 형식은 PEM입니다. 필요한 경우 -keyform DER 옵션을 사용하여 DER 키 형식을 지정합니다.

암호화 엔진을 지정하려면 다음과 같이 -engine 옵션을 사용합니다.

```
~]$ openssl pkeyutl -in plaintext -out cyphertext -inkey privkey.pem -engine id
```

여기서 **id** 는 암호화 엔진의 **ID**입니다. 엔진의 가용성을 확인하려면 다음 명령을 실행합니다.

```
~]$ openssl engine -t
```

일반 텍스트 라는 데이터 파일에 서명하려면 다음과 같이 명령을 실행합니다.

```
~]$ openssl pkeyutl -sign -in plaintext -out sigtext -inkey privkey.pem
```

서명된 데이터 파일을 확인하고 데이터를 추출하려면 다음과 같이 명령을 실행합니다.

```
~]$ openssl pkeyutl -verifyrecover -in sig -inkey key.pem
```

예를 들어 **DSA** 키를 사용하여 서명을 확인하려면 다음과 같이 명령을 실행합니다.

```
~]$ openssl pkeyutl -verify -in file -sigfile sig -inkey key.pem
```

pkeyutl(1) 도움말 페이지에서는 공개 키 알고리즘 유틸리티를 설명합니다.

Symmetric Algorithms 사용

사용 가능한 대칭 암호화 알고리즘을 나열하려면 **-l** 과 같은 지원되지 않는 옵션으로 **enc** 명령을 실행합니다.

```
~]$ openssl enc -l
```

알고리즘을 지정하려면 해당 이름을 옵션으로 사용합니다. 예를 들어 **aes-128-cbc** 알고리즘을 사용하려면 다음 구문을 사용합니다.

```
openssl enc -aes-128-cbc
```

aes-128-cbc 알고리즘을 사용하여 일반 텍스트 라는 파일을 암호화하려면 다음 명령을 입력합니다.

```
~]$ openssl enc -aes-128-cbc -in plaintext -out plaintext.aes-128-cbc
```

이전 예제에서 얻은 파일의 암호를 해독하려면 다음 예제와 같이 **-d** 옵션을 사용합니다.

```
~]$ openssl enc -aes-128-cbc -d -in plaintext.aes-128-cbc -out plaintext
```

중요

enc 명령은 **AEAD** 암호를 올바르게 지원하지 않으며 **ecb** 모드는 안전한 것으로 간주되지 않습니다. 최상의 결과를 얻으려면 **cbc,cfb,ofb** 또는 **ctr** 이외의 다른 모드를 사용하지 마십시오.

4.7.5. 메시지 다이제스트 생성

dgst 명령은 제공된 파일 또는 파일의 **16**진수로 메시지 다이제스트를 생성합니다. 이 명령은 디지털 서명 및 확인에도 사용할 수 있습니다. 메시지 다이제스트 명령은 다음 형식을 사용합니다.

```
openssl dgst algorithm -out filename -sign private-key
```

여기서 알고리즘은 **md5|md4|md2|sha|mdc2|ripemd160|dss1** 중 하나입니다. 글을 쓰는 시점에서 **SHA1** 알고리즘이 선호됩니다. **DSA**를 사용하여 서명하거나 확인해야 하는 경우 **dss1** 옵션을 **-rand** 옵션으로 지정한 임의의 데이터가 포함된 파일과 함께 사용해야 합니다.

sha1 알고리즘을 사용하여 기본 **Hex** 형식으로 메시지 다이제스트를 생성하려면 다음 명령을 실행합니다.

```
~]$ openssl dgst sha1 -out digest-file
```

개인 키 **privekey.pem** 을 사용하여 다이제스트를 디지털 서명하려면 다음 명령을 실행합니다.

```
~]$ openssl dgst sha1 -out digest-file -sign privkey.pem
```

자세한 내용은 **man dgst(1)** 을 참조하십시오.

4.7.6. 암호 해시 생성

passwd 명령은 암호의 해시를 계산합니다. 명령줄에서 암호의 해시를 계산하려면 다음과 같이 명령을 실행합니다.

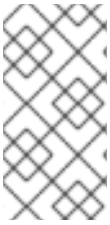
```
~]$ openssl passwd password
```

기본적으로 **-crypt** 알고리즘이 사용됩니다.

MD5 기반 **BSD** 알고리즘 **1** 을 사용하여 표준 입력에서 암호의 해시를 계산하려면 다음과 같이 명령을 실행합니다.

```
~]$ openssl passwd -1 password
```

apr1 옵션은 **BSD** 알고리즘의 **Apache** 변형을 지정합니다.



참고

FIPS 모드가 비활성화된 경우에만 **openssl passwd -1 password** 명령을 사용하십시오. 그렇지 않으면 명령이 작동하지 않습니다.

파일에 저장된 암호의 해시를 계산하고 **Salt xx** 를 사용하여 다음과 같이 명령을 실행합니다.

```
~]$ openssl passwd -salt xx -in password-file
```

암호는 표준 출력으로 전송되며 출력 파일을 지정하는 **-out** 옵션이 없습니다. 표는 해당 일반 텍스트 암호를 사용하여 암호 해시 테이블을 생성합니다.

자세한 내용 및 예제는 **man sslpasswd(1)** 을 참조하십시오.

4.7.7. Random 데이터 생성

임의의 데이터가 포함된 파일을 생성하려면 초기 파일을 사용하여 다음 명령을 실행합니다.

```
~]$ openssl rand -out rand-file -rand seed-file
```

임의의 데이터 프로세스를 확인하기 위해 여러 파일을 목록 구분 기호로 지정할 수 있습니다.

자세한 내용은 **man rand(1)** 을 참조하십시오.

4.7.8. 벤치마킹 your system

지정된 알고리즘에 대해 시스템의 컴퓨팅 속도를 테스트하려면 다음 형식으로 명령을 실행합니다.

```
~]$ openssl speed algorithm
```

여기서 알고리즘은 사용하려는 지원되는 알고리즘 중 하나입니다. 사용 가능한 알고리즘을 나열하려면 **openssl** 속도를 입력하고 탭을 누릅니다.

4.7.9. OpenSSL 구성

OpenSSL에는 **/etc/pki/tls/openssl.cnf** 설정 파일이 있으며, 이를 **OpenSSL** 라이브러리에서 읽는 마스터 구성 파일이라고 합니다. 각 애플리케이션에 대해 개별 구성 파일이 있을 수도 있습니다. 구성 파일에는 섹션 이름이 있는 여러 섹션이 포함되어 있습니다. **[section_name]**. 첫 번째 **[section_name]** 까지 파일의 첫 번째 부분을 기본 섹션이라고 합니다. **OpenSSL**이 설정 파일에서 이름을 검색하면 이름이 지정된 섹션이 먼저 검색됩니다. 명령에 옵션을 사용하여 대체 설정 파일을 지정하지 않는 한 모든 **OpenSSL** 명령은 마스터 **OpenSSL** 구성 파일을 사용합니다. 구성 파일은 **config(5)** 도움말 페이지에 자세히 설명되어 있습니다.

RFC 2는 인증서 파일의 내용을 설명합니다. 이는 다음과 같습니다.

- **Internet X.509** 공개 키 인프라 인증서 및 인증서 해지 목록(CRL) 프로필
- **Internet X.509** 공개 키 인프라 인증서 및 인증서 해지 목록(CRL) 프로필 업데이트

4.8. STUNNEL 사용

stunnel 프로그램은 클라이언트와 서버 간의 암호화 래퍼입니다. 구성 파일에 지정된 포트에서 수신 대기하고, 클라이언트와의 통신 장치를 암호화하며, 일반 포트에서 수신 대기 중인 원래 데몬으로 데이터를 전달합니다. 이렇게 하면 암호화 유형을 지원하지 않는 서비스를 보호하거나 **POODLE SSL** 취약점 **(CVE-2014-3566)**의 영향을 받는 **SSL** 버전 **2** 및 **3**과 같은 보안상의 이유로 방지하려는 서비스 보안을 향상시킬 수 있습니다. 자세한 내용은 **https://access.redhat.com/solutions/1234773**을 참조하십시오. **CUPS** 는 자체 구성에서 **SSL**을 비활성화하는 방법을 제공하지 않는 구성 요소의 예입니다.

4.8.1. stunnel 설치

root 로 다음 명령을 입력하여 **stunnel** 패키지를 설치합니다.

```
~]# yum install stunnel
```

4.8.2. stunnel을 TLS Wrapper로 구성

stunnel 을 구성하려면 다음 단계를 수행합니다.

1. 사용하는 서비스에 관계없이 **stunnel** 에 유효한 인증서가 필요합니다. 적절한 인증서가 없는 경우 인증 기관에 적용하여 인증서를 얻거나 자체 서명 인증서를 생성할 수 있습니다.

주의

항상 프로덕션 환경에서 실행되는 서버에 인증 기관에서 서명한 인증서를 사용합니다. 자체 서명된 인증서는 테스트 목적 또는 사설 네트워크에만 적합합니다.

인증 기관에서 부여한 인증서에 대한 자세한 내용은 4.7.2.1절. “인증서 서명 요청 생성” 을 참조하십시오. 반면 **stunnel** 용으로 자체 서명된 인증서를 생성하려면 **/etc/pki/tls/certs/** 디렉터리를 입력하고 다음 명령을 **root** 로 입력합니다.

```
certs]# make stunnel.pem
```

이 과정을 완료하기 위해 모든 질문에 대답하십시오.

2. 인증서가 있는 경우 **stunnel** 에 대한 구성 파일을 생성합니다. 이 파일은 모든 행이 옵션 또는 서비스 정의의 시작을 지정하는 텍스트 파일입니다. 주석과 빈 줄을 파일에 보관하여 타당성을 개선할 수 있습니다. 여기서 주석이 **Semic**로 시작하는 경우도 있습니다.

stunnel RPM 패키지에는 구성 파일을 저장할 수 있는 **/etc/stunnel/** 디렉터리가 포함되어 있습니다. **stunnel** 에는 파일 이름 또는 해당 확장자의 특수 형식이 필요하지 않지만 **/etc/stunnel/stunnel.conf** 를 사용합니다. 다음 콘텐츠는 **stunnel** 을 **TLS** 래퍼로 구성합니다.

```
cert = /etc/pki/tls/certs/stunnel.pem
; Allow only TLS, thus avoiding SSL
sslVersion = TLSv1
chroot = /var/run/stunnel
```

```
setuid = nobody
setgid = nobody
pid = /stunnel.pid
socket = l:TCP_NODELAY=1
socket = r:TCP_NODELAY=1

[service_name]
accept = port
connect = port
TIMEOUTclose = 0
```

*또는 **sslVersion = TLSv1** 이 포함된 행을 다음 행으로 교체하여 **SSL**을 방지할 수 있습니다.*

```
options = NO_SSLv2
options = NO_SSLv3
```

옵션의 목적은 다음과 같습니다.

- ***cert -** 인증서 경로*
- ***sslVersion - SSL** 버전. **SSL**과 **TLS** 가 두 개의 독립적인 암호화 프로토콜 경우에도 여기에서 **TLS**를 사용할 수 있습니다.*
- ***chroot -** 더 높은 보안을 위해 **stunnel** 프로세스가 실행되는 변경된 루트 디렉터리*
- ***setuid, setgid - stunnel** 프로세스가 실행되는 사용자 및 그룹.*
- ***pid - stunnel** 이 있는 파일은 **chroot**를 기준으로 프로세스 **ID**를 저장합니다.*
- ***socket -** 로컬 및 원격 소켓 옵션; 이 경우 네트워크 대기 시간을 개선하기 위해 **Nagle** 의 알고리즘 을 비활성화합니다.*
- ***[service_name] -** 서비스 정의 시작; 이 줄 아래에 사용된 옵션은 지정된 서비스에만 적용되는 반면 위의 옵션은 **S tunnel** 에 전역적으로 적용됩니다.*
- ***accept -** 수신 대기할 포트*

- ***connect* - 연결할 포트입니다. 보안 서비스에 사용할 포트여야 합니다.**

- ***TIMEOUTclose* - 클라이언트에서 *close_notify* 경고를 대기하는 시간(초)입니다. *0* 은 *stunnel* 이 전혀 기다리지 않도록 지시합니다.**

- ***options* - *OpenSSL* 라이브러리 옵션**

> **예 4.3. *CUPS* 보안**
>
> ***CUPS* 에 대한 *TLS* 래퍼로 *stunnel*을 구성하려면 다음 값을 사용합니다.**
>
> ```
> [cups]
> accept = 632
> connect = 631
> ```
>
> ***632* 대신 원하는 모든 무료 포트를 사용할 수 있습니다. *631* 은 *CUPS* 에서 일반적으로 사용하는 포트입니다.**

3. ***chroot* 디렉터리를 만들고 *setuid* 옵션에서 지정한 사용자에게 쓰기 액세스 권한을 부여합니다. 이렇게 하려면 *root* 로 다음 명령을 입력합니다.**

```
~]# mkdir /var/run/stunnel
~]# chown nobody:nobody /var/run/stunnel
```

그러면 *stunnel* 이 *PID* 파일을 생성할 수 있습니다.

4. ***시스템에서 새 포트에 대한 액세스를 허용하지 않는 방화벽 설정을 사용하는 경우 적절하게 변경합니다. 자세한 내용은 *5.6.7절. “GUI를 사용하여 포트 열기”* 를 참조하십시오.***

5. ***구성 파일과 *chroot* 디렉토리를 생성하고 지정된 포트에 액세스할 수 있는지 확인하는 경우 *stunnel* 을 사용할 수 있습니다.***

4.8.3. stunnel 시작, 중지 및 재시작

stunnel 을 시작하려면 **root** 로 다음 명령을 입력합니다.

```
~]# stunnel /etc/stunnel/stunnel.conf
```

기본적으로 **stunnel** 은 **/var/log/secure** 를 사용하여 출력을 기록합니다.

stunnel 을 종료하려면 다음 명령을 **root** 로 실행하여 프로세스를 종료합니다.

```
~]# kill `cat /var/run/stunnel/stunnel.pid`
```

stunnel 이 실행 중인 동안 구성 파일을 편집하는 경우 **stunnel** 을 종료하고 변경 사항이 적용되도록 다시 시작합니다.

4.9. 암호화

4.9.1. LUKS 디스크 암호화 사용

Linux Unified Key Setup-on-disk-format (또는 LUKS)을 사용하면 **Linux** 컴퓨터의 파티션을 암호화할 수 있습니다. 이는 모바일 컴퓨터 및 이동식 미디어에 대해 특히 중요합니다. **LUKS**는 여러 사용자 키를 사용하여 파티션의 대량 암호화에 사용되는 마스터 키를 해독할 수 있습니다.

LUKS 개요

LUKS의 기능

- **LUKS**는 전체 블록 장치를 암호화하므로 이동식 스토리지 미디어 또는 랩탑 디스크 드라이브와 같은 모바일 장치의 콘텐츠를 보호하기에 적합합니다.
- 암호화된 블록 장치의 기본 내용은 임의적입니다. 이렇게 하면 스왑 장치를 암호화하는 데 유용합니다. 이는 데이터 저장을 위해 특별히 포맷된 블록 장치를 사용하는 특정 데이터베이스에서도 유용할 수 있습니다.
- **LUKS**는 기존 장치 매퍼 커널 하위 시스템을 사용합니다.
- **LUKS**는 사전 공격으로부터 보호하는 암호 강화를 제공합니다.
-

LUKS 장치에는 여러 개의 키 슬롯이 포함되어 있어 사용자가 백업 키 또는 암호를 추가할 수 있습니다.

LUKS에서 수행할 수 없는 작업:

- *LUKS는 많은(8 이상) 사용자가 동일한 장치에 고유한 액세스 키를 보유해야 하는 시나리오에 적합하지 않습니다.*
- *LUKS는 파일 수준 암호화가 필요한 애플리케이션에 적합하지 않습니다.*

중요

LUKS와 같은 디스크 암호화 솔루션은 시스템이 꺼져 있을 때만 데이터를 보호합니다. 시스템이 있고 LUKS가 디스크의 암호를 해독하면 해당 디스크의 파일은 일반적으로 액세스할 수 있는 모든 사용자가 사용할 수 있습니다.

4.9.1.1. Red Hat Enterprise Linux의 LUKS 구현

Red Hat Enterprise Linux 7은 LUKS를 사용하여 파일 시스템 암호화를 수행합니다. 기본적으로 파일 시스템을 암호화하는 옵션은 설치 중에 확인되지 않습니다. 하드 드라이브를 암호화하기 위한 옵션을 선택하면 컴퓨터를 부팅할 때마다 암호를 입력하라는 메시지가 표시됩니다. 이 암호는 파티션 암호를 해독하는 데 사용되는 대량 암호화 키를 "unlocks"합니다. 기본 파티션 테이블을 수정하려면 암호화할 파티션을 선택할 수 있습니다. 이는 파티션 테이블 설정에서 설정됩니다.

LUKS에 사용되는 기본 암호(cryptsetup --help참조)는 aes-cbc-essiv:sha256(ESSIV - Encrypted Salt-Sector Initialization Vector)입니다. 설치 프로그램 Anaconda 는 기본 XTS 모드(aes-xts-plain64)를 사용합니다. LUKS의 기본 키 크기는 256비트입니다. Anaconda (XTS 모드)가 있는 LUKS의 기본 키 크기는 512비트입니다. 사용 가능한 암호는 다음과 같습니다.

- *AES - Advanced Encryption Standard - FIPS PUB 197*
- *Twofish (128비트 블록 암호)*

- ***Serpent***
- ***cast5 - RFC 2144***
- ***cast6 - RFC 2612***

4.9.1.2. 수동으로 디렉터리 암호화

주의

다음 절차에 따라 암호화할 파티션의 모든 데이터가 제거됩니다. 당신은 모든 정보를 손실 할 수 있습니다! 이 절차를 시작하기 전에 외부 소스에 데이터를 백업하십시오!

1. *쉘 프롬프트에서 **root**로 다음을 입력하여 실행 수준 **1**을 입력합니다.*

```
telinit 1
```

2. *기존 **/home** 을 마운트 해제합니다.*

```
umount /home
```

3. *이전 단계의 명령이 실패하면 **fuser** 를 사용하여 **/home** 프로세스 **hogging /home** 을 찾아 종료합니다.*

```
fuser -mvk /home
```

4. ***/home** 이 더 이상 마운트되지 않았는지 확인합니다.*

```
grep home /proc/mounts
```

-

5. *파티션을 임의의 데이터로 채웁니다.*

```
shred -v --iterations=1 /dev/VG00/LV_home
```

이 명령은 장치의 순차적 쓰기 속도로 진행되며 완료하는 데 시간이 걸릴 수 있습니다. 암호화되지 않은 데이터가 사용 된 장치에 남아 있지 않도록하고 임의의 데이터 대신 암호화된 데이터를 포함하는 장치의 부분을 난독화하는 것이 중요합니다.

6. *파티션을 초기화합니다.*

```
cryptsetup --verbose --verify-passphrase luksFormat /dev/VG00/LV_home
```

7. *새로 암호화된 장치를 엽니다.*

```
cryptsetup luksOpen /dev/VG00/LV_home home
```

8. *장치가 있는지 확인합니다.*

```
ls -l /dev/mapper | grep home
```

9. *파일 시스템을 생성합니다.*

```
mkfs.ext3 /dev/mapper/home
```

10. *파일 시스템을 마운트합니다.*

```
mount /dev/mapper/home /home
```

11. *파일 시스템이 표시되는지 확인합니다.*

```
df -h | grep home
```

12. ***/etc/crypttab** 파일에 다음을 추가합니다.*

```
home /dev/VG00/LV_home none
```

13. */etc/fstab* 파일을 편집하여 */home* 에 대한 이전 항목을 제거하고 다음 행을 추가합니다.

```
/dev/mapper/home /home ext3 defaults 1 2
```

14. 기본 *SELinux* 보안 컨텍스트를 복원합니다.

```
/sbin/restorecon -v -R /home
```

15. 머신을 재부팅합니다.

```
shutdown -r now
```

16. */etc/crypttab* 의 항목을 통해 컴퓨터에서 부팅 시 *luks* 암호를 요청합니다.

17. *root*로 로그인하여 백업을 복원합니다.

이제 컴퓨터가 꺼지면 모든 데이터가 안전하게 정지될 수 있는 암호화된 파티션이 있습니다.

4.9.1.3. 기존 장치에 새 암호 추가

다음 명령을 사용하여 기존 장치에 새 암호를 추가합니다.

```
cryptsetup luksAddKey device
```

인증을 위해 기존 *passprases* 중 하나를 입력하라는 메시지가 표시되면 새 암호를 입력하라는 메시지가 표시됩니다.

4.9.1.4. 기존 장치에서 암호 제거

다음 명령을 사용하여 기존 장치에서 암호를 제거합니다.

```
cryptsetup luksRemoveKey device
```

제거하려는 암호를 입력하라는 메시지가 표시되고 인증을 위해 나머지 암호 중 하나에 대해 메시지가 표시됩니다.

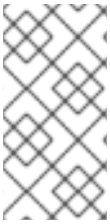
4.9.1.5. Anaconda에서 암호화된 블록 장치 생성

시스템을 설치하는 동안 암호화된 장치를 만들 수 있습니다. 이를 통해 암호화된 파티션으로 시스템을 쉽게 구성할 수 있습니다.

블록 장치 암호화를 사용하도록 설정하려면 개별 파티션, 소프트웨어 **RAID** 배열 또는 논리 볼륨을 만들 때 자동 파티션 또는 **Encrypt** 확인란을 선택할 때 암호화 확인란을 선택합니다. 파티션을 완료하면 암호화 암호를 입력하라는 메시지가 표시됩니다. 이 암호는 암호화된 장치에 액세스하는 데 필요합니다. 기존 **LUKS** 장치가 있고 설치 프로세스 초기에 올바른 암호를 제공한 경우 암호 항목 대화 상자에는 확인란도 포함됩니다. 이 확인란을 선택하면 기존에서 암호화된 각 블록 장치에서 사용 가능한 슬롯에 새 암호를 추가할 수 있음을 나타냅니다.

참고

자동 파티셔닝 화면에서 암호화 시스템 확인란을 선택한 다음 사용자 지정 레이아웃 만들기를 선택하면 블록 장치가 자동으로 암호화되지 않습니다.



참고

Kickstart를 사용하여 새로 암호화된 각 블록 장치에 대해 별도의 암호를 설정할 수 있습니다.

4.9.1.6. 추가 리소스

Red Hat Enterprise Linux 7의 **LUKS** 또는 암호화 하드 드라이브에 대한 자세한 내용은 다음 링크 중 하나를 참조하십시오.

- **LUKS** 홈 페이지
- **LUKS/cryptsetup FAQ**
- **LUKS - Linux** 통합 키 설정 **Wikipedia** 문서

- *HOWTO: 두 번째 하드 드라이브 및 pvmove를 사용하여 암호화된 PV(물리 볼륨) 생성*

4.9.2. GPG 키 생성

GPG는 모르는 사람을 포함하여 자신을 식별하고 통신을 인증하는 데 사용됩니다. GPG를 사용하면 GPG 서명 이메일을 읽는 사람이 진위 여부를 확인할 수 있습니다. 즉, GPG를 사용하면 실제로 사용자가 서명한 통신을 합리적으로 확신할 수 있습니다. GPG는 타사가 코드를 변경하거나 대화를 가로채고 메시지를 변경하는 것을 방지하기 때문에 유용합니다.

4.9.2.1. GNOME에서 GPG 키 생성

GNOME 에서 GPG 키를 생성하려면 다음 단계를 따르십시오.

1. Seahorse 유틸리티를 설치하면 GPG 키 관리가 더 쉬워집니다.

```
~]# yum install seahorse
```

2. 키를 생성하려면 Applications → Accessories 메뉴에서 Passwords and Encryption Keys (암호 및 암호화 키)를 선택합니다.

3. 파일 메뉴에서 새로 만들기를 선택한 다음 wget 키를 선택합니다. 그런 다음 Continue 를 클릭합니다.

4. 전체 이름, 이메일 주소 및 사용자를 설명하는 선택적 의견을 입력합니다. (예: John C. Smith, jsmith@example.com, Software Engineer). 생성을 클릭합니다. 키에 암호를 요청하는 대화 상자가 표시됩니다. 강력한 암호를 선택하지만 쉽게 기억할 수 있습니다. 확인 을 클릭하면 키가 생성됩니다.Click OK and the key is created.

주의

암호를 잊어버린 경우 데이터를 해독할 수 없습니다.

GPG 키 ID를 찾으려면 새로 생성된 키 옆에 있는 키 ID 열을 찾습니다. 대부분의 경우 키 ID를 요청하

는 경우 0x 6789ABCD 와 같이 키 ID 앞에 0x를 추가합니다. 개인 키의 백업을 만들어 안전한 곳에 저장해야 합니다.

4.9.2.2. nfsnobody에서 GPG 키 생성

nfsnobody에 GPG 키를 만들려면 다음 단계를 따르십시오.

1. *메인 메뉴에서 KGpg 프로그램을 시작합니다. Applications → EncryptionTool. 이전에 KGpg 를 사용한 적이 없는 경우 프로그램은 자체 GPG 키 쌍을 생성하는 프로세스를 안내합니다.*

2. *새 키 쌍을 만들라는 대화 상자가 나타납니다. 이름, 이메일 주소 및 선택적 의견을 입력합니다. 키에 대한 만료 시간 및 주요 강점(비트 수) 및 알고리즘도 선택할 수 있습니다.*

3. *다음 대화 상자에 암호를 입력합니다. 이 시점에서 키가 기본 KGpg 창에 나타납니다.*

주의

암호를 잊어버린 경우 데이터를 해독할 수 없습니다.

GPG 키 ID를 찾으려면 새로 생성된 키 옆에 있는 키 ID 열을 찾습니다. 대부분의 경우 키 ID를 요청하는 경우 0x 6789ABCD 와 같이 키 ID 앞에 0x를 추가합니다. 개인 키의 백업을 만들어 안전한 곳에 저장해야 합니다.

4.9.2.3. 명령줄을 사용하여 GPG 키 생성

1. *다음 쉘 명령을 사용합니다.*

```
~]$ gpg2 --gen-key
```

이 명령은 공개 키와 개인 키로 구성된 키 쌍을 생성합니다. 다른 사용자는 공개 키를 사용하여 통신을 인증하고 암호 해독합니다. 가능한 한 널리 공개 키를 배포하며 특히 메일링 리스트와 같은 실제 통신을 받고자 하는 사람에게 공개 키를 배포합니다.

2.

*일련의 프롬프트가 프로세스를 통해 안내합니다. 필요한 경우 **Enter** 키를 눌러 기본값을 할당합니다. 첫 번째 프롬프트에서는 원하는 키 유형을 선택할 수 있습니다.*

```
Please select what kind of key you want:
(1) RSA and RSA (default)
(2) DSA and Elgamal
(3) DSA (sign only)
(4) RSA (sign only)
Your selection?
```

*거의 모든 경우에 기본값은 올바른 옵션입니다. **RSA/RSA** 키를 사용하면 통신 서명뿐만 아니라 파일을 암호화할 수도 있습니다.*

3.

키 크기를 선택합니다.

```
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
```

*다시 한번 기본 **2048**는 거의 모든 사용자에게 충분하며 매우 강력한 보안 수준을 나타냅니다.*

4.

키가 만료될 시기를 선택합니다. 기본값이 아닌 만료 날짜를 선택하는 것이 좋습니다 . 예를 들어 키의 이메일 주소가 유효하지 않으면 만료 날짜가 다른 사용자에게 해당 공개 키 사용을 중지하도록 알립니다.

```
Please specify how long the key should be valid.
0 = key does not expire
d = key expires in n days
w = key expires in n weeks
m = key expires in n months
y = key expires in n years
key is valid for? (0)
```

*예를 들어 **1y** 값을 입력하면 키가 **1** 년 동안 유효합니다. (키가 생성된 후, 사고를 변경하는 경우 이 만료 날짜를 변경할 수 있습니다.)*

5.

gpg2 애플리케이션이 서명 정보를 요청하기 전에 다음 프롬프트가 표시됩니다.

```
Is this correct (y/N)?
```

y 를 입력하여 프로세스를 완료합니다.

6.

GPG 키의 이름 및 이메일 주소를 입력합니다. 이 프로세스는 실제 개인으로 사용자를 인증하는 것입니다. 따라서 실제 이름을 포함합니다. **bogus** 이메일 주소를 선택하면 다른 사용자가 공개 키를 찾기가 더 어려워집니다. 이렇게 하면 통신을 인증하기가 어려워집니다. 예를 들어 메일링 목록에 자체 소개하는 데 이 **GPG** 키를 사용하는 경우 해당 목록에 사용하는 이메일 주소를 입력합니다.

comment 필드를 사용하여 별칭 또는 기타 정보를 포함합니다. (일부 사용자는 다른 용도로 다른 키를 사용하고 각 키를 **"Office"** 또는 **"Open Source Projects"**)와 같은 주석으로 각 키를 식별합니다.

7.

확인 프롬프트에서 문자 **O** 를 입력하여 모든 항목이 올바르거나 다른 옵션을 사용하여 문제를 해결합니다. 마지막으로 비밀 키의 암호를 입력합니다. **gpg2** 프로그램은 입력 오류가 발생하지 않도록 암호를 두 번 입력하도록 요청합니다.

8.

마지막으로, **gpg2** 는 가능한 한 고유하게 키를 만들기 위해 임의의 데이터를 생성합니다. 마우스를 이동하고 임의의 키를 입력하거나 이 단계에서 시스템에서 다른 작업을 수행하여 프로세스 속도를 높입니다. 이 단계가 완료되면 키가 완료되고 사용할 준비가 완료됩니다.

```
pub  1024D/1B2AFA1C 2005-03-31 John Q. Doe <jqdoe@example.com>
Key fingerprint = 117C FE83 22EA B843 3E86  6486 4320 545E 1B2A FA1C
sub  1024g/CEA4B22E 2005-03-31 [expires: 2006-03-31]
```

9.

키 지문은 키에 대한 간단한 "사용자 정의"입니다. 이를 통해 다른 사용자가 변조 없이 실제 공개 키를 수신했음을 확인할 수 있습니다. 이 지문을 다운로드 할 필요가 없습니다. 언제든지 지문을 표시하려면 이 명령을 사용하여 이메일 주소를 대체하십시오.

```
~]$ gpg2 --fingerprint jqdoe@example.com
```

"GPG 키 **ID"**는 공개 키를 식별하는 **8** 또는 **16**진수 숫자로 구성됩니다. 위의 예에서 **GPG** 키 **ID**는 **1B2AFA1C** 입니다. 대부분의 경우 키 **ID**를 요청하는 경우 **0x6789ABCD** 와 같이 키 **ID** 앞에 **0x**

를 추가합니다.

주의

암호를 잊어버린 경우 키를 사용할 수 없으며 해당 키를 사용하여 암호화된 데이터가 손실됩니다.

4.9.2.4. 공개 키 암호화 정보

1. Wikipedia - 공개 키 암호화
2. HowStuffWorks - 암호화

4.9.3. 공개 키 암호화에 openCryptoki 사용

openCryptoki 는 **tokens**라는 암호화 장치에 대한**API(**애플리케이션 프로그래밍 인터페이스**)**를 정의하는 **Public-Key Cryptography Standard** 인 **PKCS#11** 의 **Linux** 구현입니다. 토큰은 하드웨어 또는 소프트웨어로 구현될 수 있습니다. 이 장에서는 **Red Hat Enterprise Linux 7**에서 **openCryptoki** 시스템을 설치, 구성 및 사용하는 방법에 대해 설명합니다.

4.9.3.1. openCryptoki 설치 및 서비스 시작

테스트 목적으로 토큰의 소프트웨어 구현을 포함하여 시스템에 기본 **openCryptoki** 패키지를 설치하려면 **root** 로 다음 명령을 입력합니다.

```
~]# yum install opencryptoki
```

사용하려는 하드웨어 토큰 유형에 따라 특정 사용 사례에 대한 지원을 제공하는 추가 패키지를 설치해야 할 수 있습니다. 예를 들어 신뢰할 수 있는 플랫폼 모듈 **(TPM)** 장치에 대한 지원을 받으려면 **opencryptoki-tpmtok** 패키지를 설치해야 합니다.

YUM 패키지 관리자를 사용하여 패키지를 설치하는 방법에 대한 일반적인 정보는 **Red Hat Enterprise Linux 7** 시스템 관리자 가이드의 패키지 설치 섹션을 참조하십시오.

openCryptoki 서비스를 활성화하려면 **pkcsslotd** 데몬을 실행해야 합니다. **root** 로 다음 명령을 실행하여 현재 세션의 데몬을 시작합니다.

```
~]# systemctl start pkcsslotd
```

부팅 시 서비스가 자동으로 시작되도록 하려면 다음 명령을 입력합니다.

```
~]# systemctl enable pkcsslotd
```

systemd 대상을 사용하여 서비스 관리 방법에 대한 자세한 내용은 **Red Hat Enterprise Linux 7** 시스템 관리자 가이드의 **systemd**를 사용하여 서비스 관리 장을 참조하십시오.

4.9.3.2. openCryptoki 구성 및 사용

시작되면 **pkcsslotd** 데몬은 **/etc/opencryptoki/opencryptoki.conf** 구성 파일을 읽습니다. 이 구성 파일은 시스템 및 슬롯과 함께 작동하도록 구성된 토큰에 대한 정보를 수집하는 데 사용됩니다.

파일은 키-값 쌍을 사용하여 개별 슬롯을 정의합니다. 각 슬롯 정의에는 설명, 사용할 토큰 라이브러리의 사양, 슬롯 제조업체의 **ID**가 포함될 수 있습니다. 선택적으로 슬롯의 하드웨어 및 펌웨어 버전을 정의할 수 있습니다. 파일 형식에 대한 설명과 개별 키에 대한 자세한 설명과 할당할 수 있는 값은 **opencryptoki.conf(5)** 매뉴얼 페이지를 참조하십시오.

런타임에 **pkcsslotd** 데몬 동작을 수정하려면 **pkcsconf** 유틸리티를 사용합니다. 이 툴을 사용하면 데몬 상태를 표시 및 구성하고 현재 구성된 슬롯과 토큰을 나열하고 수정할 수 있습니다. 예를 들어 토큰에 대한 정보를 표시하려면 다음 명령을 실행합니다(**pkcsslotd** 데몬과 통신해야 하는 루트가 아닌 모든 사용자는 **pkcs11** 시스템 그룹의 일부여야 함)

```
~]$ pkcsconf -t
```

pkcsconf 툴에서 사용할 수 있는 인수 목록은 **pkcsconf(1)** 도움말 페이지를 참조하십시오.

주의

*이 그룹의 모든 멤버가 구성된 **PKCS#11** 토큰에 액세스할 수 없도록 **OpenCryptoki** 서비스의 다른 사용자를 차단할 수 있으므로 완전히 신뢰할 수 있는 사용자만 **pkcs11** 그룹에서 멤버십을 할당해야 합니다. 이 그룹의 모든 멤버는 **openCryptoki** 의 다른 사용자 권한으로 임의의 코드를 실행할 수도 있습니다.*

4.9.4. 스마트 카드를 사용하여 OpenSSH에 인증 정보 제공

*스마트 카드는 **USB** 토큰, **MicroSD** 또는 **Smartcard** 양식 인수의 경량 하드웨어 보안 모듈입니다. 원격으로 관리할 수 있는 보안 키 저장소를 제공합니다. **Red Hat Enterprise Linux 7**에서 **OpenSSH**는 스마트 카드를 사용한 인증을 지원합니다.*

***OpenSSH**에서 스마트 카드를 사용하려면 카드의 공개 키를 **~/.ssh/authorized_keys** 파일에 저장하십시오. 클라이언트에 **opensc** 패키지에서 제공하는 **PKCS#11** 라이브러리를 설치합니다. **PKCS#11** 은 토큰이라는 암호화 장치에 대한 **API**(애플리케이션 프로그래밍 인터페이스)를 정의하는 **Public-Key Cryptography Standard**입니다. 다음 명령을 **root** 로 입력합니다.*

```
~]# yum install opensc
```

4.9.4.1. 카드에서 공개 키 검색

*카드의 키를 나열하려면 **ssh-keygen** 명령을 사용합니다. **-D** 지시문을 사용하여 공유 라이브러리(다음 예제의 **OpenSC**)를 지정합니다.*

```
~]$ ssh-keygen -D /usr/lib64/pkcs11/opensc-pkcs11.so
ssh-rsa AAAAB3NzaC1yc[...]+g4Mb9
```

4.9.4.2. 서버에 공개 키 저장

*원격 서버에서 스마트 카드를 사용하여 인증을 활성화하려면 공개 키를 원격 서버로 전송합니다. 검색된 문자열(키)을 복사하여 원격 쉘에 붙여넣거나 다음 예제의 파일(다음 예에서**smartcard.pub**)에 키를 저장하고 **ssh-copy-id** 명령을 사용하여 키를 저장하여 수행합니다.*

```
~]$ ssh-copy-id -f -i smartcard.pub user@hostname
user@hostname's password:

Number of key(s) added: 1
```

```
Now try logging into the machine, with:   "ssh user@hostname"
and check to make sure that only the key(s) you wanted were added.
```

*개인 키 파일없이 공개 키를 저장하려면 **SSH_COPY_ID_LEGACY=1** 환경 변수 또는 **-f** 옵션을 사용해야 합니다.*

4.9.4.3. 스마트 카드에 키를 사용하여 서버에 인증

***OpenSSH**는 스마트 카드에서 공개 키를 읽고 키 자체를 노출하지 않고 개인 키로 작업을 수행할 수 있습니다. 즉, 개인 키는 카드를 떠나지 않습니다. 인증을 위해 스마트 카드를 사용하여 원격 서버에 연결하려면 다음 명령을 입력하고 **pin**을 입력하여 카드를 보호합니다.*

```
[localhost ~]$ ssh -I /usr/lib64/pkcs11/opensc-pkcs11.so hostname
Enter PIN for 'Test (UserPIN)':
[hostname ~]$
```

호스트 이름을 연결하려는 실제 호스트 이름으로 바꿉니다.

*원격 서버에 연결할 때 불필요한 입력을 저장하려면 **PKCS#11** 라이브러리의 경로를 ~/**.ssh/config** 파일에 저장합니다.*

```
Host hostname
   PKCS11Provider /usr/lib64/pkcs11/opensc-pkcs11.so
```

*추가 옵션 없이 **ssh** 명령을 실행하여 연결합니다.*

```
[localhost ~]$ ssh hostname
Enter PIN for 'Test (UserPIN)':
[hostname ~]$
```

4.9.4.4. ssh-agent 를 사용하여 자동 PIN 로깅 사용

***ssh-agent** 사용을 시작하도록 환경 변수를 설정합니다. **ssh-agent** 가 일반 세션에서 이미 실행되고 있기 때문에 대부분의 경우 이 단계를 건너뛸 수 있습니다. 다음 명령을 사용하여 인증 에이전트에 연결할 수 있는지 확인합니다.*

```
~]$ ssh-add -l
Could not open a connection to your authentication agent.
~]$ eval `ssh-agent`
```

이 키를 사용하여 연결할 때마다 **PIN**을 작성하지 않도록 하려면 다음 명령을 실행하여 에이전트에 카드를 추가합니다.

```
~]$ ssh-add -s /usr/lib64/pkcs11/opensc-pkcs11.so
Enter PIN for 'Test (UserPIN)':
Card added: /usr/lib64/pkcs11/opensc-pkcs11.so
```

ssh-agent 에서 카드를 제거하려면 다음 명령을 사용합니다.

```
~]$ ssh-add -e /usr/lib64/pkcs11/opensc-pkcs11.so
Card removed: /usr/lib64/pkcs11/opensc-pkcs11.so
```

참고

FIPS 201-2는 카드에 저장된 디지털 서명 키를 사용하기 위한 조건으로 개인 **ID** 확인자**(PIV)** 카드홀더에 의한 명시적인 사용자 조치를 요구합니다. **OpenSC**가 이 요구사항을 올바르게 시행합니다.

그러나 일부 애플리케이션의 경우 카드 보유자가 각 서명에 대해 **PIN**을 입력하도록 요구하는 것은 비현실적입니다. 스마트 카드 **PIN**을 캐시하려면 **/etc/opensc-x86_64.conf** 의 **pin_cache_ignore_user_consent = true;** 옵션 앞에 **#** 문자를 제거하십시오.

자세한 내용은 **PIV Digital Signature Key (NISTIR 7863)** 보고서에 대한 카드 소유자 인증을 참조하십시오.

4.9.4.5. 추가 리소스

하드웨어 또는 소프트웨어 토큰 설정은 **Red Hat Enterprise Linux 7**의 스마트 카드 지원에 설명되어 있습니다.

스마트 카드 및 유사한 **PKCS#11** 보안 토큰을 관리하고 사용하기 위한 **pkcs11-tool** 유틸리티에 대한 자세한 내용은 **pkcs11-tool(1)** 매뉴얼 페이지를 참조하십시오.

4.9.5. 신뢰할 수 있는 암호화 키

신뢰할 수 있고 암호화된 키는 커널 키링 서비스를 활용하는 커널에서 생성된 가변 길이의 대칭 키입니다. 키가 암호화되지 않은 형태로 사용자 공간에 표시되지 않는다는 사실은 무결성을 확인할 수 있음을 의

미하며, 이는EVM(Extended verification module)에 의해 실행 중인 시스템의 무결성을 검증하고 확인할 수 있음을 의미합니다. 사용자 수준 프로그램은 암호화된 Blob 형식의 키만 액세스할 수 있습니다.

신뢰할 수 있는 키에는 키를 만들고 암호화(seal)하는 데 사용되는 신뢰할 수 있는 플랫폼 모듈 (TPM) 칩이 필요합니다. TPM 은 스토리지 루트 키(SRK)라는 2048비트 RSA 키를 사용하여 키를 봉인합니다.

그 외에도, 신뢰할 수 있는 키는 TPM 의플랫폼 구성 레지스터 (PCR) 값의 특정 세트를 사용하여 봉인될 수도 있습니다. PCR 에는 BIOS, 부트 로더 및 운영 체제를 반영하는 무결성 관리 값 세트가 포함되어 있습니다. 즉, PCR-sealed 키는 암호화된 정확히 동일한 시스템에서 TPM 에서만 해독할 수 있습니다. 그러나 PCR-sealed 신뢰할 수 있는 키가 로드되면(인증 키에 추가) 연결된 PCR 값이 확인되면 새 커널(예: 새 커널이 부팅될 수 있도록) 새 (또는 향후) PCR 값으로 업데이트할 수 있습니다. 단일 키는 각각 다른 PCR 값을 가진 여러 Blob으로 저장할 수도 있습니다.

암호화된 키에는 TPM 이 필요하지 않으므로 커널 AES 암호화를 사용하므로 신뢰할 수 있는 키보다 더 빨라집니다. 암호화된 키는 커널 생성 임의 번호를 사용하여 생성되고 마스터 키로 사용자 공간 Blob으로 내보낼 때 암호화됩니다. 이 마스터 키는 신뢰할 수 있는 키 또는 사용자 키가 될 수 있습니다. 마스터 키가 신뢰할 수 있는 키가 아닌 경우 암호화 키는 암호화하는 데 사용되는 사용자 키만큼만 안전합니다.

4.9.5.1. 키 작업

키를 사용하여 작업을 수행하기 전에 신뢰할 수 있고 암호화된 키 커널 모듈이 시스템에 로드되어야 합니다. 다른 RHEL 커널 아키텍처에 커널 모듈을 로드하는 동안 다음 사항을 고려하십시오.

- x86_64 아키텍처를 사용하는 RHEL 커널의 경우 TRUSTED_KEYS 및 ENCRYPTED_KEYS 코드는 코어 커널 코드의 일부로 빌드됩니다. 결과적으로 x86_64 시스템 사용자는 신뢰할 수 있고 암호화된 키 모듈을 로드하지 않고도 이러한 키를 사용할 수 있습니다.

- 다른 모든 아키텍처의 경우 키를 사용하여 작업을 수행하기 전에 신뢰할 수 있고 암호화된 키 커널 모듈을 로드해야 합니다. 커널 모듈을 로드하려면 다음 명령을 실행합니다.

```
~]# modprobe trusted encrypted-keys
```

keyctl 유틸리티를 사용하여 신뢰할 수 있고 암호화된 키를 생성, 로드, 내보내기 및 업데이트할 수 있습니다. keyctl 사용에 대한 자세한 내용은 keyctl(1) 을 참조하십시오.

참고

TPM (예: 신뢰할 수 있는 키를 만들고 봉인하는 경우)을 사용하려면 활성화 및 활성 상태여야 합니다. 이는 일반적으로 시스템의 BIOS 설정 또는 유틸리티의 tpm-tools 패키지에 있는 tpm_setactive 명령을 사용하여 수행할 수 있습니다. 또한 TrouSers 애플리케이션을 설치해야 하며, TPM 과 통신하기 위해 실행되는 TrouSers 제품군의 일부인 tcsd 데몬을 설치해야 합니다.

TPM 을 사용하여 신뢰할 수 있는 키를 만들려면 다음 구문으로 keyctl 명령을 실행합니다.

```
~]$ keyctl add trusted name "new keylength [options]" keyring
```

위의 구문을 사용하여 예제 명령을 다음과 같이 구성할 수 있습니다.

```
~]$ keyctl add trusted kmk "new 32" @u
642500861
```

위의 예제에서는 길이가 32바이트(256비트)인 kmk 라는 신뢰할 수 있는 키를 생성하고 사용자 인증 키(@u)에 배치합니다. 키 길이는 32~128바이트(256비트에서 1024비트)를 가질 수 있습니다. show 하위 명령을 사용하여 커널 인증 키의 현재 구조를 나열합니다.

```
~]$ keyctl show
Session Keyring
     -3 --alswrv    500   500  keyring: _ses
 97833714 --alswrv    500    -1   \_ keyring: _uid.1000
642500861 --alswrv    500   500       \_ trusted: kmk
```

print 하위 명령은 암호화된 키를 표준 출력에 출력합니다. 키를 사용자 공간 Blob으로 내보내려면 파이프 하위 명령을 다음과 같이 사용합니다.

```
~]$ keyctl pipe 642500861 > kmk.blob
```

사용자 공간 Blob에서 신뢰할 수 있는 키를 로드하려면 Blob을 인수로 다시 추가합니다.To load the trusted key from the user-space blob, use the add command again with the blob as an argument:

```
~]$ keyctl add trusted kmk "load `cat kmk.blob`" @u
268728824
```

그런 다음 TPM-sealed trusted key를 사용하여 암호화 된 보안 키를 만들 수 있습니다. 다음 명령 구문은 암호화된 키를 생성하는 데 사용됩니다.

```
~]$ keyctl add encrypted name "new [format] key-type:master-key-name keylength" keyring
```

위의 구문을 기반으로 이미 생성된 신뢰할 수 있는 키를 사용하여 암호화된 키를 생성하는 명령을 다음과 같이 구성할 수 있습니다.

```
~]$ keyctl add encrypted encr-key "new trusted:kmk 32" @u
159771175
```

TPM 을 사용할 수 없는 시스템에서 암호화된 키를 만들려면 임의의 숫자 시퀀스를 사용하여 사용자 키를 생성한 다음 실제 암호화된 키를 봉인하는 데 사용됩니다.

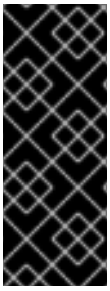
```
~]$ keyctl add user kmk-user "`dd if=/dev/urandom bs=1 count=32 2>/dev/null`" @u
427069434
```

그런 다음 **random-number** 사용자 키를 사용하여 암호화된 키를 생성합니다.

```
~]$ keyctl add encrypted encr-key "new user:kmk-user 32" @u
1012412758
```

list 하위 명령을 사용하여 지정된 커널 인증 키의 모든 키를 나열할 수 있습니다.

```
~]$ keyctl list @u
2 keys in keyring:
427069434: --alswrv  1000  1000 user: kmk-user
1012412758: --alswrv  1000  1000 encrypted: encr-key
```



중요

마스터 신뢰할 수 있는 키로 봉인되지 않은 암호화된 키는 암호화하는 데 사용되는 사용자 마스터 키(**random-number** 키)만큼만 안전합니다. 따라서 마스터 사용자 키는 부팅 프로세스 중 가능한 한 안전하게 로드되어야 하며 부팅 프로세스 중 초기에 가급적이어야 합니다.

4.9.5.2. 추가 리소스

다음 오프라인 및 온라인 리소스를 사용하여 신뢰할 수 있고 암호화된 키 사용과 관련된 추가 정보를 얻을 수 있습니다.

설치된 문서

- **keyctl(1) - keyctl** 유틸리티 및 해당 하위 명령 사용에 대해 설명합니다.

온라인 문서

- **Red Hat Enterprise Linux 7 SELinux** 사용자 및 관리자 가이드 **- Red Hat Enterprise Linux 7**용 **SELinux** 사용자 및 관리자 가이드는 **Apache HTTP Server** 와 같은 다양한 서비스를 사용하여 **SELinux** 및 문서를 구성하고 사용하는 방법에 대해 자세히 설명합니다.

- **https://www.kernel.org/doc/Documentation/security/keys-trusted-encrypted.txt - Linux** 커널의 신뢰할 수 있고 암호화된 키 기능에 대한 공식 문서입니다.

다음을 참조하십시오.

- **A.1.1**절. "고급 암호화 표준 **- AES" Advanced Encryption Standard** 에 대한 간결한 설명을 제공합니다.

- **A.2**절. "공개 키 암호화" 공용 키 암호화 방식과 사용하는 다양한 암호화 프로토콜을 설명합니다.**Describes the public-key cryptographic approach and the various cryptographic protocols it uses.**

4.9.6. Random Number Generator 사용

쉽게 손상될 수 없는 보안 암호화 키를 생성하려면 임의의 숫자의 소스가 필요합니다. 일반적으로 숫자가 클수록 고유 키를 얻을 가능성이 높아집니다. 임의의 숫자를 생성하는 엔트로피 는 일반적으로 컴퓨팅 환경 **"noise"** 또는 하드웨어 난수 생성기 를 사용하여 가져옵니다.

rng-tools 패키지의 일부인 **rngd** 데몬은 환경 잡음과 하드웨어 난수 생성기를 모두 사용하여 엔트로피 추출을 수행할 수 있습니다. 데몬은 임의성의 소스에서 제공한 데이터가 충분히 임의의지 여부를 확인한 다음 커널의 난수 엔트로피 풀에 저장합니다. 생성되는 임의의 숫자는 **/dev/random** 및 **/dev/urandom** 문자 장치를 통해 사용할 수 있습니다.

/dev/random 과 **/dev/urandom** 의 차이점은 이전이 차단 장치라는 것이며, 이는 엔트로피의 양이 적절히 임의의 출력을 생성하는 데 충분하지 않다는 것을 결정할 때 숫자 공급을 중지한다는 것을 의미합니다. 반대로 **/dev/urandom** 은 커널의 엔트로피 풀을 재사용하는 비차단 소스이므로 의사랜덤 수의 무제한 공급을 제공할 수 있습니다. 따라서 **/dev/urandom** 은 장기 암호화 키를 만드는 데 사용해서는 안 됩니다.

rng-tools 패키지를 설치하려면 **root** 사용자로 다음 명령을 실행합니다.

```
~]# yum install rng-tools
```

rngd 데몬을 시작하려면 **root** 로 다음 명령을 실행하십시오.

```
~]# systemctl start rngd
```

데몬 상태를 쿼리하려면 다음 명령을 사용합니다.

```
~]# systemctl status rngd
```

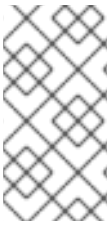
선택적 매개변수로 **rngd** 데몬을 시작하려면 직접 실행합니다. 예를 들어 **/dev/hwrandom**이외의 임의의 숫자 입력의 대체 소스를 지정하려면 다음 명령을 사용합니다.

```
~]# rngd --rng-device=/dev/hwrng
```

이전 명령은 임의의 숫자를 읽는 장치로 **/dev/hwrng** 를 사용하여 **rngd** 데몬을 시작합니다. 마찬가지로 **-o (**또는 **--random-device)** 옵션을 사용하여 임의의 숫자 출력에 대해 커널 장치를 선택할 수 있습니다(기본 **/dev/random**제외). 사용 가능한 모든 옵션 목록은 **rngd(8)** 매뉴얼 페이지를 참조하십시오.

지정된 시스템에서 사용할 수 있는 엔트로피의 소스를 확인하려면 **root** 로 다음 명령을 실행합니다.

```
~]# rngd -vf
Unable to open file: /dev/tpm0
Available entropy sources:
 DRNG
```



참고

rngd -v 명령을 입력하면 해당 프로세스가 백그라운드에서 계속 실행됩니다. **b, --background** 옵션(단 데몬 이전)은 기본적으로 적용됩니다.

TPM 장치가 없는 경우 엔트로피의 소스로 **Intel Digital Random Number Generator(DRNG)**만 표시됩니다. **CPU**가 **RDRAND** 프로세서 명령을 지원하는지 확인하려면 다음 명령을 입력합니다.

```
~]$ cat /proc/cpuinfo | grep rdrand
```

참고

*자세한 내용은 **Intel Digital Random Number Generator (DRNG) 소프트웨어 구현 가이드**를 참조하십시오.*

***rng-tools** 패키지에는 데이터의 임의성을 확인하는 데 사용할 수 있는 **rngtest** 유틸리티가 포함되어 있습니다. **/dev/random** 의 출력의 임의성 수준을 테스트하려면 다음과 같이 **rngtest** 툴을 사용합니다.*

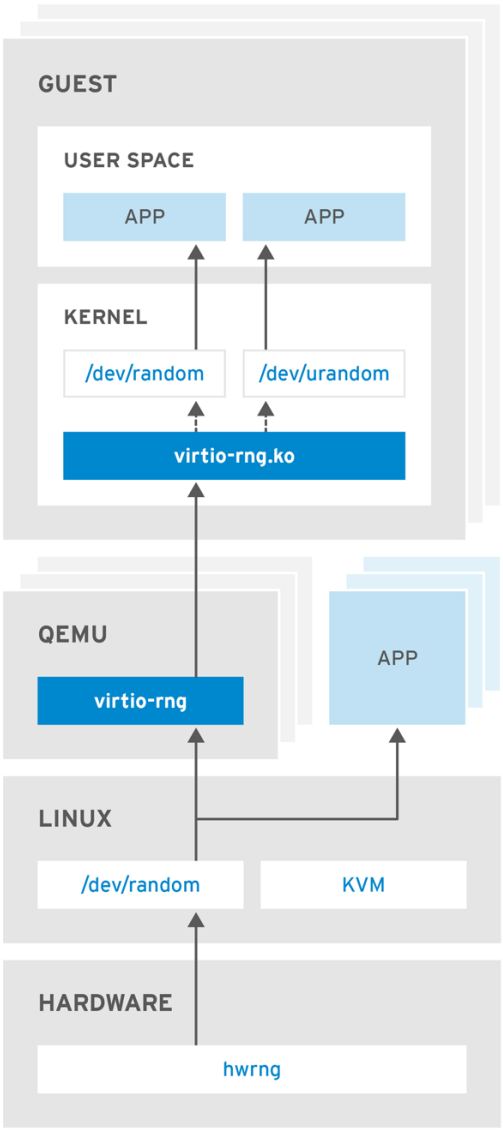
```
~]$ cat /dev/random | rngtest -c 1000
rngtest 5
Copyright (c) 2004 by Henrique de Moraes Holschuh
This is free software; see the source for copying conditions. There is NO warranty; not even for
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

rngtest: starting FIPS tests...
rngtest: bits received from input: 20000032
rngtest: FIPS 140-2 successes: 998
rngtest: FIPS 140-2 failures: 2
rngtest: FIPS 140-2(2001-10-10) Monobit: 0
rngtest: FIPS 140-2(2001-10-10) Poker: 0
rngtest: FIPS 140-2(2001-10-10) Runs: 0
rngtest: FIPS 140-2(2001-10-10) Long run: 2
rngtest: FIPS 140-2(2001-10-10) Continuous run: 0
rngtest: input channel speed: (min=1.171; avg=8.453; max=11.374)Mibits/s
rngtest: FIPS tests speed: (min=15.545; avg=143.126; max=157.632)Mibits/s
rngtest: Program run time: 2390520 microseconds
```

***rngtest** 툴 출력에 표시되는 실패 수는 테스트된 데이터의 임의성이 충분하지 않으며 신뢰할 수 없음을 나타냅니다. **rngtest** 유틸리티에 사용 가능한 옵션 목록은 **rngtest(1)** 매뉴얼 페이지를 참조하십시오.*

***Red Hat Enterprise Linux 7**은 **KVM** 가상 머신에서 호스트 머신에서 엔트로피에 액세스할 수 있는 **virtio RNG (Random Number Generator)** 장치를 도입했습니다. 권장 설정을 사용하면 **hwrng feed**를 호스트 **Linux** 커널의 엔트로피 풀(**/dev/random**)에 입력하며 **QEMU** 는 게스트가 요청한 엔트로피의 소스로 **/dev/random** 을 사용합니다.*

그림 4.1. virtio RNG 장치



RHEL_453350_0717

[D]

*이전에는 **Red Hat Enterprise Linux 7.0** 및 **Red Hat Enterprise Linux 6** 게스트에서 **rngd** 사용자 공간 데몬을 통해 호스트에서 엔트로피를 사용할 수 있었습니다. 데몬을 설정하는 것은 각 **Red Hat Enterprise Linux** 설치를 위한 수동 단계였습니다. **Red Hat Enterprise Linux 7.1**에서는 수동 단계가 제거되어 전체 프로세스가 원활하고 자동으로 수행됩니다. 이제 **rngd**를 사용할 필요가 없으며 사용 가능한 엔트로피가 특정 임계값 아래로 떨어지면 게스트 커널 자체 호스트에서 엔트로피를 가져옵니다. 게스트 커널은 요청하는 즉시 애플리케이션에서 임의의 숫자를 사용할 수 있도록 할 수 있는 위치에 있습니다.*

***Red Hat Enterprise Linux** 설치 프로그램 **Anaconda** 는 이제 설치 프로그램 이미지에 **virtio-rng** 모듈을 제공하여 **Red Hat Enterprise Linux** 설치 중에 호스트 엔트로피를 사용할 수 있도록 합니다.*

중요

시나리오에서 사용해야 하는 난수 생성기를 올바르게 결정하려면 **Red Hat Enterprise Linux** 난수 생성기 인터페이스 이해 문서를 참조하십시오.

4.10. 정책 기반 암호 해독을 사용하여 암호화된 볼륨의 자동 잠금 해제 구성

정책 기반 암호(**Policy-Based Decryption**)는 시스템에 연결된 사용자 암호,**TPM**(**Trusted Platform Module**) 장치, 예를 들어 스마트 카드 또는 특수 네트워크 서버의 지원과 같은 다른 방법을 사용하여 물리적 및 가상 머신에서 하드 드라이브의 암호화된 루트 및 보조 볼륨을 잠금 해제할 수 있는 기술 컬렉션입니다.

기술로 **PBD**를 사용하면 다른 잠금 해제 방법을 정책과 결합하여 동일한 볼륨을 다른 방식으로 잠금 해제할 수 있습니다. **Red Hat Enterprise Linux**에서 **PBD**의 현재 구현은 **Clevis** 프레임워크와 **pins**라는 플러그인으로 구성되어 있습니다. 각 핀은 별도의 잠금 해제 기능을 제공합니다. 현재 사용 가능한 두 개의 핀만 **TPM** 또는 네트워크 서버로 볼륨을 잠금 해제할 수 있는 핀입니다.

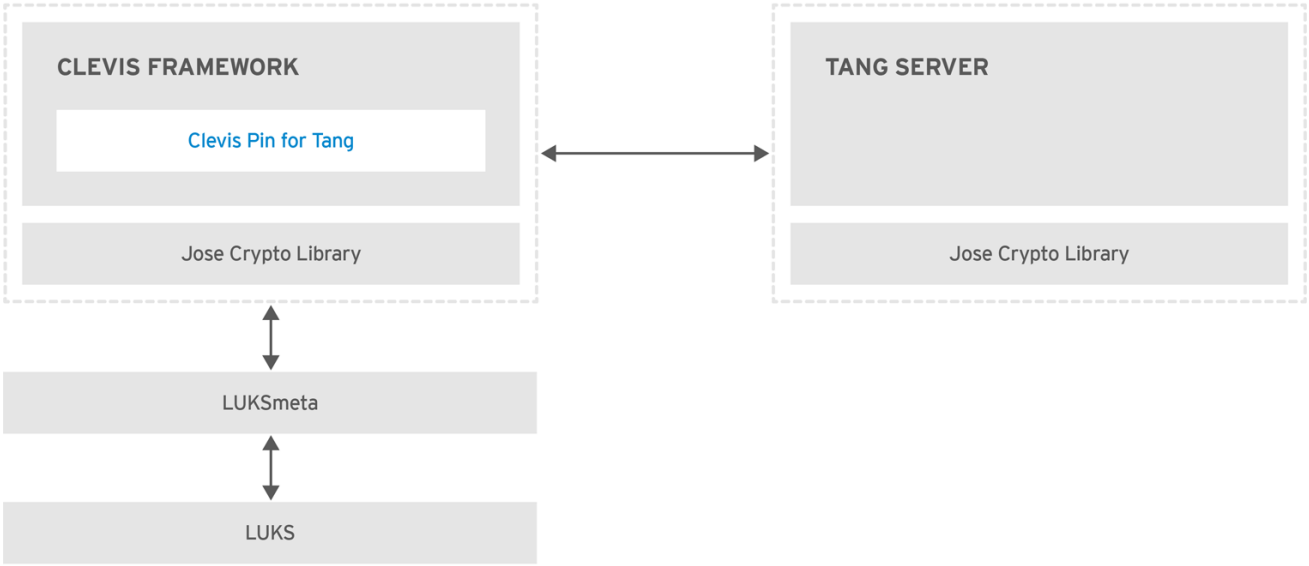
NBDE(**Network Bound Disc Encryption**)는 암호화된 볼륨을 특수 네트워크 서버에 바인딩할 수 있는 **PBD** 기술의 하위 범주입니다. **Clevis**의 현재 구현에는 **Tang** 서버 및 **Tang** 서버 자체에 대한 **Clevis** 핀이 포함되어 있습니다.

4.10.1. 네트워크 Bound 디스크 암호화

NBDE(**Network-Bound Disk Encryption**)를 사용하면 시스템을 다시 시작할 때 수동으로 암호를 입력하지 않고도 물리적 및 가상 머신에서 하드 드라이브의 루트 볼륨을 암호화할 수 있습니다.

Red Hat Enterprise Linux 7에서 **KnativeServing**는 다음과 같은 구성 요소 및 기술을 통해 구현됩니다.

그림 4.2. Clevis 및 Tang을 사용한 네트워크 Bound 디스크 암호화



Tang은 데이터를 네트워크 상에 바인딩하는 서버입니다. 시스템이 특정 보안 네트워크에 바인딩될 때 데이터를 포함하는 시스템을 사용할 수 있습니다. **Tang**은 상태 비저장이며 **TLS** 또는 인증이 필요하지 않습니다. 서버가 모든 암호화 키를 저장하고 사용된 모든 키에 대한 지식이 있는 에스크로 기반 솔루션과 달리 **Tang**은 클라이언트 키와 상호 작용하지 않으므로 클라이언트로부터 식별 정보를 얻을 수 없습니다.

Clevis 는 자동 암호 해독을 위한 플러그형 프레임워크입니다. **KnativeServing**에서 **Clevis**는 **LUKS** 볼륨의 자동 잠금을 해제하는 기능을 제공합니다. **clevis** 패키지는 기능의 클라이언트 쪽을 제공합니다.

Clevis PIN은 **Clevis** 프레임워크의 플러그인입니다. 이러한 핀 중 하나는 **DASD** 서버 **- Tang**과의 상호 작용을 구현하는 플러그인입니다.

Clevis 및 **Tang**은 네트워크 바인딩 암호화를 제공하는 일반 클라이언트 및 서버 구성 요소입니다. **Red Hat Enterprise Linux 7**에서는 **LUKS**와 함께 를 사용하여 네트워크 **Bound** 디스크 암호화를 수행하기 위해 루트 및 루트가 아닌 스토리지 볼륨을 암호화 및 암호 해독하는 데 사용됩니다.

클라이언트 및 서버 측 구성 요소 모두 **José** 라이브러리를 사용하여 암호화 및 암호 해독 작업을 수행합니다.

Clevis 프로비저닝을 시작할 때 **Tang** 서버의 **Clevis** 핀은 **Tang** 서버의 공개된 **symmetric** 키 목록을 가져옵니다. 또는 키가 **symmetric**이므로 **Tang**의 공개 키 목록을 대역에서 배포할 수 있으므로 클라이언트가 **Tang** 서버에 액세스하지 않고도 작동할 수 있습니다. 이 모드를 오프라인 프로비저닝 이라고 합니다.

Tang의 **Clevis** 핀은 공개 키 중 하나를 사용하여 고유하고 암호화 방식으로 암호화 키를 생성합니다.

*이 키를 사용하여 데이터를 암호화하면 키가 삭제됩니다. **Clevis** 클라이언트는 이 프로비저닝 작업에서 생성한 상태를 편리한 위치에 저장해야 합니다. 데이터를 암호화하는 이 프로세스는 프로비저닝 단계입니다. **Clevis**의 프로비저닝 상태는 **luksmeta** 패키지를 활용하는 **LUKS** 헤더에 저장됩니다.*

클라이언트가 데이터에 액세스할 준비가 되면 프로비저닝 단계에서 생성된 메타데이터를 로드하고 암호화 키를 복구할 수 있습니다. 이 과정은 회복 단계입니다.

***Clevis**는 자동으로 잠금 해제될 수 있도록 핀을 사용하여 **LUKS** 볼륨을 바인딩합니다. 바인딩 프로세스가 성공적으로 완료되면 제공된 **Dracut unlocker**를 사용하여 디스크의 잠금을 해제할 수 있습니다.*

*네트워크 연결이 설정되기 전에 시작해야 하는 파일 시스템이 포함된 **/tmp,/var, /var** 및 **/usr/local/** 디렉토리와 같은 모든 **LUKS** 암호화 장치는 루트 볼륨으로 간주됩니다. 또한 **/var/log/, var/log/audit/** 또는 **/opt** 과 같이 네트워크가 실행되기 전에 실행되는 모든 마운트 지점을 루트 장치로 전환한 후 조기에 마운트해야 합니다. **/etc/fstab** 파일에 **_netdev** 옵션이 없는 상태에서 **root** 볼륨을 식별할 수도 있습니다.*

4.10.2. 암호화 클라이언트 설치 - Clevis

***Clevis** 플러그형 프레임워크와 암호화된 볼륨(클라이언트)이 있는 머신에 고정을 설치하려면 **root** 로 다음 명령을 입력합니다.*

```
~]# yum install clevis
```

*데이터의 암호를 해독하려면 **clevis decrypt** 명령을 사용하여 **JWE(**암호화 텍스트**)**를 제공합니다.*

```
~]$ clevis decrypt < JWE > PLAINTEXT
```

*자세한 내용은 기본 제공 **CLI** 도움말을 참조하십시오.*

```
~]$ clevis
Usage: clevis COMMAND [OPTIONS]

  clevis decrypt      Decrypts using the policy defined at encryption time
  clevis encrypt http Encrypts using a REST HTTP escrow server policy
  clevis encrypt sss  Encrypts using a Shamir's Secret Sharing policy
  clevis encrypt tang Encrypts using a Tang binding server policy
  clevis encrypt tpm2 Encrypts using a TPM2.0 chip binding policy

~]$ clevis decrypt
Usage: clevis decrypt < JWE > PLAINTEXT

Decrypts using the policy defined at encryption time
```

```
~]$ clevis encrypt tang
Usage: clevis encrypt tang CONFIG < PLAINTEXT > JWE

Encrypts using a Tang binding server policy

This command uses the following configuration properties:

  url: <string>   The base URL of the Tang server (REQUIRED)

  thp: <string>   The thumbprint of a trusted signing key

  adv: <string>   A filename containing a trusted advertisement
  adv: <object>   A trusted advertisement (raw JSON)

Obtaining the thumbprint of a trusted signing key is easy. If you
have access to the Tang server's database directory, simply do:

    $ jose jwk thp -i $DBDIR/$SIG.jwk

Alternatively, if you have certainty that your network connection
is not compromised (not likely), you can download the advertisement
yourself using:

    $ curl -f $URL/adv > adv.jws
```

4.10.3. SELinux를 사용하여 Enforcing 모드에서 Tang Server 배포

Red Hat Enterprise Linux 7.7 이상에서는 **tangd_port_t SELinux** 유형을 제공하며 **Tang** 서버를 **SELinux** 강제 모드로 제한된 서비스로 배포할 수 있습니다.

사전 요구 사항

- **policycoreutils-python-utils** 패키지 및 해당 종속 항목이 설치됩니다.

절차

1. **tang** 패키지 및 해당 종속 항목을 설치하려면 **root** 로 다음 명령을 입력합니다.

```
~]# yum install tang
```

2. **(예: 7500/tcp) .occupied** 포트를 선택하고 **tangd** 서비스가 해당 포트에 바인딩되도록 허용합니다.

```
~]# semanage port -a -t tangd_port_t -p tcp 7500
```

포트는 한 번에 하나의 서비스에서만 사용할 수 있으므로 이미 사용되고 있는 포트를 사용하

려는 경우 **ValueError**를 의미합니다. 포트가 이미 정의된 오류 메시지입니다.

3. 방화벽에서 포트를 엽니다.

```
~]# firewall-cmd --add-port=7500/tcp
~]# firewall-cmd --runtime-to-permanent
```

4. **systemd**를 사용하여 **tangd** 서비스를 활성화합니다.

```
~]# systemctl enable tangd.socket
Created symlink from /etc/systemd/system/multi-user.target.wants/tangd.socket to
/usr/lib/systemd/system/tangd.socket.
```

5. 덮어쓰기 파일을 생성합니다.

```
~]# systemctl edit tangd.socket
```

6. 다음 편집기 화면에서 **/etc/systemd/system/tangd.socket.d/** 디렉터리에 있는 빈 **override.conf** 파일을 열고 다음 행을 추가하여 **Tang** 서버의 기본 포트를 **80**에서 이전에 선택한 숫자로 변경합니다.

```
[Socket]
ListenStream=
ListenStream=7500
```

파일을 저장하고 편집기를 종료합니다.

7. 변경된 구성을 다시 로드하고 **tangd** 서비스를 시작합니다.

```
~]# systemctl daemon-reload
```

8. 구성이 작동하는지 확인합니다.

```
~]# systemctl show tangd.socket -p Listen
Listen=[::]:7500 (Stream)
```

9. **tangd** 서비스를 시작합니다.

```
~]# systemctl start tangd.socket
```

tangd 는 **systemd** 소켓 활성화 메커니즘을 사용하므로 첫 번째 연결이 들어오는 즉시 서버가 시작됩니다. 새로 생성된 암호화 키 세트는 처음 시작할 때 자동으로 생성됩니다.

수동 키 생성과 같은 암호화 작업을 수행하려면 **jose** 유틸리티를 사용합니다. **jose -h** 명령을 입력하거나 **jose(1)** 매뉴얼 페이지를 참조하십시오.

예 4.4. Tang 키 교체

정기적으로 키를 순환하는 것이 중요합니다. 회전해야하는 정확한 간격은 애플리케이션, 키 크기 및 기관 정책에 따라 다릅니다. 몇 가지 일반적인 권장 사항은 **Cryptographic Key Length Recommendation** 페이지를 참조하십시오.

키를 회전하려면 키 데이터베이스 디렉터리(일반적으로 **/var/db/tang**)에서 새 키 생성으로 시작합니다. 예를 들어 다음 명령으로 새 서명을 생성하고 키를 교환할 수 있습니다.

```
~]# DB=/var/db/tang
~]# jose jwk gen -i '{"alg":"ES512"}' -o $DB/new_sig.jwk
~]# jose jwk gen -i '{"alg":"ECMR"}' -o $DB/new_exc.jwk
```

이전 키의 이름을 지정하여 앞에 . 을 지정하여 광고에서 숨깁니다. 다음 예제의 파일 이름은 키 데이터베이스 디렉터리의 실제 및 고유한 파일 이름과 다릅니다.

```
~]# mv $DB/old_sig.jwk $DB/.old_sig.jwk
~]# mv $DB/old_exc.jwk $DB/.old_exc.jwk
```

Tang은 모든 변경 사항을 즉시 선택합니다. 다시 시작하지 않아도 됩니다.

이 시점에서 새 클라이언트 바인딩은 새 키를 선택하고 이전 클라이언트는 이전 키를 계속 사용할 수 있습니다. 모든 이전 클라이언트가 새 키를 사용해야 하는 경우 이전 키를 제거할 수 있습니다.

주의

클라이언트가 계속 사용하는 동안 이전 키를 제거하면 데이터 손실이 발생할 수 있습니다.

4.10.3.1. 고가용성 시스템 배포

***Tang**은 고가용성 배포를 구축하기 위한 두 가지 방법을 제공합니다.*

1. *클라이언트 중복(권장)*

   *클라이언트는 여러 **Tang** 서버에 바인딩할 수 있는 기능을 사용하여 구성해야 합니다. 이 설정에서 각 **Tang** 서버에는 고유한 키가 있으며 클라이언트는 이러한 서버의 하위 집합에 연결하여 암호를 해독할 수 있습니다. **Clevis**는 이미 **sss** 플러그인을 통해 이 워크플로를 지원합니다.*

   이 설정에 대한 자세한 내용은 다음 도움말 페이지를 참조하십시오.

   - ***Tang(8), High Availability** 섹션*
   - ***Clevis(1), Shamir**의 시크릿 공유 섹션*
   - ***clevis-encrypt-sss(1)***

   ***Red Hat**은 고가용성 배포에 이 방법을 권장합니다.*

2. *키 공유*

   *중복성을 위해 **Tang**의 인스턴스를 두 개 이상 배포할 수 있습니다. 두 번째 또는 후속 인스턴스를 설정하려면 **tang** 패키지를 설치하고 **SSH**를 통해 **rsync** 를 사용하여 키 디렉터리를 새 호스*

트에 복사합니다. 키를 공유할 경우 키 손상의 위험이 증가하고 추가 자동화 인프라가 필요하므로 이 방법을 사용하지 않는 것이 좋습니다.

4.10.4. Tang을 사용하여 Clevis 시스템의 Encryption Client 배포

사전 요구 사항

- Clevis 프레임워크가 설치되어 있어야 합니다. 참조 4.10.2절. “암호화 클라이언트 설치 - Clevis”
- Tang 서버를 사용할 수 있습니다. 참조 4.10.3절. “SELinux를 사용하여 Enforcing 모드에서 Tang Server 배포”

절차

Clevis 암호화 클라이언트를 Tang 서버에 바인딩하려면 clevis encrypt tang 하위 명령을 사용합니다.

```
~]$ clevis encrypt tang '{"url":"http://tang.srv"}' < PLAINTEXT > JWE
The advertisement contains the following signing keys:

_OsIk0T-E2l6qjfdDiwVmidoZjA

Do you wish to trust these keys? [ynYN] y
```

위 예제의 http://tang.srv URL을 tang 이 설치된 서버의 URL과 일치하도록 변경합니다. JWE 출력 파일에는 암호화된 암호화 텍스트가 포함되어 있습니다. 이 암호화 텍스트는 PLAINTEXT 입력 파일에서 읽습니다.

데이터의 암호를 해독하려면 clevis decrypt 명령을 사용하여 JWE(암호화 텍스트)를 제공합니다.

```
~]$ clevis decrypt < JWE > PLAINTEXT
```

자세한 내용은 clevis-encrypt-tang(1) 매뉴얼 페이지를 참조하거나 기본 제공 CLI 도움말을 사용합니다.

```
~]$ clevis
Usage: clevis COMMAND [OPTIONS]

  clevis decrypt      Decrypts using the policy defined at encryption time
  clevis encrypt http Encrypts using a REST HTTP escrow server policy
  clevis encrypt sss  Encrypts using a Shamir's Secret Sharing policy
  clevis encrypt tang Encrypts using a Tang binding server policy
```

```
  clevis encrypt tang Encrypts using a Tang binding server policy
  clevis luks bind    Binds a LUKSv1 device using the specified policy
  clevis luks unlock  Unlocks a LUKSv1 volume

~]$ clevis decrypt
Usage: clevis decrypt < JWE > PLAINTEXT

Decrypts using the policy defined at encryption time

~]$ clevis encrypt tang
Usage: clevis encrypt tang CONFIG < PLAINTEXT > JWE

Encrypts using a Tang binding server policy

This command uses the following configuration properties:

  url: <string>   The base URL of the Tang server (REQUIRED)

  thp: <string>   The thumbprint of a trusted signing key

  adv: <string>   A filename containing a trusted advertisement
  adv: <object>   A trusted advertisement (raw JSON)

Obtaining the thumbprint of a trusted signing key is easy. If you
have access to the Tang server's database directory, simply do:

    $ jose jwk thp -i $DBDIR/$SIG.jwk

Alternatively, if you have certainty that your network connection
is not compromised (not likely), you can download the advertisement
yourself using:

    $ curl -f $URL/adv > adv.jws
```

4.10.5. TPM 2.0 정책을 사용하여 암호화 클라이언트 배포

64비트 **Intel** 또는 **64**비트 **AMD** 아키텍처가 있는 시스템에서는 신뢰할 수 있는 플랫폼 모듈 **2.0(TPM 2.0)** 칩을 사용하여 암호화되는 클라이언트를 배포하려면 **JSON** 구성 오브젝트 형식의 유일한 인수와 함께 **clevis encrypt tpm2** 하위 명령을 사용하십시오.

```
~]$ clevis encrypt tpm2 '{}' < PLAINTEXT > JWE
```

다른 계층 구조, 해시 및 키 알고리즘을 선택하려면 구성 속성을 지정합니다.

```
~]$ clevis encrypt tpm2 '{"hash":"sha1","key":"rsa"}' < PLAINTEXT > JWE
```

데이터의 암호를 해독하려면 암호화 텍스트**(JWE)**를 제공합니다.

```
~]$ clevis decrypt < JWE > PLAINTEXT
```

또한 핀은 플랫폼 구성 등록 **(PCR)** 상태에 대한 밀봉 데이터를 지원합니다. 이렇게 하면 **PCRs** 해시 값이 밀봉 시 사용된 정책과 일치하는 경우에만 데이터가 음소거되지 않을 수 있습니다.

예를 들어, 인덱스 **0** 및 **SHA1** 은행에 대해 **1**을 사용하여 데이터를 **PCR**에 봉인하려면 다음을 수행합니다.

```
~]$ clevis encrypt tpm2 '{"pcr_bank":"sha1","pcr_ids":"0,1"}' < PLAINTEXT > JWE
```

자세한 내용과 가능한 구성 속성 목록은 **clevis-encrypt-tpm2(1)** 매뉴얼 페이지를 참조하십시오.

4.10.6. 루트 볼륨의 수동 등록 구성

기존 **LUKS** 암호화된 루트 볼륨의 잠금을 자동으로 해제하려면 **clevis-luks** 하위 패키지를 설치하고 **clevis luks bind** 명령을 사용하여 볼륨을 **Tang** 서버에 바인딩합니다.

```
~]# yum install clevis-luks
```

```
~]# clevis luks bind -d /dev/sda tang '{"url":"http://tang.srv"}'
The advertisement contains the following signing keys:

_OsIk0T-E2l6qjfdDiwVmidoZjA

Do you wish to trust these keys? [ynYN] y
You are about to initialize a LUKS device for metadata storage.
Attempting to initialize it may result in data loss if data was
already written into the LUKS header gap in a different format.
A backup is advised before initialization is performed.

Do you wish to initialize /dev/sda? [yn] y
Enter existing LUKS password:
```

이 명령은 다음 네 가지 단계를 수행합니다.

1. **LUKS** 마스터 키와 동일한 엔트로피를 사용하여 새 키를 만듭니다.
2. **Clevis**를 사용하여 새 키를 암호화합니다.

3. ***LUKSMeta*를 사용하여 *LUKS* 헤더에 *Clevis JWE* 오브젝트를 저장합니다.**

4. ***LUKS*에 사용할 새 키를 활성화합니다.**

*이 디스크는 이제 **Clevis** 정책과 함께 기존 암호를 사용하여 잠금 해제할 수 있습니다. 자세한 내용은 **clevis-luks-bind(1)** 매뉴얼 페이지를 참조하십시오.*

참고

*바인딩 절차에서는 사용 가능한 **LUKS** 암호 슬롯이 하나 이상 있다고 가정합니다. **clevis luks bind** 명령은 슬롯 중 하나를 사용합니다.*

*Clevis JWE 오브젝트가 **LUKS** 헤더에 성공적으로 배치되었는지 확인하려면 **luksmeta show** 명령을 사용합니다.*

```
~]# luksmeta show -d /dev/sda
0   active empty
1   active cb6e8904-81ff-40da-a84a-07ab9ab5715e
2 inactive empty
3 inactive empty
4 inactive empty
5 inactive empty
6 inactive empty
7 inactive empty
```

초기 부팅 시스템이 디스크 바인딩을 처리할 수 있도록 하려면 이미 설치된 시스템에서 다음 명령을 입력합니다.

```
~]# yum install clevis-dracut
~]# dracut -f --regenerate-all
```

중요

고정 **IP** 구성**(DHCP** 제외**)**이 있는 클라이언트에 대해 **DASD**를 사용하려면 네트워크 구성을 수동으로 **dracut** 툴에 전달합니다. 예를 들면 다음과 같습니다.

```
~]# dracut -f --regenerate-all --kernel-cmdline "ip=192.0.2.10 netmask=255.255.255.0
gateway=192.0.2.1 nameserver=192.0.2.45"
```

또는 정적 네트워크 정보를 사용하여 **/etc/dracut.conf.d/** 디렉터리에 **.conf** 파일을 만듭니다. 예를 들어 다음과 같습니다.

```
~]# cat /etc/dracut.conf.d/static_ip.conf
kernel_cmdline="ip=10.0.0.103 netmask=255.255.252.0 gateway=10.0.0.1
nameserver=10.0.0.1"
```

초기 **RAM** 디스크 이미지를 다시 생성합니다.

```
~]# dracut -f --regenerate-all
```

자세한 내용은 **dracut.cmdline(7)** 매뉴얼 페이지를 참조하십시오.

4.10.7. Kickstart를 사용하여 자동화된 등록 구성

Clevis는 완전 자동화된 등록 프로세스를 제공하기 위해 **Kickstart**와 통합할 수 있습니다.

1.

Kickstart에 임시 암호로 **/boot** 이외의 모든 마운트 지점에 대해 **LUKS** 암호화가 활성화되도록 디스크를 파티션하도록 지시합니다. 암호는 이 등록 프로세스 단계에서 임시적입니다.

```
part /boot --fstype="xfs" --ondisk=vda --size=256
part / --fstype="xfs" --ondisk=vda --grow --encrypted --passphrase=temppass
```

예를 들어 **OSPP-complaint** 시스템에는 더 복잡한 구성이 필요합니다.

```
part /boot --fstype="xfs" --ondisk=vda --size=256
part / --fstype="xfs" --ondisk=vda --size=2048 --encrypted --passphrase=temppass
part /var --fstype="xfs" --ondisk=vda --size=1024 --encrypted --passphrase=temppass
part /tmp --fstype="xfs" --ondisk=vda --size=1024 --encrypted --passphrase=temppass
part /home --fstype="xfs" --ondisk=vda --size=2048 --grow --encrypted --
passphrase=temppass
```

```
part /var/log --fstype="xfs" --ondisk=vda --size=1024 --encrypted --passphrase=temppass
part /var/log/audit --fstype="xfs" --ondisk=vda --size=1024 --encrypted --
passphrase=temppass
```

2. **%packages** 섹션에 나열하여 관련 **Clevis** 패키지를 설치합니다.

```
%packages
clevis-dracut
%end
```

3. **%post** 섹션에서 바인딩을 수행하기 위해 **clevis luks bind** 를 호출합니다. 임시 암호를 삭제합니다.

```
%post
clevis luks bind -f -k- -d /dev/vda2 \
tang '{"url":"http://tang.srv","thp":"_OsIk0T-E2l6qjfdDiwVmidoZjA"}' \ <<< "temppass"
cryptsetup luksRemoveKey /dev/vda2 <<< "temppass"
%end
```

위의 예에서 **Tang** 서버에서 신뢰하는 지문을 바인딩 구성의 일부로 지정하여 완전히 비대화식으로 지정합니다.

Tang 서버 대신 **TPM 2.0** 정책을 사용할 때 유사한 절차를 사용할 수 있습니다.

Kickstart 설치에 대한 자세한 내용은 **Red Hat Enterprise Linux 7** 설치 가이드를 참조하십시오. **Linux Unified Key Setup-on-disk-format(LUKS)**에 대한 자세한 내용은 **4.9.1**절. “**LUKS** 디스크 암호화 사용” 에서 참조하십시오.

4.10.8. Removable 스토리지 장치의 자동 잠금 해제 구성

USB 드라이브와 같은 **LUKS**로 암호화된 이동식 스토리지 장치의 잠금을 자동으로 해제하려면 **clevis-udisks2** 패키지를 설치합니다.

```
~]# yum install clevis-udisks2
```

시스템을 재부팅한 다음, **4.10.6**절. “루트 볼륨의 수동 등록 구성” 에 설명된 대로 **clevis luks bind** 명령을 사용하여 바인딩 단계를 수행합니다. 예를 들면 다음과 같습니다.

```
~]# clevis luks bind -d /dev/sdb1 tang '{"url":"http://tang.srv"}'
```

이제 GNOME 데스크탑 세션에서 LUKS로 암호화된 이동식 장치의 잠금을 자동으로 해제할 수 있습니다. Clevis 정책에 바인딩된 장치는 clevis luks unlock 명령으로 잠금 해제할 수도 있습니다.

```
~]# clevis luks unlock -d /dev/sdb1
```

Tang 서버 대신 TPM 2.0 정책을 사용할 때 유사한 절차를 사용할 수 있습니다.

4.10.9. 부팅 시 루트가 아닌 볼륨의 자동 잠금 해제 구성

root가 LUKS로 암호화된 볼륨을 잠금 해제하려면 다음 단계를 수행합니다.

1. clevis-systemd 패키지를 설치합니다.

```
~]# yum install clevis-systemd
```

2. Clevis의 잠금 해제 서비스를 활성화합니다.

```
~]# systemctl enable clevis-luks-askpass.path
Created symlink from /etc/systemd/system/remote-fs.target.wants/clevis-luks-askpass.path to
/usr/lib/systemd/system/clevis-luks-askpass.path.
```

3. 4.10.6절. “루트 볼륨의 수동 등록 구성” 에 설명된 대로 clevis luks bind 명령을 사용하여 바인딩 단계를 수행합니다.

4. 시스템 부팅 중에 암호화된 블록 장치를 설정하려면 _netdev 옵션과 함께 해당 행을 /etc/crypttab 구성 파일에 추가합니다. 자세한 내용은 crypttab(5) 매뉴얼 페이지를 참조하십시오.

5. /etc/fstab 파일의 액세스 가능한 파일 시스템 목록에 볼륨을 추가합니다. 이 설정 파일에서 _netdev 옵션도 사용합니다. 자세한 내용은 fstab(5) 매뉴얼 페이지를 참조하십시오.

4.10.10. EgressIP 네트워크에서 가상 머신 배포

clevis luks bind 명령은 LUKS 마스터 키를 변경하지 않습니다. 가상 시스템 또는 클라우드 환경에서 사용할 LUKS 암호화된 이미지를 생성하는 경우 이 이미지를 실행하는 모든 인스턴스가 마스터 키를 공유합니다. 이는 매우 안전하지 않으며 항상 피해야 합니다.

이는 Clevis의 제한이 아니라 LUKS의 설계 원칙입니다. 클라우드에서 암호화된 루트 볼륨을 사용하려면 클라우드에서 Red Hat Enterprise Linux의 각 인스턴스에 대해 설치 프로세스(일반적으로 Kickstart 사용)를 수행해야 합니다. LUKS 마스터 키도 공유하지 않고 이미지를 공유할 수 없습니다.

가상화 환경에서 자동 잠금 해제를 배포하려면 Kickstart 파일(4.10.7절. “Kickstart를 사용하여 자동화된 등록 구성”참조) 또는 다른 자동화된 프로비저닝 툴과 함께 시스템을 사용하여 각 암호화된 VM에 고유한 마스터 키가 있는지 확인하는 것이 좋습니다.

4.10.11. DASD를 사용하여 클라우드 환경에 대해 자동으로 등록할 수 있는 VM 이미지 빌드

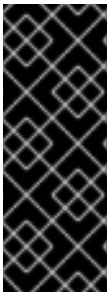
클라우드 환경에 자동으로 표시될 수 있는 암호화된 이미지를 배포하면 고유한 문제를 제공할 수 있습니다. 다른 가상화 환경과 마찬가지로 LUKS 마스터 키를 공유하지 않도록 단일 이미지에서 시작된 인스턴스 수를 줄이는 것이 좋습니다.

따라서 공용 리포지토리에서 공유되지 않고 제한된 인스턴스 배포에 대한 기반을 제공하는 사용자 지정 이미지를 생성하는 것이 좋습니다. 생성할 정확한 인스턴스 수는 배포의 보안 정책에 따라 정의되어야 하며 LUKS 마스터 키 공격 벡터와 관련된 위험 허용 오차를 기반으로 합니다.

LUKS 지원 자동 배포를 빌드하려면 Lorax 또는 virt-install과 같은 시스템을 Kickstart 파일과 함께 사용하여 이미지 빌드 프로세스 중 마스터 키의 고유성을 보장해야 합니다.

클라우드 환경에서는 여기에서 고려할 두 가지 Tang 서버 배포 옵션을 사용할 수 있습니다. 먼저 Tang 서버는 클라우드 환경 자체 내에 배포할 수 있습니다. 둘째, Tang 서버는 두 인프라 간에 VPN 링크를 사용하여 독립 인프라에 클라우드 외부에서 배포할 수 있습니다.

클라우드에서 기본적으로 Tang을 배포하면 쉽게 배포할 수 있습니다. 그러나 다른 시스템의 암호 텍스트와 인프라를 공유하면 Tang 서버의 개인 키와 Clevis 메타데이터가 동일한 물리적 디스크에 저장될 수 있습니다. 이 물리적 디스크에 대한 액세스 권한은 암호화 텍스트 데이터를 완전히 손상시킬 수 있습니다.



중요

이러한 이유로 Red Hat은 데이터가 저장된 위치와 Tang이 실행 중인 시스템 간에 물리적 분리를 유지할 것을 강력히 권장합니다. 클라우드와 Tang 서버를 분리하면 Tang 서버의 개인 키를 실수로 Clevis 메타데이터와 결합할 수 없습니다. 또한 클라우드 인프라가 위험한 경우 Tang 서버의 로컬 제어 기능을 제공합니다.

4.10.12. 추가 리소스

여러 LUKS 장치(Clevis+Tang 잠금 해제) 지식 베이스 문서를 사용하여 네트워크 Bound 디스크 암호화를 설정하는 방법 문서.

자세한 내용은 다음 도움말 페이지를 참조하십시오.

- *tang(8)*
- *clevis(1)*
- *jose(1)*
- *clevis-luks-unlockers(1)*
- *tang-nagios(1)*

4.11. AIDE로 무결성 확인

AIDE(Advanced Intrusion Detection Environment)는 시스템에 파일 데이터베이스를 생성한 다음 해당 데이터베이스를 사용하여 파일 무결성을 확인하고 시스템 침입을 탐지하는 유틸리티입니다.

4.11.1. AIDE설치

aide 패키지를 설치하려면 root 로 다음 명령을 입력합니다.

```
~]# yum install aide
```

초기 데이터베이스를 생성하려면 root 로 다음 명령을 입력합니다.

```
~]# aide --init

AIDE, version 0.15.1

### AIDE database at /var/lib/aide/aide.db.new.gz initialized.
```

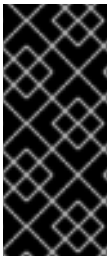
참고

기본 구성에서 **aide --init** 명령은 **/etc/aide.conf** 파일에 정의된 디렉토리와 파일 집합만 확인합니다. **AIDE** 데이터베이스에 추가 디렉터리 또는 파일을 포함하고 감시된 매개변수를 변경하려면 그에 따라 **/etc/aide.conf** 를 편집합니다.

데이터베이스 사용을 시작하려면 초기 데이터베이스 파일 이름에서 **.new** 하위 문자열을 제거합니다.

```
~]# mv /var/lib/aide/aide.db.new.gz /var/lib/aide/aide.db.gz
```

AIDE 데이터베이스의 위치를 변경하려면 **/etc/aide.conf** 파일을 편집하고 **DBDIR** 값을 수정합니다. 보안을 강화하기 위해 데이터베이스, 구성 및 **/usr/sbin/aide** 바이너리 파일을 읽기 전용 미디어와 같은 보안 위치에 저장하십시오.



중요

AIDE 데이터베이스 위치가 변경된 후 **SELinux** 거부를 방지하려면 그에 따라 **SELinux** 정책을 업데이트합니다. 자세한 내용은 **SELinux** 사용자 및 관리자 가이드를 참조하십시오.

4.11.2. 무결성 검사 수행

수동 점검을 시작하려면 **root** 로 다음 명령을 입력합니다.

```
~]# aide --check
AIDE 0.15.1 found differences between database and filesystem!!
Start timestamp: 2017-03-30 14:12:56

Summary:
  Total number of files: 147173
  Added files:   1
  Removed files:  0
  Changed files:  2
...
```

최소한 **AIDE** 는 매주 검사를 실행하도록 구성해야 합니다. 대부분의 **AIDE** 는 매일 실행해야 합니다. 예를 들어, **cron** 을 사용하여 **4:05** 에서 매일 **AIDE** 실행을 예약하려면 (시스템 관리자 가이드의 시스템 작업 자동화 장 참조) **/etc/crontab** 에 다음 행을 추가하십시오.

```
05 4 * * * root /usr/sbin/aide --check
```

4.11.3. AIDE 데이터베이스 업데이트

패키지 업데이트 또는 구성 파일 조정과 같은 시스템 변경 사항을 확인한 후 기준 **AIDE** 데이터베이스를 업데이트합니다.

```
~]# aide --update
```

aide --update 명령은 **/var/lib/aide/aide.db.new.gz** 데이터베이스 파일을 만듭니다. 무결성 검사를 위해 사용을 시작하려면 파일 이름에서 **.new** 부분 문자열을 제거합니다.

4.11.4. 추가 리소스

AIDE에 대한 자세한 내용은 다음 문서를 참조하십시오.

- **AIDE(1)** 도움말 페이지
- **aide.conf(5) man page**
- *Red Hat Enterprise Linux 7의 보안 구성 가이드(OpenSCAP 보안 가이드): AIDE로 무결성 확인*

4.12. USBGUARD사용

USBGuard 소프트웨어 프레임워크는 장치 속성을 기반으로 기본 화이트리스트 및 블랙리스트 기능을 구현하여 개입하는 **USB** 장치에 대한 시스템 보호 기능을 제공합니다. 사용자 정의 정책을 적용하기 위해 **USBGuard** 는 **Linux** 커널 **USB** 장치 권한 부여 기능을 사용합니다. **USBGuard** 프레임워크는 다음 구성 요소를 제공합니다.

- 동적 상호 작용 및 정책 적용을 위한 프로세스 간 통신(**IPC**) 인터페이스가 있는 데몬 구성 요소입니다.
- 실행 중인 **USBGuard** 인스턴스와 상호 작용하는 명령줄 인터페이스입니다.
- **USB** 장치 권한 부여 정책을 작성하는 규칙 언어입니다.

- 공유 라이브러리에 구현된 데몬 구성 요소와 상호 작용하기 위한 **C++ API**입니다.

4.12.1. USBGuard설치

usbguard 패키지를 설치하려면 **root** 로 다음 명령을 입력합니다.

```
~]# yum install usbguard
```

초기 규칙 세트를 생성하려면 다음 명령을 **root** 로 입력합니다.

```
~]# usbguard generate-policy > /etc/usbguard/rules.conf
```

참고

USBGuard 규칙 세트를 사용자 지정하려면 **/etc/usbguard/rules.conf** 파일을 편집합니다. 자세한 내용은 **usbguard-rules.conf(5)** 매뉴얼 페이지를 참조하십시오. 자세한 내용은 **4.12.3**절. “규칙 언어를 사용하여 고유한 정책 만들기” 을 참조하십시오.

USBGuard 데몬을 시작하려면 **root** 로 다음 명령을 입력합니다.

```
~]# systemctl start usbguard.service
~]# systemctl status usbguard
● usbguard.service - USBGuard daemon
   Loaded: loaded (/usr/lib/systemd/system/usbguard.service; disabled; vendor preset: disabled)
   Active: active (running) since Tue 2017-06-06 13:29:31 CEST; 9s ago
     Docs: man:usbguard-daemon(8)
 Main PID: 4984 (usbguard-daemon)
   CGroup: /system.slice/usbguard.service
           └─4984 /usr/sbin/usbguard-daemon -k -c /etc/usbguard/usbguard-daem...
```

USBGuard 가 시스템을 시작할 때 자동으로 시작되도록 하려면 **root** 로 다음 명령을 사용합니다.

```
~]# systemctl enable usbguard.service
Created symlink from /etc/systemd/system/basic.target.wants/usbguard.service to
/usr/lib/systemd/system/usbguard.service.
```

USBGuard 에서 인식하는 모든 **USB** 장치를 나열하려면 **root** 로 다음 명령을 입력합니다.

```
~]# usbguard list-devices
```

```
1: allow id 1d6b:0002 serial "0000:00:06.7" name "EHCI Host Controller" hash
"JDOb0BiktYs2ct3mSQKopnOOV2h9MGYADwhT+oUtF2s=" parent-hash
"4PHGcaDKWtPjKDwYpIRG722cB9SlGz9l9Iea93+Gt9c=" via-port "usb1" with-interface 09:00:00
...
6: block id 1b1c:1ab1 serial "000024937962" name "Voyager" hash
"CrXgiaWIf2bZAU+5WkzOE7y0rdSO82XMzubn7HDb95Q=" parent-hash
"JDOb0BiktYs2ct3mSQKopnOOV2h9MGYADwhT+oUtF2s=" via-port "1-3" with-interface 08:06:50
```

*시스템과 상호 작용하도록 장치를 인증하려면 **allow-device** 옵션을 사용합니다.*

```
~]# usbguard allow-device 6
```

*장치를 인증 해제하고 시스템에서 제거하려면 **reject-device** 옵션을 사용합니다. 장치의 인증을 해제하려면 **block-device** 옵션과 함께 **usbguard** 명령을 사용합니다.*

```
~]# usbguard block-device 6
```

***usbguard** 는 다음 의미와 함께 블록 및 거부 용어를 사용합니다.*

- ***Block -** 지금은 이 장치와 연결하지 마십시오.*
- ***reject -** 이 장치가 존재하지 않는 것처럼 무시합니다.*

***usbguard** 명령의 모든 옵션을 보려면 **--help** 지시문으로 입력합니다.*

```
~]$ usbguard --help
```

4.12.2. 백색 목록 및 블랙리스트 생성

***usbguard-daemon.conf** 파일은 명령줄 옵션을 구문 분석한 후 **usbguard** 데몬에 의해 로드되고 데몬의 런타임 매개 변수를 구성하는 데 사용됩니다. 기본 구성 파일(**/etc/usbguard/usbguard-daemon.conf**)을 재정의하려면 **-c** 명령줄 옵션을 사용합니다. 자세한 내용은 **usbguard-daemon(8)** 매뉴얼 페이지를 참조하십시오.*

*화이트 목록 또는 검정 목록을 만들려면 **usbguard-daemon.conf** 파일을 편집하고 다음 옵션을 사용합니다.*

usbguard 구성 파일

RuleFile=<path>

usbguard 데몬은 이 파일을 사용하여 에서 정책 규칙 세트를 로드하고 IPC 인터페이스를 통해 수신한 새 규칙을 작성합니다.

IPCAllowedUsers=<username> [<username> ...]

데몬이 IPC 연결을 수락할 공백으로 구분된 사용자 이름의 목록입니다.

IPCAllowedGroups=<groupname> [<groupname> ...]

데몬이 IPC 연결을 수락할 공백으로 구분된 그룹 이름의 목록입니다.

IPCAccessControlFiles=<path>

IPC 액세스 제어 파일을 포함하는 디렉터리의 경로입니다.

ImplicitPolicyTarget=<target>

정책의 규칙과 일치하지 않는 장치를 처리하는 방법. 허용되는 값: allow, block, reject.

PresentDevicePolicy=<policy>

데몬이 시작될 때 이미 연결된 장치를 처리하는 방법:

- *허용 - 모든 현재 장치 인증*
- *block - 현재 모든 장치 인증 해제*
- *거부 - 현재 장치 제거*
- *계속 - 내부 상태를 동기화하고 그대로 둡니다.*

- ***apply-policy -*** *모든 현재 장치의 규칙 세트를 평가합니다.*

PresentControllerPolicy=<policy>

데몬이 시작될 때 이미 연결된 ***USB*** *컨트롤러를 처리하는 방법:*

- *허용 -* ***모든 현재 장치 인증***
- ***block -*** *현재 모든 장치 인증 해제*
- *거부 -* ***현재 장치 제거***
- *계속 -* ***내부 상태를 동기화하고 그대로 둡니다.***
- ***apply-policy -*** *모든 현재 장치의 규칙 세트를 평가합니다.*

예 4.5. usbguard *구성*

다음 구성 파일은 ***usbguard*** *데몬을 구성하여* ***/etc/usbguard/rules.conf*** *파일에서 규칙을 로드하고* ***usbguard*** *그룹의 사용자만* ***IPC*** *인터페이스를 사용할 수 있습니다.*

```
RuleFile=/etc/usbguard/rules.conf
IPCAccessControlFiles=/etc/usbguard/IPCAccessControl.d/
```

IPC ACL(*액세스 제어 목록)을 지정하려면* ***usbguard add-user*** *또는* ***usbguard remove-user*** *명령을 사용합니다. 자세한 내용은* ***usbguard(1)*** *를 참조하십시오. 이 예에서* ***usbguard*** *그룹의 사용자가* ***USB*** *장치 권한 부여 상태를 수정하고,* ***USB*** *장치를 나열하고, 예외 이벤트를 수신하고,* ***USB*** *권한 부여 정책을 나열하도록 허용하려면 다음 명령을* ***root*** *로 입력합니다.*

```
~]# usbguard add-user -g usbguard --devices=modify,list,listen --policy=list --exceptions=listen
```

중요

데몬은 **USBGuard** 공용 **IPC** 인터페이스를 제공합니다. **Red Hat Enterprise Linux**에서 이 인터페이스에 대한 액세스는 기본적으로 **root** 사용자로만 제한됩니다. **IPCAccessControlFiles** 옵션(권장) 또는 **IPCAllowedUsers** 및 **IPCAllowedGroups** 옵션을 설정하여 **IPC** 인터페이스에 대한 액세스를 제한하는 것이 좋습니다. 이렇게 하면 **IPC** 인터페이스가 모든 로컬 사용자에게 노출되므로 **ACL**이 구성되지 않은 상태로 두지 않고 **USB** 장치의 권한 부여 상태를 조작하고 **USBGuard** 정책을 수정할 수 있습니다.

자세한 내용은 **usbguard-daemon.conf(5)** 도움말 페이지의 **IPC** 액세스 제어 섹션을 참조하십시오.

4.12.3. 규칙 언어를 사용하여 고유한 정책 만들기

usbguard 데몬은 일련의 규칙에 의해 정의된 정책을 기반으로 **USB** 장치를 인증할지 여부를 결정합니다. **USB** 장치가 시스템에 삽입되면 데몬은 기존 규칙을 순차적으로 검색하고 일치하는 규칙이 있는 경우 규칙 타겟을 기반으로 장치를 승인(허용), 암호 해독(차단) 또는 제거(재정)합니다. 일치하는 규칙이 없으면 암시적 기본 대상을 기반으로 결정됩니다. 이 암시적인 기본값은 사용자가 결정할 때까지 장치를 차단하는 것입니다.

언어 문법은 다음과 같습니다.

```
rule ::= target device_id device_attributes conditions.

target ::= "allow" | "block" | "reject".

device_id ::= "*:*" | vendor_id ":*" | vendor_id ":" product_id.

device_attributes ::= device_attributes | attribute.
device_attributes ::= .

conditions ::= conditions | condition.
conditions ::= .
```

대상, 장치 사양 또는 장치 속성과 같은 규칙 언어에 대한 자세한 내용은 **usbguard-rules.conf(5)** 매뉴얼 페이지를 참조하십시오.

예 4.6. usbguard 예제 정책

USB 대용량 스토리지 장치를 허용하고 다른 모든 것을 차단하십시오.

*이 정책은 대용량 저장 장치가 아닌 모든 장치를 차단합니다. **USB** 플래쉬 디스크에 숨겨진 키보드 인터페이스가 있는 장치가 차단됩니다. 단일 대용량 스토리지 인터페이스가 있는 장치만 운영 체제와 상호 작용할 수 있습니다. 정책은 단일 규칙으로 구성됩니다.*

```
allow with-interface equals { 08:*:* }
```

*차단 규칙이 없기 때문에 차단 규칙이 암시적입니다. 암시적 차단은 **USBGuard** 이벤트를 수신하는 데스크탑 애플릿이 암시적 타겟이 장치에 대해 선택된 경우 결정을 요청할 수 있기 때문에 데스크탑 사용자에게 유용합니다.*

*특정 **Yubikey** 장치가 특정 포트를 통해 연결되도록 허용*

해당 포트의 다른 모든 항목을 거부합니다.

```
allow 1050:0011 name "Yubico Yubikey II" serial "0001234567" via-port "1-2" hash
"044b5e168d40ee0245478416caf3d998"
reject via-port "1-2"
```

인터페이스의 의심스러운 조합이 있는 장치 거부

*키보드 또는 네트워크 인터페이스를 구현하는 **USB** 플래시 디스크는 매우 의심스럽습니다. 다음 규칙 세트는 **USB** 플래시 디스크를 허용하고 추가 및 의심스런 인터페이스가 있는 장치를 명시적으로 거부하는 정책을 형성합니다.*

```
allow with-interface equals { 08:*:* }
reject with-interface all-of { 08:*:* 03:00:* }
reject with-interface all-of { 08:*:* 03:01:* }
reject with-interface all-of { 08:*:* e0:*:* }
reject with-interface all-of { 08:*:* 02:*:* }
```

참고

*블랙리스트는 잘못된 접근 방식이며 장치 세트를 블랙리스트에 등록하고 나머지는 허용해서는 안 됩니다. 위의 정책은 차단이 암시적인 기본값이라고 가정합니다. **"bad"**로 간주되는 장치 세트를 거부하는 것은 시스템의 노출을 가능한 한 이러한 장치로 제한하는 좋은 방법입니다.*

*키보드 전용 **USB** 장치 허용*

*다음 규칙은 키보드 인터페이스가 이미 허용되는 **USB** 장치가 없는 경우에만 키보드 전용 **USB** 장치를 허용합니다.*

```
allow with-interface one-of { 03:00:01 03:01:01 } if !allowed-matches(with-interface one-of {
03:00:01 03:01:01 })
```

usbguard generate-policy 명령을 사용하는 초기 정책 생성 후 **/etc/usbguard/rules.conf** 를 편집하여 **USBGuard** 정책 규칙을 사용자 지정합니다.

```
~]$ usbguard generate-policy > rules.conf
~]$ vim rules.conf
```

업데이트된 정책을 설치하고 변경 사항을 적용하려면 다음 명령을 사용하십시오.

```
~]# install -m 0600 -o root -g root rules.conf /etc/usbguard/rules.conf
```

4.12.4. 추가 리소스

USBGuard 에 대한 자세한 내용은 다음 문서를 참조하십시오.

- **usbguard(1)** 도움말 페이지
- **usbguard-rules.conf(5) man page**
- **usbguard-daemon(8)** 도움말 페이지
- **usbguard-daemon.conf(5) man page**
- **USBGuard** 홈페이지

4.13. TLS 설정 강화

TLS (Transport Layer Security)는 네트워크 통신을 보호하는 데 사용되는 암호화 프로토콜입니다. 기본 키 교환 프로토콜,인증 방법 및 암호화 알고리즘 을 구성하여 시스템 보안 설정을 강화할 때 지원되는 클라이언트의 범위를 넓히면 결과적으로 보안 수준이 낮아집니다. 반대로 엄격한 보안 설정은 클라이언트와의 호환성이 제한되어 일부 사용자가 시스템에서 잠길 수 있습니다. 가장 엄격한 사용 가능한 구성을 대상으로 하고 호환성을 위해 필요한 경우에만 완화해야 합니다.

Red Hat Enterprise Linux 7에 포함된 라이브러리에서 제공하는 기본 설정은 대부분의 배포에 충분히 안전합니다. TLS 구현은 기존 클라이언트 또는 서버에서의 연결을 방지하지 못하는 동시에 보안 알고리즘을 사용합니다. 보안 알고리즘 또는 프로토콜을 지원하지 않거나 연결할 수 없는 기존 클라이언트 또는 서버를 지원하지 않는 엄격한 보안 요구 사항이 있는 환경에서 이 섹션에 설명된 강화된 설정을 적용합니다.

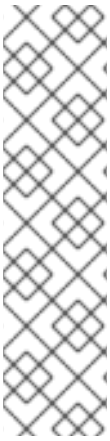
4.13.1. 사용할 알고리즘 선택

선택 및 구성해야 하는 여러 구성 요소가 있습니다. 다음 각각은 결과 구성의 견고성(및, 클라이언트의 지원 수준) 또는 솔루션이 시스템에 있는 컴퓨팅 요구 사항에 직접적인 영향을 미칩니다.

프로토콜 버전

최신 버전의 TLS 는 최상의 보안 메커니즘을 제공합니다. 이전 버전의 TLS (또는 SSL)에 대한 지원을 포함해야 하는 강력한 이유가 없는 경우 시스템에서 최신 버전의 TLS 를 사용하여 연결을 협상할 수 있습니다.

SSL 버전 2 또는 3을 사용한 협상은 허용하지 마십시오. 이러한 두 버전에는 심각한 보안 취약점이 있습니다. TLS 버전 1.0 이상을 사용하는 협상만 허용합니다. 현재 TLS, 1.2를 사용하는 것이 좋습니다.



참고

현재 모든 TLS 버전의 보안은 TLS 확장, 특정 암호(아래 참조) 및 기타 해결 방법에 따라 다릅니다. 모든 TLS 연결 피어는 보안 재협상 표시(RFC5746)를 구현해야 하며 압축을 지원하지 않아야 하며 CBC-mode 암호(Lucky Thirteen 공격)에 대한 타이밍 공격에 대한 완화 조치를 구현해야 합니다. TLS 1.0 클라이언트는 레코드 분할을 추가로 구현해야 합니다(BEAST 공격에 대한 해결 방법). TLS 1.2 에서는 알려진 문제가 없는 AES-GCM,AES-CCM 또는 Camellia-GCM 와 같은 관련 데이터 (AEAD) 모드 암호를 사용하여 인증된 암호화를 지원합니다. 언급된 모든 완화 조치는 Red Hat Enterprise Linux에 포함된 암호화 라이브러리에서 구현됩니다.

프로토콜 버전 및 권장 사용에 대한 간략한 개요는 표 4.6. “프로토콜 버전” 을 참조하십시오.

표 4.6. 프로토콜 버전

프로토콜 버전	사용 권장 사항

프로토콜 버전	사용 권장 사항
SSL v2	사용하지 마십시오. 심각한 보안 취약점이 있습니다.
SSL v3	사용하지 마십시오. 심각한 보안 취약점이 있습니다.
TLS 1.0	필요한 경우 상호 운용성을 목적으로 사용합니다. 상호 운용성을 보장하는 방식으로 완화할 수 없는 알려진 문제가 있어 완화 조치가 기본적으로 활성화되어 있지 않습니다. 최신 암호화 제품군을 지원하지 않습니다.
TLS 1.1	필요한 경우 상호 운용성을 목적으로 사용합니다. 알려진 문제는 없지만 **Red Hat Enterprise Linux**의 모든 **TLS** 구현에 포함된 프로토콜 수정에 의존합니다. 최신 암호화 제품군을 지원하지 않습니다.
TLS 1.2	권장 버전입니다. 최신 **AEAD** 암호화 제품군을 지원합니다.

Red Hat Enterprise Linux의 일부 구성 요소는 **TLS 1.1** 또는 **1.2** 를 지원하더라도 **TLS 1.0** 을 사용하도록 구성되어 있습니다. 이는 최신 버전의 **TLS** 를 지원하지 않을 수 있는 외부 서비스와 가장 높은 수준의 상호 운용성을 달성하기 위한 시도에 의해 촉진됩니다. 상호 운용성 요구 사항에 따라 사용 가능한 최고 버전의 **TLS** 를 활성화합니다.

중요

SSL v3 는 사용하지 않는 것이 좋습니다. 그러나 일반적인 사용을 위해 안전하지 않고 적합하지 않은 것으로 간주되는 경우에도 **SSL v3** 을 계속 활성화해야 하는 경우 **4.8절. "stunnel 사용"** 에서 암호화를 지원하지 않는 서비스를 사용하는 경우에도 **stunnel** 을 사용하여 통신을 안전하게 암호화하는 방법에 대한 지침은 에서 사용되지 않고 안전하지 않은 암호화 모드를 사용할 수 있는 경우에만 사용할 수 있는 방법에 대한 지침을 참조하십시오.

암호화 제품군

최신의 보안 암호 제품군은 안전하지 않은 오래 된 암호 제품군을 선호해야 합니다. 항상 암호화 또는 인증을 전혀 제공하지 않는 **eNULL** 및 **aNULL** 암호화 제품군 사용을 비활성화합니다. 가능한 경우 중요한 단점을 가진 **RC4** 또는 **HMAC-md5** 기반 암호화 제품군도 비활성화해야 합니다. 동일한 이름이 저용인 내보내기 암호 모음에 적용되고, 이는 의도적으로 설정되었기 때문에 쉽게 중단할 수 있습니다.

즉시 안전하지는 않지만 짧은 유용한 수명 동안 **128** 비트의 보안을 제공하는 암호화 제품군은 고려해서는 안 됩니다. **128**비트의 보안을 사용하는 알고리즘은 적어도 몇 년 동안 분리할 수 없을 것으로 예상되므로 적극 권장됩니다. **3DES** 암호가 **168** 비트의 사용을 알리는 반면, 실제로 **112** 비트의 보안을 제공합니다.

서버 키가 손상된 경우에도 암호화된 데이터의 기밀성(**PFS**) 을 지원하는 암호화 제품군에 항상 우선순위를 부여합니다. 이 규칙은 빠른 **RSA** 키 교환에서 제외되지만 **ECDHE** 와 **DHE** 를 사용할 수 있습니다. 두 가지 중에서 **ECDHE** 가 더 빠르므로 선호하는 선택이 더 빠릅니다.

또한 **CBC-mode** 암호를 패딩 또는**acle** 공격에 취약하지 않으므로 **AEAD** 암호(예: **AES-GCM**)에 우선순위를 부여해야 합니다. 또한 대부분의 경우 **AES -GCM** 는 특히 하드웨어에 **AES** 에 대한 암호화 가속기가 있는 경우 **CBC** 모드보다 빠릅니다.

ECDSA 인증서와 함께 **ECDHE** 키 교환을 사용하는 경우 트랜잭션이 순수 **RSA** 키 교환보다 훨씬 빠릅니다. 레거시 클라이언트에 대한 지원을 제공하기 위해 서버에 두 개의 인증서와 키 쌍을 설치할 수 있습니다. 하나는 **ECDSA** 키(새 클라이언트용)와 **RSA** 키(기존 키용)와 함께 설치합니다.

공개 키 길이

RSA 키를 사용하는 경우 항상 최소 **SHA-256**에서 서명된 최소 **3072**비트의 키 길이를 선호하며, 이 키 길이는 **true 128**비트의 보안에 충분히 큽니다.

주의

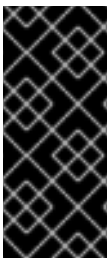
시스템의 보안은 체인에서 가장 약한 링크만큼 강력합니다. 예를 들어 강력한 암호만으로는 좋은 보안이 보장되지 않습니다. 키와 인증서는CA(인증 기관)에서 키에 서명하는 데 사용하는 해시 함수 및 키뿐만 아니라 중요합니다.

4.13.2. TLS 구현 사용

Red Hat Enterprise Linux 7은 다양한 완전한 기능을 갖춘 **TLS** 구현과 함께 배포됩니다. 이 섹션에서는 **OpenSSL** 및 **GnuTLS** 의 구성에 대해 설명합니다. 개별 애플리케이션에서 **TLS** 지원을 구성하는 방법에 대한 지침은 **4.13.3**절. “특정 애플리케이션 구성” 을 참조하십시오.

사용 가능한 **TLS** 구현에서는 **TLS-secured** 통신을 설정하고 사용할 때 함께 제공되는 모든 요소를 정의하는 다양한 암호화 제품군에 대한 지원을 제공합니다.

다양한 구현에 포함된 툴을 사용하여 **4.13.1**절. “사용할 알고리즘 선택” 에 설명된 권장 사항을 고려하는 동안 사용 사례에 가장 적합한 보안을 제공하는 암호화 제품군을 나열하고 지정합니다. 그런 다음 결과 암호화 제품군을 사용하여 개별 애플리케이션의 협상 및 보안 연결을 구성할 수 있습니다.



중요

*사용하는 **TLS** 구현 또는 해당 구현을 사용하는 애플리케이션을 업그레이드할 때마다 설정을 확인하십시오. 새로운 버전에서는 활성화할 필요가 없고 현재 구성이 비활성화되지 않는 새로운 암호화 제품군을 도입할 수 있습니다.*

4.13.2.1. OpenSSL에서 Cipher Suite 작업

OpenSSL 은 **SSL** 및 **TLS** 프로토콜을 지원하는 툴킷 및 암호화 라이브러리입니다. **Red Hat Enterprise Linux 7**에서 구성 파일은 **/etc/pki/tls/openssl.cnf** 에 제공됩니다. 이 구성 파일의 형식은 **config(1)** 에 설명되어 있습니다. **4.7.9**절. “OpenSSL 구성” 도 참조하십시오.

OpenSSL 설치에서 지원하는 모든 암호화 제품군 목록을 가져오려면 다음과 같이 **openssl** 명령을 **ciphers** 하위 명령과 함께 사용하십시오.

```
~]$ openssl ciphers -v 'ALL:COMPLEMENTOFALL'
```

기타 매개 변수(**OpenSSL** 문서의 암호화 문자열 및 키워드 로 참조)를 **ciphers** 하위 명령에 전달하여 출력을 좁힙니다. 특수 키워드를 사용하여 특정 조건을 충족하는 모음만 나열할 수 있습니다. 예를 들어 **HIGH** 그룹에 속하는 것으로 정의된 제품군만 나열하려면 다음 명령을 사용합니다.

```
~]$ openssl ciphers -v 'HIGH'
```

사용 가능한 키워드 및 암호 문자열 목록은 **ciphers(1)** 매뉴얼 페이지를 참조하십시오.

4.13.1절. "사용할 알고리즘 선택" 에 설명된 권장 사항을 충족하는 암호화 제품군 목록을 가져오려면 다음과 유사한 명령을 사용합니다.

```
~]$ openssl ciphers -v 'kEECDH+aECDSA+AES:kEECDH+AES+aRSA:kEDH+aRSA+AES' |
column -t
ECDHE-ECDSA-AES256-GCM-SHA384  TLSv1.2  Kx=ECDH  Au=ECDSA  Enc=AESGCM(256)
Mac=AEAD
ECDHE-ECDSA-AES256-SHA384      TLSv1.2  Kx=ECDH  Au=ECDSA  Enc=AES(256)
Mac=SHA384
ECDHE-ECDSA-AES256-SHA         SSLv3    Kx=ECDH  Au=ECDSA  Enc=AES(256)    Mac=SHA1
ECDHE-ECDSA-AES128-GCM-SHA256  TLSv1.2  Kx=ECDH  Au=ECDSA  Enc=AESGCM(128)
Mac=AEAD
ECDHE-ECDSA-AES128-SHA256      TLSv1.2  Kx=ECDH  Au=ECDSA  Enc=AES(128)
Mac=SHA256
ECDHE-ECDSA-AES128-SHA         SSLv3    Kx=ECDH  Au=ECDSA  Enc=AES(128)    Mac=SHA1
ECDHE-RSA-AES256-GCM-SHA384    TLSv1.2  Kx=ECDH  Au=RSA    Enc=AESGCM(256)
Mac=AEAD
ECDHE-RSA-AES256-SHA384        TLSv1.2  Kx=ECDH  Au=RSA    Enc=AES(256)    Mac=SHA384
ECDHE-RSA-AES256-SHA           SSLv3    Kx=ECDH  Au=RSA    Enc=AES(256)    Mac=SHA1
ECDHE-RSA-AES128-GCM-SHA256    TLSv1.2  Kx=ECDH  Au=RSA    Enc=AESGCM(128)
Mac=AEAD
ECDHE-RSA-AES128-SHA256        TLSv1.2  Kx=ECDH  Au=RSA    Enc=AES(128)    Mac=SHA256
ECDHE-RSA-AES128-SHA           SSLv3    Kx=ECDH  Au=RSA    Enc=AES(128)    Mac=SHA1
DHE-RSA-AES256-GCM-SHA384      TLSv1.2  Kx=DH    Au=RSA    Enc=AESGCM(256)
Mac=AEAD
DHE-RSA-AES256-SHA256          TLSv1.2  Kx=DH    Au=RSA    Enc=AES(256)    Mac=SHA256
DHE-RSA-AES256-SHA             SSLv3    Kx=DH    Au=RSA    Enc=AES(256)    Mac=SHA1
DHE-RSA-AES128-GCM-SHA256      TLSv1.2  Kx=DH    Au=RSA    Enc=AESGCM(128)
Mac=AEAD
DHE-RSA-AES128-SHA256          TLSv1.2  Kx=DH    Au=RSA    Enc=AES(128)    Mac=SHA256
DHE-RSA-AES128-SHA             SSLv3    Kx=DH    Au=RSA    Enc=AES(128)    Mac=SHA1
```

위의 명령은 모든 비보안 암호를 생략하고 임시 **elliptic** 곡선 **Diffie-Hellman** 키 교환 및 **ECDSA** 암호를 우선적으로 제공하며 **RSA** 키 교환(전자 보안을 완벽하게보장)을 생략합니다.

이는 다소 엄격한 구성이며 광범위한 클라이언트와의 호환성을 허용하기 위해 실제 시나리오에서 조건을 완화해야 할 수 있습니다.

4.13.2.2. GnuTLS에서 Cipher Suite 작업

GnuTLS 는 **SSL** 및 **TLS** 프로토콜 및 관련 기술을 구현하는 통신 라이브러리입니다.

참고

Red Hat Enterprise Linux 7에 대한 **GnuTLS** 설치는 대부분의 사용 사례에 충분한 보안을 제공하는 최적의 기본 구성 값을 제공합니다. 특수 보안 요구 사항을 충족할 필요가 없는 경우 제공된 기본값을 사용하는 것이 좋습니다.

-l (또는 **--list**) 옵션과 함께 **gnutls-cli** 명령을 사용하여 지원되는 모든 암호화 제품군을 나열합니다.

```
~]$ gnutls-cli -l
```

l 옵션으로 표시되는 암호화 제품군 목록을 축소하려면 하나 이상의 매개변수(**GnuTLS** 문서의 우선 순위 문자열 및 키워드 로 참조)를 **--priority** 옵션에 전달합니다. 사용 가능한 모든 우선 순위 문자열 목록은 **http://www.gnutls.org/manual/gnutls.html#Priority-Strings** 의 **GnuTLS** 설명서를 참조하십시오. 예를 들어 다음 명령을 실행하여 최소 **128**비트의 보안을 제공하는 암호화 제품군 목록을 가져옵니다.

```
~]$ gnutls-cli --priority SECURE128 -l
```

4.13.1절. “사용할 알고리즘 선택” 에 설명된 권장 사항을 충족하는 암호화 제품군 목록을 가져오려면 다음과 유사한 명령을 사용합니다.

```
~]$ gnutls-cli --priority SECURE256:+SECURE128:-VERS-TLS-ALL:+VERS-TLS1.2:-RSA:-DHE-
DSS:-CAMELLIA-128-CBC:-CAMELLIA-256-CBC -l
Cipher suites for SECURE256:+SECURE128:-VERS-TLS-ALL:+VERS-TLS1.2:-RSA:-DHE-DSS:-
CAMELLIA-128-CBC:-CAMELLIA-256-CBC
TLS_ECDHE_ECDSA_AES_256_GCM_SHA384                0xc0, 0x2c      TLS1.2
TLS_ECDHE_ECDSA_AES_256_CBC_SHA384                0xc0, 0x24      TLS1.2
TLS_ECDHE_ECDSA_AES_256_CBC_SHA1                  0xc0, 0x0a      SSL3.0
TLS_ECDHE_ECDSA_AES_128_GCM_SHA256                0xc0, 0x2b      TLS1.2
TLS_ECDHE_ECDSA_AES_128_CBC_SHA256                0xc0, 0x23      TLS1.2
TLS_ECDHE_ECDSA_AES_128_CBC_SHA1                  0xc0, 0x09      SSL3.0
TLS_ECDHE_RSA_AES_256_GCM_SHA384                  0xc0, 0x30      TLS1.2
TLS_ECDHE_RSA_AES_256_CBC_SHA1                    0xc0, 0x14      SSL3.0
TLS_ECDHE_RSA_AES_128_GCM_SHA256                  0xc0, 0x2f      TLS1.2
TLS_ECDHE_RSA_AES_128_CBC_SHA256                  0xc0, 0x27      TLS1.2
TLS_ECDHE_RSA_AES_128_CBC_SHA1                    0xc0, 0x13      SSL3.0
TLS_DHE_RSA_AES_256_CBC_SHA256                    0x00, 0x6b      TLS1.2
TLS_DHE_RSA_AES_256_CBC_SHA1                      0x00, 0x39      SSL3.0
TLS_DHE_RSA_AES_128_GCM_SHA256                    0x00, 0x9e      TLS1.2
TLS_DHE_RSA_AES_128_CBC_SHA256                    0x00, 0x67      TLS1.2
TLS_DHE_RSA_AES_128_CBC_SHA1                      0x00, 0x33      SSL3.0
```

```
Certificate types: CTYPE-X.509
Protocols: VERS-TLS1.2
Compression: COMP-NULL
Elliptic curves: CURVE-SECP384R1, CURVE-SECP521R1, CURVE-SECP256R1
PK-signatures: SIGN-RSA-SHA384, SIGN-ECDSA-SHA384, SIGN-RSA-SHA512, SIGN-ECDSA-
SHA512, SIGN-RSA-SHA256, SIGN-DSA-SHA256, SIGN-ECDSA-SHA256
```

위의 명령은 최소 **128**비트의 보안이 설정된 암호를 사용하여 출력을 암호로 제한하고 강력한 보안에 우선순위를 부여합니다. 또한 **RSA** 키 교환 및 **DSS** 인증도 금지합니다.

이는 다소 엄격한 구성이며 광범위한 클라이언트와의 호환성을 허용하기 위해 실제 시나리오에서 조건을 완화해야 할 수 있습니다.

4.13.3. 특정 애플리케이션 구성

다양한 애플리케이션에서 **TLS** 에 대한 자체 구성 메커니즘을 제공합니다. 이 섹션에서는 가장 일반적으로 사용되는 서버 애플리케이션에서 사용하는 **TLS**관련 구성 파일에 대해 설명하고 일반적인 구성의 예를 제공합니다.

사용하도록 선택한 구성에 관계없이 항상 서버 애플리케이션에서 서버 측 암호 순서를 적용해야 합니다. 사용할 암호화 제품군은 구성하는 순서에 따라 결정됩니다.

4.13.3.1. Apache HTTP 서버 구성

Apache HTTP 서버는 **TLS** 요구 사항에 **OpenSSL** 및 **NSS** 라이브러리를 모두 사용할 수 있습니다. **TLS** 라이브러리 선택 사항에 따라 **mod_ssl** 또는 **mod_nss** 모듈**(eponymous** 패키지에서 제공됨**)**을 설치해야 합니다. 예를 들어 **OpenSSL mod_ssl** 모듈을 제공하는 패키지를 설치하려면 **root**로 다음 명령을 실행합니다.

```
~]# yum install mod_ssl
```

mod_ssl 패키지는 **Apache HTTP** 서버의 **TLS**관련 설정을 수정하는 데 사용할 수 있는 **/etc/httpd/conf.d/ssl.conf** 구성 파일을 설치합니다. 마찬가지로 **mod_nss** 패키지는 **/etc/httpd/conf.d/nss.conf** 구성 파일을 설치합니다.

httpd-manual 패키지를 설치하여 **TLS** 구성을 포함하여 **Apache HTTP Server** 에 대한 전체 문서를 가져옵니다. **/etc/httpd/conf.d/ssl.conf** 구성 파일에서 사용 가능한 지시문은 **/usr/share/httpd/manual/mod/mod_ssl.html** 에 자세히 설명되어 있습니다. 다양한 설정의 예로는 **/usr/share/httpd/manual/ssl/ssl_howto.html** 에 있습니다.

/etc/httpd/conf.d/ssl.conf 구성 파일의 설정을 수정할 때 최소한 다음 세 가지 지시문을 고려해야 합니다.

SSLProtocol

허용하려는 TLS 버전(또는 SSL)을 지정하려면 이 지시문을 사용합니다.

SSLCipherSuite

이 지시문을 사용하여 선호하는 암호화 제품군을 지정하거나 허용하지 않을 암호화 제품군을 비활성화합니다.

SSLHonorCipherOrder

연결 클라이언트가 지정한 암호 순서를 준수하는지 확인하기 위해 이 지시문의 주석을 on 으로 설정합니다.

예를 들어 다음과 같습니다.

```
SSLProtocol all -SSLv2 -SSLv3
SSLCipherSuite HIGH:!aNULL:!MD5
SSLHonorCipherOrder on
```

위의 구성은 최소 사양이며, 4.13.1절. "사용할 알고리즘 선택" 에 설명된 권장 사항에 따라 크게 강화할 수 있습니다.

mod_nss 모듈을 구성하고 사용하려면 /etc/httpd/conf.d/nss.conf 구성 파일을 수정합니다. mod_nss 모듈은 mod_ssl 로부터 파생되며, 따라서 구성 파일의 구조가 아니라, 사용 가능한 지시문과 많은 기능을 공유합니다. mod_nss 지시문에는 SSL 대신 NSS 접두사가 있습니다. mod_nss 에 적용되지 않는 mod_ssl 구성 지시문 목록을 포함하여 mod_nss 에 대한 자세한 내용은 https://git.fedorahosted.org/cgit/mod_nss.git/plain/docs/mod_nss.html 를 참조하십시오.

4.13.3.2. Dovecot 메일 서버 구성

TLS 를 사용하도록 Dovecot 메일 서버의 설치를 구성하려면 /etc/dovecot/conf.d/10-ssl.conf 구성 파일을 수정합니다. /usr/share/doc/dovecot-2.2.10/wiki/SSL.DovecotConfiguration.txt 에서 해당 파일에서 사용할 수 있는 기본 구성 지시문 중 일부에 대한 설명을 찾을 수 있습니다(이 도움말 파일은 Dovet의 표준 설치와 함께 설치됨).

/etc/dovecot/conf.d/10-ssl.conf 구성 파일의 설정을 수정할 때 다음 세 지시문을 최소한으로 고려해야 합니다.

ssl_protocols

허용하려는 TLS 버전(또는 SSL)을 지정하려면 이 지시문을 사용합니다.

ssl_cipher_list

이 지시문을 사용하여 선호하는 암호화 제품군을 지정하거나 허용하지 않을 암호화 제품군을 비활성화합니다.

ssl_prefer_server_ciphers

연결 클라이언트가 지정한 암호 순서를 준수하는지 확인하기 위해 이 지시문의 주석을 제거하고 yes 로 설정합니다.

예를 들어 다음과 같습니다.

```
ssl_protocols = !SSLv2 !SSLv3
ssl_cipher_list = HIGH:!aNULL:!MD5
ssl_prefer_server_ciphers = yes
```

위의 구성은 최소 사양이며, 4.13.1절. "사용할 알고리즘 선택" 에 설명된 권장 사항에 따라 크게 강화할 수 있습니다.

4.13.4. 추가 정보

TLS 구성 및 관련 항목에 대한 자세한 내용은 아래 나열된 리소스를 참조하십시오.

설치된 문서

- *config(1) - /etc/ssl/openssl.conf 구성 파일의 형식을 설명합니다.*
- *ciphers(1) - 사용 가능한 OpenSSL 키워드 및 암호화 문자열 목록을 포함합니다.*
- */usr/share/httpd/manual/ mod_ssl.html - Apache HTTP Server 용 mod_ssl 모듈에서 사용하는 /etc/httpd/conf.d/ssl.conf 구성 파일에서 사용할 수 있는 지시문에 대한 자세한 설명을*

포함합니다.

- /usr/share/httpd/manual/ssl/ssl_howto.html - Apache HTTP Server 용 mod_ssl 모듈에서 사용하는 /etc/httpd/conf.d/ssl.conf 구성 파일에 실제 설정의 실제 예를 포함합니다.

- /usr/share/doc/dovecot-2.2.10/wiki/SSL.DovecotConfiguration.txt - Dovecot /conf.d/10-ssl.conf 설정 파일에서 사용할 수 있는 기본 설정 지시문에 대해 설명합니다.

온라인 문서

- Red Hat Enterprise Linux 7 SELinux 사용자 및 관리자 가이드 - Red Hat Enterprise Linux 7용 SELinux 사용자 및 관리자 가이드는 Apache HTTP Server 와 같은 다양한 서비스를 사용하여 SELinux 및 문서를 구성하고 사용하는 방법에 대해 자세히 설명합니다.

- http://tools.ietf.org/html/draft-ietf-uta-tls-bcp-00 - TLS 및 DTLS 보안 사용을 위한 권장 사항

다음을 참조하십시오.

- A.2.4절. “SSL/TLS” SSL 및 TLS 프로토콜에 대한 간결한 설명을 제공합니다.

- 4.7절. “OpenSSL 사용” 특히 OpenSSL 을 사용하여 키를 생성 및 관리하고 인증서를 생성하고, 파일을 암호화 및 해독하는 방법을 설명합니다.

4.14. 공유 시스템 인증서 사용

Shared System Certificates 스토리지를 사용하면 NSS, GnuTLS, OpenSSL, Java가 시스템 인증서 앵커와 블랙 리스트 정보를 검색하기 위한 기본 소스를 공유할 수 있습니다. 기본적으로 신뢰 저장소에는 긍정 및 부정적인 신뢰성을 포함한 Mozilla CA 목록이 포함되어 있습니다. 시스템을 사용하면 핵심 Mozilla CA 목록을 업데이트하거나 다른 인증서 목록을 선택할 수 있습니다.

4.14.1. 시스템 전체의 보안 저장소 사용

Red Hat Enterprise Linux 7에서 통합 시스템 전체 신뢰 저장소는 /etc/pki/ca-trust/ 및 /usr/share/pki/ca-trust-source/ 디렉터리에 있습니다. /usr/share/pki/ca-trust-source/의 신뢰 설정은 /etc/pki/ca-trust/의 설정보다 우선 순위가 낮습니다.

인증서 파일은 설치된 하위 디렉터리에 따라 처리됩니다.

- **/usr/share/pki/ca-trust-source/anchors/** 또는 **/etc/pki/ca-trust/source/anchors/** - 신뢰 앵커의 경우. 4.5.6절. “Trust Anchors 이해”을 참조하십시오.

- **/usr/share/pki/ca-trust-source/blacklist/** 또는 **/etc/pki/ca-trust/source/blacklist/** -에서 신뢰할 수 없는 인증서의 경우.

- 확장된 **BEGIN TRUSTED** 파일 형식의 인증서의 경우 **/usr/share/pki/ca-trust-source/** 또는 **/etc/pki/ca-trust/source/** 입니다.

4.14.2. 새 인증서 추가

간단한 **PEM** 또는 **DER** 파일 형식의 인증서를 시스템의 **CA** 목록에 추가하려면 인증서 파일을 **/usr/share/pki/ca-trust-source/anchors/** 또는 **/etc/pki/ca-trust/source/anchors/** 디렉터리에 복사합니다. 시스템 전체의 신뢰 저장소 구성을 업데이트하려면 **update-ca-trust** 명령을 사용합니다. 예를 들면 다음과 같습니다.

```
# cp ~/certificate-trust-examples/Cert-trust-test-ca.pem /usr/share/pki/ca-trust-source/anchors/
# update-ca-trust
```

참고

Firefox 브라우저에서는 **update-ca-trust** 를 실행하지 않고 추가된 인증서를 사용할 수 있지만 **CA**가 변경된 후 **update-ca-trust** 를 실행하는 것이 좋습니다. 또한 **Firefox, Epiphany** 또는 **Chromium**과 같은 브라우저에서 파일을 캐시하고 현재의 시스템 인증서 구성을 로드하려면 브라우저의 캐시를 지우거나 브라우저를 다시 시작해야 할 수도 있습니다.

4.14.3. 신뢰할 수 있는 시스템 인증서 관리

신뢰 앵커를 나열, 추출, 추가, 제거 또는 변경하려면 **trust** 명령을 사용합니다. 이 명령에 대한 기본 도움말을 보려면 인수 없이 또는 **--help** 지시문을 사용하여 입력합니다.

```
$ trust
usage: trust command <args>...

Common trust commands are:
  list            List trust or certificates
  extract         Extract certificates and trust
```

```
  extract-compat   Extract trust compatibility bundles
  anchor           Add, remove, change trust anchors
  dump             Dump trust objects in internal format

See 'trust <command> --help' for more information
```

*모든 시스템 신뢰 앵커 및 인증서를 나열하려면 **trust list** 명령을 사용합니다.*

```
$ trust list
pkcs11:id=%d2%87%b4%e3%df%37%27%93%55%f6%56%ea%81%e5%36%cc%8c%1e%3f%bd;ty
pe=cert
    type: certificate
    label: ACCVRAIZ1
    trust: anchor
    category: authority

pkcs11:id=%a6%b3%e1%2b%2b%49%b6%d7%73%a1%aa%94%f5%01%e7%73%65%4c%ac%50;t
ype=cert
    type: certificate
    label: ACEDICOM Root
    trust: anchor
    category: authority
...
[output has been truncated]
```

***trust** 명령의 모든 하위 명령은 상세한 기본 도움말을 제공합니다. 예를 들면 다음과 같습니다.*

```
$ trust list --help
usage: trust list --filter=<what>

  --filter=<what>     filter of what to export
                      ca-anchors        certificate anchors
                      blacklist         blacklisted certificates
                      trust-policy      anchors and blacklist (default)
                      certificates      all certificates
                      pkcs11:object=xx  a PKCS#11 URI
  --purpose=<usage>   limit to certificates usable for the purpose
                      server-auth       for authenticating servers
                      client-auth       for authenticating clients
                      email             for email protection
                      code-signing      for authenticating signed code
                      1.2.3.4.5...      an arbitrary object id
  -v, --verbose       show verbose debug output
  -q, --quiet         suppress command output
```

시스템 전체 신뢰 저장소에 신뢰 앵커를 저장하려면 신뢰 앵커 하위 명령을 사용하여 인증서를 지정합니다 . 예를 들면 다음과 같습니다.

```
# trust anchor path.to/certificate.crt
```

인증서를 제거하려면 **path.to a certificate** 또는 인증서 **ID**를 사용합니다.

```
# trust anchor --remove path.to/certificate.crt
# trust anchor --remove "pkcs11:id=%AA%BB%CC%DD%EE;type=cert"
```

4.14.4. 추가 리소스

자세한 내용은 다음 도움말 페이지를 참조하십시오.

- **update-ca-trust(8)**
- **trust(1)**

4.15. MACSEC 사용

미디어 액세스 제어 보안 (**MACsec, IEEE 802.1AE**)은 **FC-AES-128** 알고리즘을 사용하여 **LAN**의 모든 트래픽을 암호화하고 인증합니다. **MACsec** 은 **IP** 뿐만 아니라 **ARP(Address Resolution Protocol), Neighbor Discovery(ND)** 또는 **DHCP** 도 보호할 수 있습니다. **IPsec** 은 애플리케이션 계층(계층 **7**)의 네트워크 계층(계층 **3**) 및 **SSL** 또는 **TLS** 에서 작동하지만 **MACsec** 은 데이터 링크 계층(계층 **2**)에서 작동합니다. 다른 네트워킹 계층에서 **MACsec** 과 보안 프로토콜을 결합하여 이러한 표준에서 제공하는 다양한 보안 기능을 활용할 수 있습니다.

MACsec 네트워크 아키텍처, 사용 사례 시나리오 및 구성 예제에 대한 자세한 내용은 **MACsec:** 네트워크 트래픽 문서를 암호화하는 다른 솔루션 에서 참조하십시오.

wpa_supplicant 및 **NetworkManager** 를 사용하여 **MACsec**을 설정하는 방법은 **Red Hat Enterprise Linux 7** 네트워킹 가이드를 참조하십시오.

4.16. SCRUB를 사용하여 데이터 보안 제거

scrub 유틸리티는 특수 파일 또는 디스크 장치에 패턴을 설정하여 데이터 검색을 더 어렵게 만듭니다. **scrub** 를 사용하면 디스크에 임의의 데이터를 쓰는 것보다 빠릅니다. 이 프로세스는 고가용성, 신뢰성 및 데이터 보호를 제공합니다.

scrub 명령을 사용하려면 **scrub** 패키지를 설치합니다.

```
~]# yum install scrub
```

scrub 유틸리티는 다음 기본 모드 중 하나에서 작동합니다.

문자 또는 블록 장치

전체 디스크에 해당하는 특수 파일이 제거되어 해당 디스크의 모든 데이터가 삭제됩니다. 이것이 가장 효과적인 방법입니다.

```
scrub [OPTIONS] special file
```

파일

일반 파일이 수정되어 파일의 데이터만 삭제됩니다.

```
scrub [OPTIONS] file
```

디렉터리

X 옵션을 사용하면 파일 시스템이 가득 찰 때까지 디렉터리가 생성되고 파일로 채워집니다. 그런 다음 파일이 파일 모드에서 로 스크럽됩니다.

```
scrub -X [OPTIONS] directory
```

예 4.7. 원시 장치 제거

기본 **National Nuclear Security Administration (NNSA)** 패턴을 사용하여 원시 장치 **/dev/sdf1** 을 **scrub** 하려면 다음 명령을 입력합니다.

```
~]# scrub /dev/sdf1
scrub: using NNSA NAP-14.1-C patterns
scrub: please verify that device size below is correct!
scrub: scrubbing /dev/sdf1 1995650048 bytes (~1GB)
scrub: random  |.....................................................|
scrub: random  |.....................................................|
scrub: 0x00    |.....................................................|
scrub: verify  |.....................................................|
```

예 4.8. 파일 삭제

1. **1MB** 파일을 만듭니다.
   -

```
~]$ base64 /dev/urandom | head -c $[ 1024*1024 ] > file.txt
```

2. *파일 크기를 표시합니다.*

```
~]$ ls -lh
total 1.0M
-rw-rw-r--. 1 username username 1.0M Sep  8 15:23 file.txt
```

3. *파일의 내용을 표시합니다.*

```
~]$ head -1 file.txt
JnNpaTEveB/lYsbM9lhuJdw+0jKhwClBUsxLXLAyB8ultotUlNHKKUeS/7bCRKDogE
P+yJm8VQkL
```

4. *파일을 정리합니다.*

```
~]$ scrub file.txt
scrub: using NNSA NAP-14.1-C patterns
scrub: scrubbing file.txt 1048576 bytes (~1024KB)
scrub: random  |.............................................|
scrub: random  |.............................................|
scrub: 0x00    |.............................................|
scrub: verify  |.............................................|
```

5. *파일 내용을 스크럽했는지 확인합니다.*

```
~]$ cat file.txt
SCRUBBED!
```

6. *파일 크기가 동일하게 유지되는지 확인합니다.*

```
~]$ ls -lh
total 1.0M
-rw-rw-r--. 1 username username 1.0M Sep  8 15:24 file.txt
```

Scrub 모드, 옵션, 방법 및 경고에 대한 자세한 내용은 **scrub(1)** 도움말 페이지를 참조하십시오.

5장. 방화벽 사용

5.1. FIREWALLD시작하기

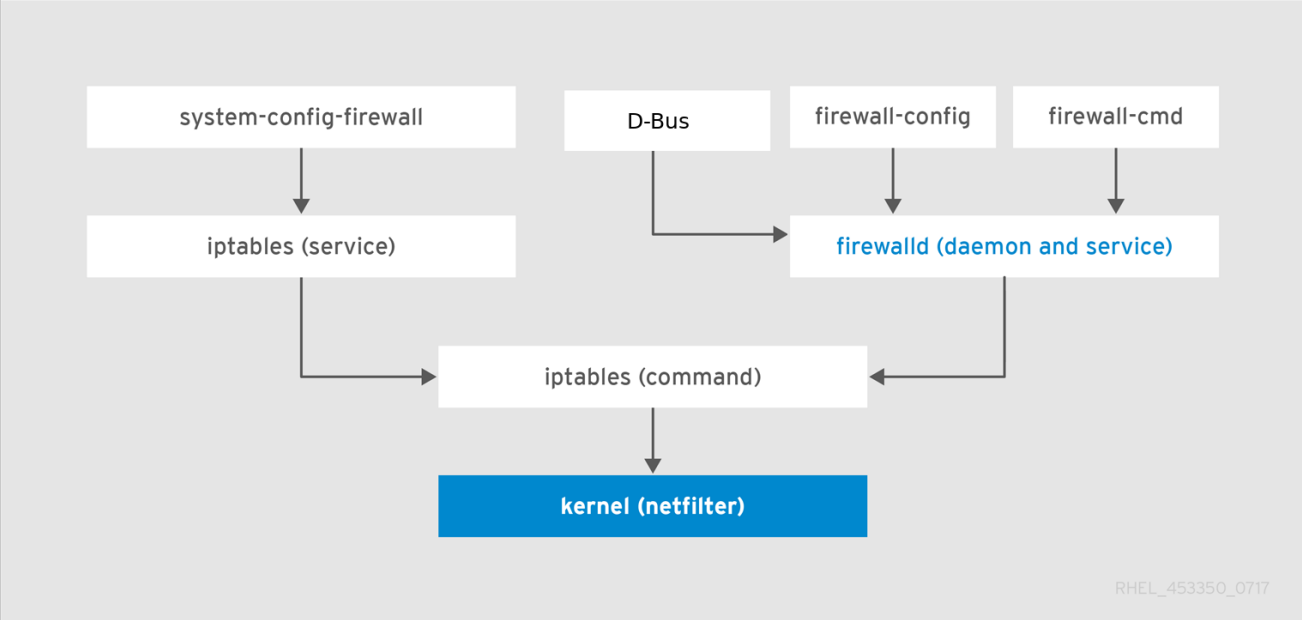
방화벽 은 원치 않는 트래픽으로부터 컴퓨터를 외부로부터 보호하는 방법입니다. 사용자가 방화벽 규칙 세트를 정의하여 호스트 시스템에서 들어오는 네트워크 트래픽을 제어할 수 있습니다. 이러한 규칙은 들어오는 트래픽을 정렬하고 차단하거나 를 허용하는 데 사용됩니다.

firewalld 는 **D-Bus** 인터페이스를 사용하여 동적 사용자 지정 가능 호스트 기반 방화벽을 제공하는 방화벽 서비스 데몬입니다. 동적이므로 규칙이 변경될 때마다 방화벽 데몬을 재시작할 필요 없이 규칙을 생성, 변경 및 삭제할 수 있습니다.

firewalld 는 트래픽 관리를 간소화하는 영역 및 서비스 의 개념을 사용합니다. 영역은 사전 정의된 규칙 집합입니다. 네트워크 인터페이스와 소스를 영역에 할당할 수 있습니다. 허용된 트래픽은 컴퓨터가 연결된 네트워크에 따라 다르며 이 네트워크가 할당된 보안 수준에 따라 달라집니다. 방화벽 서비스는 특정 서비스에 대한 들어오는 트래픽을 허용하고 영역 내에서 적용됩니다.

서비스는 네트워크 통신에 대해 하나 이상의 포트 또는 주소를 사용합니다. 방화벽은 포트를 기반으로 통신을 필터링합니다. 서비스에 대한 네트워크 트래픽을 허용하려면 해당 포트를 열어야 합니다.**firewalld** 는 명시적으로 열려 있지 않은 포트의 모든 트래픽을 차단합니다. 신뢰할 수 있는 것과 같은 일부 영역에서는 기본적으로 모든 트래픽을 허용합니다.

그림 **5.1.** 방화벽 스택



[D]

5.1.1. 영역

firewalld 는 사용자가 해당 네트워크 내의 인터페이스와 트래픽에 배치하기로 결정한 신뢰 수준에 따라 네트워크를 다른 영역으로 분리하는 데 사용할 수 있습니다. 연결은 하나의 영역에만 속할 수 있지만, 여러 네트워크 연결에 영역을 사용할 수 있습니다.

NetworkManager 는 인터페이스 영역을 firewalld 에 알립니다. NetworkManager, firewall-config 툴 또는 firewall-cmd 명령줄 도구를 사용하여 인터페이스에 영역을 할당할 수 있습니다. 두 번째는 적절한 NetworkManager 구성 파일만 편집합니다. firewall-cmd 또는 firewall-config 를 사용하여 인터페이스의 영역을 변경하는 경우 요청은 NetworkManager 로 전달되며,firewalld 에서는 처리되지 않습니다.

사전 정의된 영역은 /usr/lib/firewalld/zones/ 디렉터리에 저장되며 사용 가능한 모든 네트워크 인터페이스에 즉시 적용할 수 있습니다. 이러한 파일은 수정된 후에만 /etc/firewalld/zones/ 디렉토리에 복사됩니다. 다음 표에서는 사전 정의된 영역의 기본 설정을 설명합니다.

블록

들어오는 네트워크 연결은 IPv4 및 IPv6 용으로 icmp6-adm-prohibited 에 대한 icmp-host-prohibited 메시지와 함께 거부됩니다. 시스템 내에서 시작된 네트워크 연결만 가능합니다.

dmz

내부 네트워크에 대한 제한된 액세스 권한으로 공개적으로 액세스할 수 있는 내구성 있는 영역의 컴퓨터의 경우. 선택한 들어오는 연결만 허용됩니다.

drop

들어오는 모든 네트워크 패킷은 알림 없이 삭제됩니다. 발신 네트워크 연결만 가능합니다.

external

특히 라우터의 경우 마스커레이딩이 활성화된 외부 네트워크에서 사용합니다. 네트워크의 다른 컴퓨터를 신뢰하지 않고 컴퓨터를 손상시키지 않습니다. 선택한 들어오는 연결만 허용됩니다.

집

대부분 네트워크에있는 다른 컴퓨터를 신뢰할 때 집에서 사용하기 위해. 선택한 들어오는 연결만 허용됩니다.

internal

내부 네트워크에서 사용되는 경우 네트워크의 다른 컴퓨터를 대부분 신뢰할 수 있습니다. 선택한 들어오는 연결만 허용됩니다.

공개

네트워크의 다른 컴퓨터를 신뢰하지 않는 공공 영역에서 사용하기 위해. 선택한 들어오는 연결만 허용됩니다.

trusted

모든 네트워크 연결이 허용됩니다.

work

네트워크의 다른 컴퓨터를 대부분 신뢰할 수 있는 작업에서 사용하기 위해 선택한 들어오는 연결만 허용됩니다.

이러한 영역 중 하나는 기본 영역으로 설정됩니다. **NetworkManager** 에 인터페이스 연결이 추가되면 기본 영역에 할당됩니다. 설치 시 **firewalld** 의 기본 영역이 공개 영역으로 설정됩니다. 기본 영역을 변경할 수 있습니다.

참고

네트워크 영역 이름은 자체 설명으로 선택되었으며 사용자가 합리적인 결정을 신속하게 내릴 수 있도록 선택되었습니다. 보안 문제를 방지하려면 기본 영역 구성을 검토하고 요구 사항 및 위험 평가에 따라 불필요한 서비스를 비활성화합니다.

5.1.2. 사전 정의된 서비스

서비스는 로컬 포트, 프로토콜, 소스 포트 및 대상 목록 및 서비스가 활성화된 경우 방화벽 도우미 모듈 목록일 수 있습니다. 서비스를 사용하면 모든 것을 하나씩 설정하는 대신 단일 단계에서 포트 열기, 프로

토콜 정의, 패킷 전달 활성화 등의 여러 작업을 수행할 수 있으므로 사용자가 시간을 절약할 수 있습니다.

*서비스 구성 옵션 및 일반 파일 정보는 **firewalld.service(5)** 도움말 페이지에 설명되어 있습니다. 서비스는 **service-name.xml** 형식으로 이름이 지정된 개별 **XML** 구성 파일을 통해 지정됩니다. 프로토콜 이름은 **firewalld** 에서 서비스 또는 애플리케이션 이름보다 우선합니다.*

5.1.3. 런타임 및 영구 설정

*런타임 모드에서 커밋된 모든 변경 사항은 **firewalld** 가 실행되는 동안만 적용됩니다. **firewalld** 를 다시 시작하면 설정이 영구 값으로 되돌아갑니다.*

*재부팅 후에도 변경 사항을 적용하려면 **--permanent** 옵션을 사용하여 다시 적용합니다. 또는 **firewalld** 가 실행되는 동안 변경 사항을 영구적으로 만들려면 **--runtime-to-permanent firewall-cmd** 옵션을 사용합니다.*

***firewalld** 가 **--permanent** 옵션만 사용하여 실행되는 동안 규칙을 설정하는 경우 **firewalld** 를 다시 시작하기 전에 적용되지 않습니다. 그러나 **firewalld** 를 다시 시작하면 모든 열려 있는 포트를 종료하고 네트워킹 트래픽을 중지합니다.*

5.1.4. CLI를 사용하여 런타임 및 영구 구성에서 설정 수정

***CLI**를 사용하면 두 모드에서 동시에 방화벽 설정을 수정하지 않습니다. 런타임 또는 영구 모드만 수정합니다. 영구 모드에서 방화벽 설정을 수정하려면 **firewall-cmd** 명령과 함께 **--permanent** 옵션을 사용합니다.*

```
~]# firewall-cmd --permanent <other options>
```

이 옵션이 없으면 명령은 런타임 모드를 수정합니다.

두 모드의 설정을 변경하려면 다음 두 가지 방법을 사용할 수 있습니다.

1. *런타임 설정을 변경한 후 다음과 같이 영구적으로 설정합니다.*

```
~]# firewall-cmd <other options>
~]# firewall-cmd --runtime-to-permanent
```

2. 영구 설정을 설정하고 설정을 런타임 모드로 다시 로드합니다.

```
~]# firewall-cmd --permanent <other options>
~]# firewall-cmd --reload
```

첫 번째 방법을 사용하면 영구 모드로 적용하기 전에 설정을 테스트할 수 있습니다.

참고

특히 원격 시스템에서는 잘못된 설정으로 인해 사용자가 시스템에서 잠길 수 없습니다. 이러한 상황을 방지하려면 **--timeout** 옵션을 사용합니다. 지정된 시간 후에 모든 변경 사항이 이전 상태로 되돌아갑니다. 이 옵션을 사용하면 **--permanent** 옵션이 제외됩니다.

예를 들어 **SSH** 서비스를 **15**분 동안 추가하려면 다음을 수행합니다.

```
~]# firewall-cmd --add-service=ssh --timeout 15m
```

5.2. FIREWALL-CONFIG GUI 구성 툴 설치

firewall-config GUI 구성 도구를 사용하려면 **firewall-config** 패키지를 **root** 로 설치합니다.

```
~]# yum install firewall-config
```

또는 **GNOME** 에서 **Super** 키를 사용하고 **Software** 을 입력하여 소프트웨어 소스 애플리케이션을 시작합니다. 오른쪽 상단에 있는 검색 버튼을 선택한 후 표시되는 검색 상자에 **firewall** 을 입력합니다. 검색 결과에서 방화벽 항목을 선택하고 설치 버튼을 클릭합니다.

firewall-config 를 실행하려면 **firewall-config** 명령을 사용하거나 **Super** 키를 눌러 활동 개요 를 입력하고 **firewall** 를 입력한 후 **Enter** 키를 누릅니다.

5.3. FIREWALLD의 현재 상태 및 설정 보기

5.3.1. firewalld의 현재 상태 보기

방화벽 서비스인 **firewalld** 는 기본적으로 시스템에 설치됩니다. **firewalld CLI** 인터페이스를 사용하여 서비스가 실행 중인지 확인합니다.

서비스 상태를 보려면 다음을 수행합니다.

```
~]# firewall-cmd --state
```

서비스 상태에 대한 자세한 내용은 **systemctl status** 하위 명령을 사용합니다.

```
~]# systemctl status firewalld
firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor pr
   Active: active (running) since Mon 2017-12-18 16:05:15 CET; 50min ago
     Docs: man:firewalld(1)
 Main PID: 705 (firewalld)
    Tasks: 2 (limit: 4915)
   CGroup: /system.slice/firewalld.service
           └─705 /usr/bin/python3 -Es /usr/sbin/firewalld --nofork --nopid
```

또한 설정을 편집하기 전에 **firewalld** 가 설정된 방법과 어떤 규칙이 강제 적용되는지 알아야 합니다. 방화벽 설정을 표시하려면 다음을 참조하십시오. **5.3.2절. "현재 firewalld 설정 보기"**

5.3.2. 현재 firewalld 설정 보기

5.3.2.1. GUI를 사용하여 허용된 서비스 보기

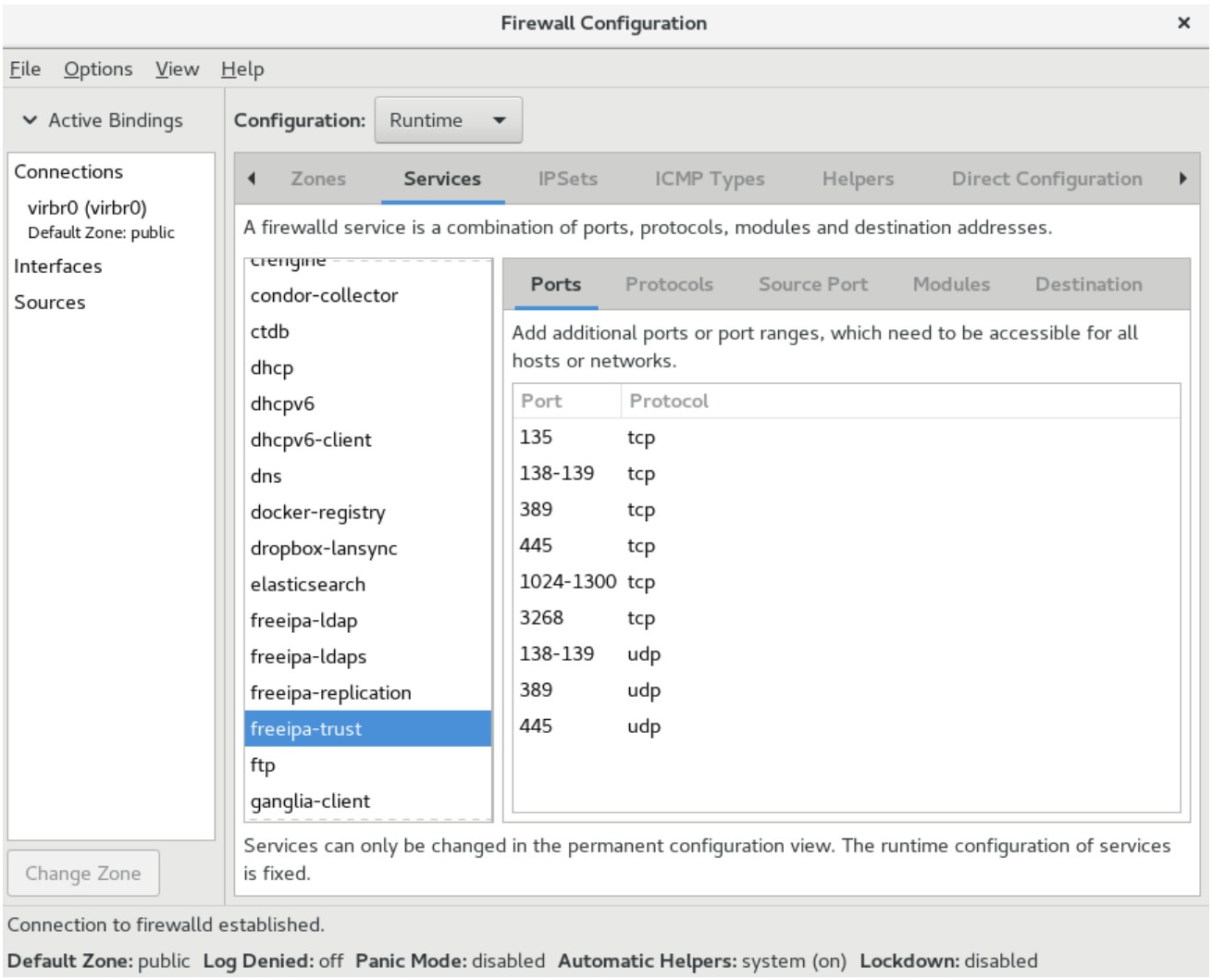
그래픽 **firewall-config** 도구를 사용하여 서비스 목록을 보려면 **Super** 키를 눌러 활동 개요를 입력하고 방화벽을 입력한 후 **Enter** 키를 누릅니다. **firewall-config** 도구가 표시됩니다. 이제 **Services** (서비스) 탭에서 서비스 목록을 볼 수 있습니다.

또는 명령줄을 사용하여 그래픽 방화벽 구성 툴을 시작하려면 다음 명령을 입력합니다.

```
~]$ firewall-config
```

방화벽 구성 창이 열립니다. 이 명령은 일반 사용자로 실행할 수 있지만 간혹 관리자 암호를 입력하라는 메시지가 표시됩니다.

그림 5.2. firewall-config의 Services 탭



[D]

5.3.2.2. CLI를 사용하여 firewalld 설정 보기

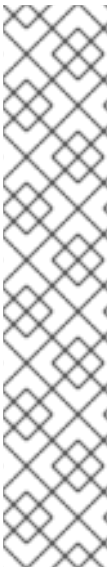
CLI 클라이언트를 사용하면 현재 방화벽 설정을 다르게 볼 수 있습니다. --list-all 옵션은 firewalld 설정에 대한 전체 개요를 표시합니다.

firewalld 는 영역을 사용하여 트래픽을 관리합니다. --zone 옵션으로 영역을 지정하지 않으면 활성 네트워크 인터페이스 및 연결에 할당된 기본 영역에서 명령이 유효합니다.

기본 영역의 모든 관련 정보를 나열하려면 다음을 수행합니다.

```
~]# firewall-cmd --list-all
public
  target: default
  icmp-block-inversion: no
  interfaces:
  sources:
```

```
  services: ssh dhcpv6-client
  ports:
  protocols:
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```



참고

*설정을 표시할 영역을 지정하려면 **firewall-cmd --list-all** 명령에 **--zone=zone-name** 인수를 추가합니다. 예를 들면 다음과 같습니다.*

```
~]# firewall-cmd --list-all --zone=home
home
  target: default
  icmp-block-inversion: no
  interfaces:
  sources:
  services: ssh mdns samba-client dhcpv6-client
... [output truncated]
```

*서비스 또는 포트와 같은 특정 정보에 대한 설정을 보려면 특정 옵션을 사용합니다. 명령 도움말을 사용하여 **firewalld** 매뉴얼 페이지를 참조하거나 옵션 목록을 가져옵니다.*

```
~]# firewall-cmd --help

Usage: firewall-cmd [OPTIONS...]

General Options
  -h, --help          Prints a short help text and exists
  -V, --version       Print the version string of firewalld
  -q, --quiet         Do not print status messages

Status Options
  --state             Return and print firewalld state
  --reload            Reload firewall and keep state information
... [output truncated]
```

예를 들어 현재 영역에서 허용되는 서비스를 확인하려면 다음을 수행합니다.

```
~]# firewall-cmd --list-services
ssh dhcpv6-client
```

***CLI** 툴을 사용하여 특정 하위 파트의 설정을 나열하기 어려울 수 있습니다. 예를 들어 **SSH** 서비스를*

허용하고 **firewalld** 는 서비스에 필요한 포트(22)를 엽니다. 나중에 허용된 서비스를 나열하면 목록에 **SSH** 서비스가 표시되지만 열려 있는 포트를 나열하는 경우 표시되지 않습니다. 따라서 전체 정보를 받으려면 **--list-all** 옵션을 사용하는 것이 좋습니다.

5.4. FIREWALLD시작

firewalld 를 시작하려면 **root** 로 다음 명령을 입력합니다.

```
~]# systemctl unmask firewalld
~]# systemctl start firewalld
```

시스템 시작 시 **firewalld** 가 자동으로 시작되도록 하려면 **root** 로 다음 명령을 입력합니다.

```
~]# systemctl enable firewalld
```

5.5. FIREWALLD중지

firewalld 를 중지하려면 **root** 로 다음 명령을 입력합니다.

```
~]# systemctl stop firewalld
```

firewalld 가 시스템을 시작할 때 자동으로 시작되지 않도록 하려면 **root** 로 다음 명령을 입력합니다.

```
~]# systemctl disable firewalld
```

firewalld D-Bus 인터페이스에 액세스하여 **firewalld** 를 시작하지 않고 다른 서비스에 **firewalld** 가 필요한 경우에도 다음 명령을 **root** 로 입력합니다.

```
~]# systemctl mask firewalld
```

5.6. 트래픽 제어

5.6.1. 사전 정의된 서비스

그래픽 **firewall-config** 도구, **firewall-cmd** 및 **firewall-offline-cmd** 를 사용하여 서비스를 추가 및 제거할 수 있습니다.

또는 **/etc/firewalld/services/** 디렉터리에서 **XML** 파일을 편집할 수 있습니다. 사용자가 서비스를 추가하거나 변경하지 않으면 **/etc/firewalld/services/** 에서 해당 **XML** 파일을 찾을 수 없습니다. **/usr/lib/firewalld/services/** 디렉터리에 있는 파일은 서비스를 추가하거나 변경하려는 경우 템플릿으로 사용할 수 있습니다.

5.6.2. CLI를 사용하여 긴급 케이스의 모든 트래픽 비활성화

시스템 공격과 같은 긴급 상황에서 모든 네트워크 트래픽을 비활성화하고 공격자를 차단할 수 있습니다.

네트워킹 트래픽을 즉시 비활성화하려면 다음에서 패닉 모드를 전환합니다.

```
~]# firewall-cmd --panic-on
```

패닉 모드를 전환하면 방화벽을 영구 설정으로 되돌립니다. **panic** 모드를 전환하려면 다음을 수행합니다.

```
~]# firewall-cmd --panic-off
```

패닉 모드가 꺼졌는지 여부를 확인하려면 다음을 사용하십시오.

```
~]# firewall-cmd --query-panic
```

5.6.3. CLI를 사용하여 사전 정의 서비스로 트래픽 제어

트래픽을 제어하는 가장 간단한 방법은 **firewalld** 에 사전 정의된 서비스를 추가하는 것입니다. 이렇게 하면 필요한 모든 포트가 열리고 서비스 정의 파일에 따라 다른 설정이 변경됩니다.

1. 서비스가 아직 허용되지 않았는지 확인합니다.

```
~]# firewall-cmd --list-services
ssh dhcpv6-client
```

2. 사전 정의된 모든 서비스를 나열합니다.

```
~]# firewall-cmd --get-services
RH-Satellite-6 amanda-client amanda-k5-client bacula bacula-client bitcoin bitcoin-rpc
bitcoin-testnet bitcoin-testnet-rpc ceph ceph-mon cfengine condor-collector ctdb dhcp dhcpv6
dhcpv6-client dns docker-registry ...
[output truncated]
```

3. 허용된 서비스에 서비스를 추가합니다.

```
~]# firewall-cmd --add-service=<service-name>
```

4. 새 설정을 영구적으로 설정합니다.

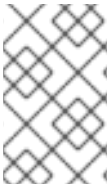
```
~]# firewall-cmd --runtime-to-permanent
```

5.6.4. GUI를 사용하여 사전 정의 서비스로 트래픽 제어

사전 정의된 서비스 또는 사용자 지정 서비스를 활성화하거나 비활성화하려면 **firewall-config** 도구를 시작하고 서비스를 구성할 네트워크 영역을 선택합니다. **Services** 탭을 선택하고 신뢰할 수 있는 각 서비스 유형의 확인란을 선택합니다. 확인란을 선택 취소하여 서비스를 차단합니다.

서비스를 편집하려면 **firewall-config** 툴을 시작하고 **Configuration** (구성) 레이블이 지정된 메뉴에서 **Permanent** 를 선택합니다. 추가 아이콘과 메뉴 버튼은 서비스 창 하단에 표시됩니다. 구성할 서비스를 선택합니다.

Ports,Protocols, Source Port (포트, 프로토콜, 소스 포트) 탭에서는 선택한 서비스의 포트, 프로토콜, 소스 포트를 추가, 변경, 제거할 수 있습니다. **modules** 탭은 **Netfilter** 도우미 모듈을 구성하기 위한 것입니다. 대상 탭을 사용하면 특정 대상 주소 및 인터넷 프로토콜(**IPv4** 또는 **IPv6**)으로 트래픽을 제한할 수 있습니다.



참고

런타임 모드에서 서비스 설정을 변경할 수 없습니다.

5.6.5. 새 서비스 추가

그래픽 **firewall-config** 도구, **firewall-cmd** 및 **firewall-offline-cmd** 를 사용하여 서비스를 추가 및 제거할 수 있습니다. 또는 **/etc/firewalld/services/** 에서 **XML** 파일을 편집할 수 있습니다. 사용자가 서비스를 추가하거나 변경하지 않으면 **/etc/firewalld/services/** 에서 해당 **XML** 파일을 찾을 수 없습니다.

/usr/lib/firewalld/services/ 파일은 서비스를 추가하거나 변경하려는 경우 템플릿으로 사용할 수 있습니다.

터미널에 새 서비스를 추가하려면 firewalld 가 활성화되지 않은 경우 firewall-cmd 또는 firewall-offline-cmd 를 사용하십시오. 다음 명령을 입력하여 새 서비스와 빈 서비스를 추가합니다.

```
~]$ firewall-cmd --new-service=service-name
```

로컬 파일을 사용하여 새 서비스를 추가하려면 다음 명령을 사용하십시오.

```
~]$ firewall-cmd --new-service-from-file=service-name.xml
```

추가 --name=service-name옵션을 사용하여 서비스 이름을 변경할 수 있습니다.

서비스 설정이 변경되면 업데이트된 서비스 복사본이 /etc/firewalld/services/ 에 배치됩니다.

root 로 다음 명령을 입력하여 서비스를 수동으로 복사할 수 있습니다.

```
~]# cp /usr/lib/firewalld/services/service-name.xml /etc/firewalld/services/service-name.xml
```

firewalld 는 /usr/lib/firewalld/services 의 파일을 맨 처음에 로드합니다. 파일이 /etc/firewalld/services 에 있고 유효한 경우 /usr/lib/firewalld/services 의 일치하는 파일을 재정의합니다. /etc/firewalld/services 에서 일치하는 파일이 제거되었거나 firewalld 가 서비스의 기본값을 로드하라는 요청을 받은 경우 /usr/lib/firewalld/services 의 overriden 파일이 사용되는 즉시 사용됩니다. 이는 영구 환경에만 적용됩니다. 런타임 환경에서도 이러한 대체를 가져오려면 다시 로드해야 합니다.

5.6.6. CLI를 사용하여 포트 제어

포트는 운영 체제가 네트워크 트래픽을 수신 및 구분하고 이에 따라 시스템 서비스로 전달할 수 있는 논리적 장치입니다. 이는 일반적으로 포트에서 수신 대기하는 데몬으로 표시됩니다. 이는 이 포트로 들어오는 모든 트래픽을 대기합니다.

일반적으로 시스템 서비스는 예약된 표준 포트에서 수신 대기합니다. 예를 들어 httpd 데몬은 포트 80 에서 수신 대기합니다. 그러나 시스템 관리자는 기본적으로 보안을 강화하거나 기타 이유로 인해 다른 포트에서 수신 대기하도록 데몬을 구성합니다.

포트 열기

열려 있는 포트를 통해 시스템에서 보안 위험을 나타내는 외부에서 액세스할 수 있습니다. 일반적으로 포트를 닫고 특정 서비스에 필요한 경우에만 엽니다.

현재 영역에서 열려 있는 포트 목록을 가져오려면 다음을 수행합니다.

1. *허용된 모든 포트를 나열합니다.*

```
~]# firewall-cmd --list-ports
```

2. *들어오는 트래픽에 대해 허용 가능한 포트에 포트를 추가하여 해당 포트를 엽니다.*

```
~]# firewall-cmd --add-port=port-number/port-type
```

3. *새 설정을 영구적으로 설정합니다.*

```
~]# firewall-cmd --runtime-to-permanent
```

*포트 유형은 **tcp,udp,sctp** 또는 **dccp** 입니다. 유형은 네트워크 통신 유형과 일치해야 합니다.*

포트 닫기

*열린 포트가 더 이상 필요하지 않으면 **firewalld** 에서 해당 포트를 닫습니다. 포트를 열어 두면 보안 위험이 있으므로 사용하지 않는 즉시 불필요한 포트를 모두 닫는 것이 좋습니다.*

포트를 종료하려면 허용된 포트 목록에서 제거합니다.

1. *허용된 모든 포트를 나열합니다.*

```
~]# firewall-cmd --list-ports
[WARNING]
====
This command will only give you a list of ports that have been opened as ports. You will not be able to see any open ports that have been opened as a service. Therefore, you should consider using the --list-all option instead of --list-ports.
====
```

2. 들어오는 트래픽에 대해 허용된 포트에서 포트를 제거하여 해당 포트를 종료합니다.

```
~]# firewall-cmd --remove-port=port-number/port-type
```

3. 새 설정을 영구적으로 설정합니다.

```
~]# firewall-cmd --runtime-to-permanent
```

5.6.7. GUI를 사용하여 포트 열기

방화벽을 통한 트래픽을 특정 포트로 허용하려면 **firewall-config** 도구를 시작하고 변경하려는 설정이 있는 네트워크 영역을 선택합니다. **Ports** 탭을 선택하고 오른쪽에서 추가 버튼을 클릭합니다. 포트 및 프로토콜 창이 열립니다.

허용할 포트 번호 또는 범위를 입력합니다. 목록에서 **tcp** 또는 **udp** 를 선택합니다.

5.6.8. GUI를 사용하여 프로토콜로 트래픽 제어

특정 프로토콜을 사용하여 방화벽을 통한 트래픽을 허용하려면 **firewall-config** 도구를 시작하고 변경하려는 설정이 있는 네트워크 영역을 선택합니다. 프로토콜 탭을 선택하고 오른쪽의 추가 버튼을 클릭합니다. 프로토콜 창이 열립니다.

목록에서 프로토콜을 선택하거나 기타 프로토콜 확인란을 선택하고 필드에 프로토콜을 입력합니다.

5.6.9. GUI를 사용하여 소스 포트 열기

특정 포트에서 방화벽을 통한 트래픽을 허용하려면 **firewall-config** 도구를 시작하고 변경하려는 설정이 있는 네트워크 영역을 선택합니다. 소스 포트 탭을 선택하고 오른쪽에서 추가 버튼을 클릭합니다. 소스 포트 창이 열립니다.

허용할 포트 번호 또는 범위를 입력합니다. 목록에서 **tcp** 또는 **udp** 를 선택합니다.

5.7. 영역 작업

영역은 들어오는 트래픽을 더 투명하게 관리하는 개념을 나타냅니다. 영역은 네트워킹 인터페이스에 연결되거나 다양한 소스 주소가 할당됩니다. 각 영역의 방화벽 규칙을 독립적으로 관리하여 복잡한 방화

벽 설정을 정의하고 트래픽에 적용할 수 있습니다.

5.7.1. 영역 나열

시스템에서 사용할 수 있는 영역을 확인하려면 다음을 수행합니다.

```
~]# firewall-cmd --get-zones
```

firewall-cmd --get-zones 명령은 시스템에서 사용할 수 있는 모든 영역을 표시하지만 특정 영역에 대한 세부 정보는 표시되지 않습니다.

모든 영역에 대한 자세한 정보를 보려면 다음을 수행합니다.

```
~]# firewall-cmd --list-all-zones
```

특정 영역에 대한 자세한 정보를 보려면 다음을 수행합니다.

```
~]# firewall-cmd --zone=zone-name --list-all
```

5.7.2. Certain Zone의 firewalld 설정 수정

5.6.3절. “CLI를 사용하여 사전 정의 서비스로 트래픽 제어” 및 5.6.6절. “CLI를 사용하여 포트 제어” 은 현재 작업 영역의 범위에서 서비스를 추가하거나 포트를 수정하는 방법을 설명합니다. 경우에 따라 다른 영역에 규칙을 설정해야 하는 경우가 있습니다.

다른 영역에서 작업하려면 **--zone=zone-name** 옵션을 사용합니다. 예를 들어 퍼블릭 영역의 **SSH** 서비스를 허용하려면 다음을 수행합니다.

```
~]# firewall-cmd --add-service=ssh --zone=public
```

5.7.3. 기본 영역 변경

시스템 관리자는 구성 파일의 네트워킹 인터페이스에 영역을 할당합니다. 인터페이스가 특정 영역에 할당되지 않으면 기본 영역에 할당됩니다. **firewalld** 서비스를 다시 시작한 후 **firewalld** 는 기본 영역의 설정을 로드하고 활성 상태로 설정합니다.

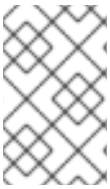
기본 영역을 설정하려면 다음을 수행합니다.

1. *현재 기본 영역을 표시합니다.*

```
~]# firewall-cmd --get-default-zone
```

2. *새 기본 영역을 설정합니다.*

```
~]# firewall-cmd --set-default-zone zone-name
```



참고

*이 절차에 따라 **--permanent** 옵션이 없어도 설정이 영구 설정입니다.*

5.7.4. 영역에 네트워크 인터페이스 할당

서로 다른 영역에 대해 다른 규칙 세트를 정의한 다음, 사용되는 인터페이스의 영역을 변경하여 설정을 빠르게 변경할 수 있습니다. 여러 인터페이스를 사용하면 각 인터페이스를 통해 들어오는 트래픽을 구분하도록 특정 영역을 설정할 수 있습니다.

특정 인터페이스에 영역을 할당하려면 다음을 수행합니다.

1. *활성 영역 및 여기에 할당된 인터페이스를 나열합니다.*

```
~]# firewall-cmd --get-active-zones
```

2. *인터페이스를 다른 영역에 할당합니다.*

```
~]# firewall-cmd --zone=zone-name --change-interface=<interface-name>
```

참고

*다시 시작해도 **--permanent** 옵션을 사용하여 설정을 영구적으로 설정할 필요가 없습니다. 새 기본 영역을 설정하면 설정이 영구적으로 됩니다.*

5.7.5. 네트워크 연결에 기본 영역 할당

NetworkManager 에서 연결을 관리하는 경우 사용되는 영역을 알고 있어야 합니다. 모든 네트워크 연결에 대해, 이동식 장치가 있는 컴퓨터의 위치에 따라 다양한 방화벽 설정의 유연성을 제공하는 영역을 지정할 수 있습니다. 따라서 회사 또는 집과 같은 다른 위치에 영역 및 설정을 지정할 수 있습니다.

인터넷 연결에 대한 기본 영역을 설정하려면 **NetworkManager GUI**를 사용하거나 **/etc/sysconfig/network-scripts/ifcfg-connection-name** 파일을 편집하고 영역을 이 연결에 할당하는 행을 추가합니다.

```
ZONE=zone-name
```

5.7.6. 새 영역 생성

사용자 지정 영역을 사용하려면 새 영역을 생성하고 사전 정의된 영역처럼 사용합니다. 새 영역에는 **--permanent** 옵션이 필요합니다. 그렇지 않으면 명령이 작동하지 않습니다.

새 영역을 생성하려면 다음을 수행합니다.

1. 새 영역을 생성합니다.

```
~]# firewall-cmd --new-zone=zone-name
```

2. 새 영역이 영구 설정에 추가되었는지 확인합니다.

```
~]# firewall-cmd --get-zones
```

3. 새 설정을 영구적으로 설정합니다.

```
~]# firewall-cmd --runtime-to-permanent
```

5.7.7. 구성 파일을 사용하여 새 영역 생성

영역은 영역 구성 파일을 사용하여 생성할 수도 있습니다. 이 방법은 새 영역을 생성해야 하지만 다른 영역에서의 설정을 재사용하고 일부만 변경하려고 할 때 유용할 수 있습니다.

firewalld 영역 구성 파일에는 영역에 대한 정보가 포함되어 있습니다. 다음은 XML 파일 형식으로 영역 설명, 서비스, 포트, 프로토콜, icmp-blocks, masquerade, forward-ports 및 풍부한 언어 규칙입니다. 파일 이름은 zone-name 의 길이가 현재 17 characters로 제한되는 zone-name.xml 이어야 합니다. 영역 구성 파일은 /usr/lib/firewalld/zones/ 및 /etc/firewalld/zones/ 디렉터리에 있습니다.

다음 예제에서는 TCP 및 UDP 프로토콜 둘 다에 대해 하나의 서비스(SSH)와 하나의 포트 범위를 허용하는 구성을 보여줍니다.

```
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>My zone</short>
  <description>Here you can describe the characteristic features of the zone.</description>
  <service name="ssh"/>
  <port port="1025-65535" protocol="tcp"/>
  <port port="1025-65535" protocol="udp"/>
</zone>
```

해당 영역의 설정을 변경하려면 섹션을 추가하여 포트를 추가하고 포트, 서비스 등을 전달합니다. 자세한 내용은 firewalld.zone 매뉴얼 페이지를 참조하십시오.

5.7.8. 영역 대상을 사용하여 Incoming Traffic에 기본 동작 설정

모든 영역에 대해 추가 지정되지 않은 들어오는 트래픽을 처리하는 기본 동작을 설정할 수 있습니다. 이러한 동작은 영역의 대상을 설정하여 정의됩니다. 기본,ACCEPT,REJECT, DROP 등 세 가지 옵션이 있습니다. 대상을 ACCEPT 로 설정하면 특정 규칙에 의해 비활성화된 패킷을 제외한 모든 수신 패킷을 허용하게 됩니다. 대상을 REJECT 또는 DROP 으로 설정하면 특정 규칙에서 허용된 패킷을 제외한 모든 수신 패킷을 비활성화합니다. 패킷이 거부되면 패킷이 삭제될 때 전송되는 정보가 없는 동안 소스 시스템에 거부에 대한 정보가 제공됩니다.

영역의 대상을 설정하려면 다음을 수행합니다.

1. *기본 대상을 확인하려면 특정 영역에 대한 정보를 나열합니다.*

```
~]$ firewall-cmd --zone=zone-name --list-all
```

2. *영역에 새 대상을 설정합니다.*

```
~]# firewall-cmd --zone=zone-name --set-target=<default|ACCEPT|REJECT|DROP>
```

5.8. ZONE을 사용하여 소스에 따라 트래픽 관리

영역을 사용하여 소스에 따라 들어오는 트래픽을 관리할 수 있습니다. 이를 통해 들어오는 트래픽을 정렬하고 다른 영역을 통해 라우팅하여 해당 트래픽으로 도달할 수 있는 서비스를 허용하거나 허용하지 않도록 할 수 있습니다.

영역에 소스를 추가하면 영역이 활성화되고 해당 소스에서 들어오는 트래픽이 전달됩니다. 지정된 소스의 트래픽에 적절하게 적용되는 각 영역의 다른 설정을 지정할 수 있습니다. 하나의 네트워크 인터페이스만 있어도 더 많은 영역을 사용할 수 있습니다.

5.8.1. 소스 추가

들어오는 트래픽을 특정 소스로 라우팅하려면 소스를 해당 영역에 추가합니다. 소스는 **IP** 주소 또는 **CIDR(Classless Inter-domain Routing)** 표기법의 **IP** 마스크일 수 있습니다.

1. 현재 영역에서 소스를 설정하려면 다음을 수행합니다.

```
~]# firewall-cmd --add-source=<source>
```

2. 특정 영역의 소스 **IP** 주소를 설정하려면 다음을 수행합니다.

```
~]# firewall-cmd --zone=zone-name --add-source=<source>
```

다음 절차에서는 신뢰할 수 있는 영역에서 **tekton 15** 에서 들어오는 모든 트래픽을 허용합니다.

1. 사용 가능한 모든 영역을 나열합니다.

```
~]# firewall-cmd --get-zones
```

2. 영구 모드의 신뢰할 수 있는 영역에 소스 **IP**를 추가합니다.

```
~]# firewall-cmd --zone=trusted --add-source=192.168.2.15
```

3. 새 설정을 영구적으로 설정합니다.

```
~]# firewall-cmd --runtime-to-permanent
```

5.8.2. 소스 제거

영역에서 소스를 제거하면 해당 영역에서 들어오는 트래픽이 줄어듭니다.

1. 필수 영역에 허용된 소스를 나열합니다.

```
~]# firewall-cmd --zone=zone-name --list-sources
```

2. 영역에서 소스를 영구적으로 제거합니다.

```
~]# firewall-cmd --zone=zone-name --remove-source=<source>
```

3. 새 설정을 영구적으로 설정합니다.

```
~]# firewall-cmd --runtime-to-permanent
```

5.8.3. 소스 포트 추가

origin 포트를 기반으로 트래픽 정렬을 활성화하려면 **--add-source-port** 옵션을 사용하여 소스 포트를 지정합니다. 이를 **--add-source** 옵션과 결합하여 트래픽을 특정 **IP** 주소 또는 **IP** 범위로 제한할 수도 있습니다.

소스 포트를 추가하려면 다음을 수행합니다.

```
~]# firewall-cmd --zone=zone-name --add-source-port=<port-name>/<tcp|udp|sctp|dccp>
```

5.8.4. 소스 포트 제거

소스 포트를 제거하면 원본 포트에 따라 트래픽 정렬을 비활성화합니다.

소스 포트를 제거하려면 다음을 수행합니다.

```
~]# firewall-cmd --zone=zone-name --remove-source-port=<port-name>/<tcp|udp|sctp|dccp>
```

5.8.5. 영역 및 소스를 사용하여 특정 도메인에 대해서만 서비스를 허용

특정 네트워크의 트래픽이 시스템의 서비스를 사용하도록 허용하려면 영역 및 소스를 사용합니다. 다음 절차에서는 **192.0.2.0/24** 네트워크의 **HTTP** 트래픽만 허용하고 다른 트래픽은 차단됩니다.

주의

*이 시나리오를 구성할 때 기본 대상이 있는 영역을 사용합니다. 대상이 **ACCEPT**로 설정된 영역을 사용하는 것은 **192.0.2.0/24**의 트래픽 때문에 모든 네트워크 연결이 수락되므로 보안 위험이 있습니다.*

1. *사용 가능한 모든 영역을 나열합니다.*

```
~]# firewall-cmd --get-zones
block dmz drop external home internal public trusted work
```

2. ***IP** 범위를 내부 영역에 추가하여 소스에서 영역까지 발생하는 트래픽을 라우팅합니다.*

```
~]# firewall-cmd --zone=internal --add-source=192.0.2.0/24
```

3. ***HTTP** 서비스를 내부 영역에 추가합니다.*

```
~]# firewall-cmd --zone=internal --add-service=http
```

4. *새 설정을 영구적으로 설정합니다.*

```
~]# firewall-cmd --runtime-to-permanent
```

5. *내부 영역이 활성화되어 있고 서비스에서 허용되는지 확인합니다.*

```
~]# firewall-cmd --zone=internal --list-all
internal (active)
  target: default
  icmp-block-inversion: no
  interfaces:
```

```
sources: 192.0.2.0/24
services: dhcpv6-client mdns samba-client ssh http
...
```

5.8.6. 프로토콜 기반 영역에 의해 허용되는 트래픽 구성

프로토콜을 기반으로 한 영역에서 들어오는 트래픽을 수락하도록 허용할 수 있습니다. 지정된 프로토콜을 사용하는 모든 트래픽은 영역에 의해 허용되며, 이 영역에서는 추가 규칙과 필터링을 적용할 수 있습니다.

영역에 프로토콜 추가

특정 영역에 프로토콜을 추가하면 이 프로토콜을 사용하는 모든 트래픽을 이 영역에서 승인할 수 있습니다.

영역에 프로토콜을 추가하려면 다음을 수행합니다.

```
~]# firewall-cmd --zone=zone-name --add-protocol=port-name/tcp|udp|sctp|dccp|igmp
```

참고

멀티 캐스트 트래픽을 수신하려면 **--add-protocol** 옵션과 함께 **igmp** 값을 사용합니다.

영역에서 프로토콜 제거

특정 영역에서 프로토콜을 제거하면 해당 영역이 이 프로토콜을 기반으로 모든 트래픽 수락을 중지합니다.

영역에서 프로토콜을 제거하려면 다음을 수행합니다.

```
~]# firewall-cmd --zone=zone-name --remove-protocol=port-name/tcp|udp|sctp|dccp|igmp
```

5.9. 포트 전달

firewalld 를 사용하여 시스템의 특정 포트에 도달하는 들어오는 트래픽이 다른 내부 포트 또는 다른 시스템의 외부 포트로 전달되도록 포트 리디렉션을 설정할 수 있습니다.

5.9.1. 리디렉션할 포트 추가

한 포트에서 다른 포트로 트래픽을 리디렉션하기 전에 패킷의 포트, 사용된 프로토콜 및 리디렉션 위치 등 세 가지 사항을 알아야 합니다.

포트를 다른 포트로 리디렉션하려면 다음을 수행합니다.

```
~]# firewall-cmd --add-forward-port=port=port-number:proto=tcp|udp|sctp|dccp:toport=port-number
```

*포트를 다른 **IP** 주소의 다른 포트로 리디렉션하려면 다음을 수행합니다.*

1. *전달할 포트를 추가합니다.*

```
~]# firewall-cmd --add-forward-port=port=port-number:proto=tcp|udp:toport=port-number:toaddr=IP
```

2. ***masquerade**를 활성화합니다.*

```
~]# firewall-cmd --add-masquerade
```

*예 **5.1.** 동일한 머신의 **TCP** 포트 **80**을 포트 **88**로 리디렉션*

포트를 리디렉션하려면 다음을 수행합니다.

1. ***TCP** 트래픽의 경우 포트 **80**을 포트 **88**로 리디렉션합니다.*

```
~]# firewall-cmd --add-forward-port=port=80:proto=tcp:toport=88
```

2. *새 설정을 영구적으로 설정합니다.*

```
~]# firewall-cmd --runtime-to-permanent
```

3. *포트가 리디렉션되었는지 확인합니다.*

```
~]# firewall-cmd --list-all
```

5.9.2. 리디렉션된 포트 제거

리디렉션된 포트를 제거하려면 다음을 수행합니다.

```
~]# firewall-cmd --remove-forward-port=port=port-number:proto=<tcp|udp>:toport=port-
number:toaddr=<IP>
```

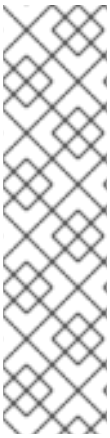
다른 주소로 리디렉션된 전달된 포트를 제거하려면 다음을 수행합니다.

1. 전달된 포트를 제거합니다.

```
~]# firewall-cmd --remove-forward-port=port=port-number:proto=<tcp|udp>:toport=port-
number:toaddr=<IP>
```

2. **masquerade**를 비활성화합니다.

```
~]# firewall-cmd --remove-masquerade
```



참고

이 방법을 사용하여 포트를 리디렉션하는 것은 **IPv4** 기반 트래픽에서만 작동합니다. **IPv6** 리디렉션 설정의 경우 리치 규칙을 사용해야 합니다. 자세한 내용은 **5.15**절. “"Rich Language" 구문을 사용하여 복잡한 방화벽 규칙 구성”의 내용을 참조하십시오.

외부 시스템으로 리디렉션하려면 마스커레이딩을 활성화해야 합니다. 자세한 내용은 **5.10**절. “IP 주소 Masquerading 구성”의 내용을 참조하십시오.

예 5.2. 동일한 머신의 포트 **88**로 전달된 **TCP** 포트 **80** 제거

포트 리디렉션을 제거하려면 다음을 수행합니다.

1. 리디렉션된 포트를 나열합니다.

```
~]# firewall-cmd --list-forward-ports
port=80:proto=tcp:toport=88:toaddr=
```

2. 방화벽에서 리디렉션된 포트를 제거합니다.

```
~]# firewall-cmd --remove-forward-port=port=80:proto=tcp:toport=88:toaddr=
```

3. 새 설정을 영구적으로 설정합니다.

```
~]# firewall-cmd --runtime-to-permanent
```

5.10. IP 주소 MASQUERADING 구성

IP 마스커레이딩은 한 컴퓨터가 네트워크에 대한 IP 게이트웨이 역할을 하는 프로세스입니다. 마스커레이딩의 경우 게이트웨이는 나가는 인터페이스의 IP를 항상 동적으로 조회하고 패킷의 소스 주소를 이 주소로 대체합니다.

발신 인터페이스의 IP가 변경될 수 있는 경우 마스커레이딩을 사용합니다. 마스커레이딩의 일반적인 사용 사례는 라우터가 인터넷에서 라우팅되지 않는 개인 IP 주소를 라우터에서 발신 인터페이스의 공용 동적 IP 주소로 교체하는 경우입니다.

IP 마스커레이딩이 활성화되어 있는지 확인하려면 (예: 외부 영역의 경우) 다음 명령을 root 로 입력합니다.

```
~]# firewall-cmd --zone=external --query-masquerade
```

이 명령은 활성화된 경우 종료 상태 0 으로 yes 를 출력합니다. 그렇지 않으면 종료 상태 1 로 no 를 출력합니다. zone 이 생략되면 기본 영역이 사용됩니다.

IP 마스커레이딩을 활성화하려면 root 로 다음 명령을 입력합니다.

```
~]# firewall-cmd --zone=external --add-masquerade
```

이 설정을 영구적으로 설정하려면 --permanent 옵션을 추가한 명령을 반복합니다.

IP 마스커레이딩을 비활성화하려면 root 로 다음 명령을 입력합니다.

```
~]# firewall-cmd --zone=external --remove-masquerade
```

이 설정을 영구적으로 설정하려면 **--permanent** 옵션을 추가한 명령을 반복합니다.

자세한 내용은 다음을 참조하십시오.

- 6.3.1절. “다양한 NAT 유형: 마스커레이딩, 소스 NAT, 대상 NAT 및 리디렉션”
- 6.3.2절. “nftables를 사용하여 마스커레이딩 구성”

5.11. ICMP 요청 관리

ICMP(Internet Control Message Protocol)는 다양한 네트워크 장치에서 요청 서비스를 사용할 수 없다는 연결 문제를 나타내는 오류 메시지 및 운영 정보를 보내는 데 사용하는 지원 프로토콜입니다. **ICMP**는 시스템 간에 데이터를 교환하는 데 사용되지 않기 때문에 **TCP** 및 **UDP**와 같은 전송 프로토콜과 다릅니다.

유감스럽게도 **ICMP** 메시지, 특히 **echo-request** 및 **echo-reply** 를 사용하여 네트워크에 대한 정보를 공개하고 다양한 종류의 부정 활동을 위해 이러한 정보를 오용할 수 있습니다. 따라서 **firewalld** 를 사용하면 **ICMP** 요청을 차단하여 네트워크 정보를 보호할 수 있습니다.

5.11.1. ICMP 요청 나열

ICMP 요청은 **/usr/lib/firewalld/icmptypes/** 디렉터리에 있는 개별 **XML** 파일에 설명되어 있습니다. 이러한 파일을 읽고 요청에 대한 설명을 볼 수 있습니다. **firewall-cmd** 명령은 **ICMP** 요청 조작을 제어합니다.

사용 가능한 모든 **ICMP** 유형을 나열하려면 다음을 수행하십시오.

```
~]# firewall-cmd --get-icmptypes
```

ICMP 요청은 **IPv4, IPv6** 또는 두 프로토콜 모두에서 사용할 수 있습니다. **ICMP** 요청이 사용되는 프로토콜을 보려면 다음을 수행합니다.

```
~]# firewall-cmd --info-icmptype=<icmptype>
```

ICMP 요청의 상태는 요청이 현재 차단되었거나 그렇지 않은 경우 **no** 인 경우 **yes** 를 표시합니다.

ICMP 요청이 현재 차단되었는지 확인하려면 다음을 수행하십시오.

```
~]# firewall-cmd --query-icmp-block=<icmptype>
```

5.11.2. ICMP 요청 차단 또는 차단 해제

서버가 **ICMP** 요청을 차단하면 일반적으로 수행할 정보를 제공하지 않습니다. 그러나 이것이 어떠한 정보도 전혀 제공되지 않는다는 의미는 아닙니다. 클라이언트는 특정 **ICMP** 요청이 차단되는 정보를 수신합니다(거부). **ICMP** 요청을 차단하는 것은 특히 **IPv6** 트래픽에서 통신 문제를 일으킬 수 있으므로 신중하게 고려해야 합니다.

ICMP 요청이 현재 차단되었는지 확인하려면 다음을 수행하십시오.

```
~]# firewall-cmd --query-icmp-block=<icmptype>
```

ICMP 요청을 차단하려면 다음을 수행합니다.

```
~]# firewall-cmd --add-icmp-block=<icmptype>
```

ICMP 요청 블록을 제거하려면 다음을 수행합니다.

```
~]# firewall-cmd --remove-icmp-block=<icmptype>
```

5.11.3. 모든 정보를 제공하지 않고 ICMP 요청을 차단

일반적으로 **ICMP** 요청을 차단하면 클라이언트는 이를 차단하고 있음을 알고 있습니다. 따라서 라이브 **IP** 주소를 스니핑하는 잠재적 공격자는 여전히 **IP** 주소가 온라인 상태임을 확인할 수 있습니다. 이 정보를 완전히 숨기려면 모든 **ICMP** 요청을 삭제해야 합니다.

모든 **ICMP** 요청을 차단 및 삭제하려면 다음을 수행합니다.

1. 영역의 대상을 **DROP** 으로 설정합니다.

   ```
   ~]# firewall-cmd --set-target=DROP
   ```

2. 새 설정을 영구적으로 설정합니다.

```
~]# firewall-cmd --runtime-to-permanent
```

*이제 명시적으로 허용된 트래픽을 제외하고 **ICMP** 요청을 포함한 모든 트래픽이 삭제됩니다.*

*특정 **ICMP** 요청을 차단 및 삭제하고 다른 요청을 허용하려면 다음을 수행합니다.*

1. *영역의 대상을 **DROP** 으로 설정합니다.*

```
~]# firewall-cmd --set-target=DROP
```

2. ***ICMP** 블록 인버전을 추가하여 모든 **ICMP** 요청을 한 번에 차단합니다.*

```
~]# firewall-cmd --add-icmp-block-inversion
```

3. *허용할 **ICMP** 요청에 대해 **ICMP** 블록을 추가합니다.*

```
~]# firewall-cmd --add-icmp-block=<icmptype>
```

4. *새 설정을 영구적으로 설정합니다.*

```
~]# firewall-cmd --runtime-to-permanent
```

*블록 **inversion** 은 **ICMP** 요청 블록의 설정을 반전하므로 이전에 차단되지 않은 모든 요청이 차단됩니다. 차단된 사람들은 차단되지 않습니다. 즉, 요청을 차단 해제해야 하는 경우 **blocking** 명령을 사용해야 합니다.*

이를 완전히 허용 설정으로 되돌리려면 다음을 수행합니다.

1. *영역의 대상을 **default** 또는 **ACCEPT** 로 설정합니다.*

```
~]# firewall-cmd --set-target=default
```

2. ***ICMP** 요청에 대해 추가된 모든 블록을 제거합니다.*

```
~]# firewall-cmd --remove-icmp-block=<icmptype>
```

3. **ICMP** 블록 버전 제거:

```
~]# firewall-cmd --remove-icmp-block-inversion
```

4. 새 설정을 영구적으로 설정합니다.

```
~]# firewall-cmd --runtime-to-permanent
```

5.11.4. GUI를 사용하여 ICMP 필터 구성

ICMP 필터를 활성화하거나 비활성화하려면 **firewall-config** 도구를 시작하고 메시지가 필터링될 네트워크 영역을 선택합니다. **ICMP** 필터 탭을 선택하고 필터링할 각 **ICMP** 메시지 유형의 확인란을 선택합니다. 필터를 비활성화하려면 확인란을 지웁니다. 이 설정은 방향에 따라 설정되며 기본값은 모든 것을 허용합니다.

ICMP 필터 를 반전하려면 오른쪽에 있는 반전 필터 확인란을 클릭합니다. 이제 표시된 **ICMP** 유형만 수락되며 다른 모든 유형은 거부됩니다. **DROP** 대상을 사용하는 영역에서는 삭제됩니다.

5.12. FIREWALLD를 사용하여 IP 세트 설정 및 제어

firewalld 에서 지원하는 **IP** 세트 유형 목록을 보려면 **root**로 다음 명령을 입력합니다.

```
~]# firewall-cmd --get-ipset-types
hash:ip hash:ip,mark hash:ip,port hash:ip,port,ip hash:ip,port,net hash:mac hash:net hash:net,iface
hash:net,net hash:net,port hash:net,port,net
```

5.12.1. 명령줄 클라이언트를 사용하여 IP 설정 옵션 구성

IP 세트는 **firewalld** 영역에서 소스로 사용할 수도 있고 리치 규칙의 소스로도 사용할 수 있습니다. **Red Hat Enterprise Linux 7**에서 기본 방법은 직접 규칙에 **firewalld** 로 생성된 **IP** 세트를 사용하는 것입니다.

영구 환경에서 **firewalld** 에 알려진 **IP** 세트를 나열하려면 다음 명령을 **root** 로 사용하십시오.

```
~]# firewall-cmd --permanent --get-ipsets
```

새 **IP** 세트를 추가하려면 영구 환경을 **root** 로 사용하여 다음 명령을 사용하십시오.

```
~]# firewall-cmd --permanent --new-ipset=test --type=hash:net
success
```

이전 명령은 **test** 라는 이름과 **IPv4** 에 대한 **hash:net** 유형으로 새 **IP** 세트를 생성합니다. **IPv6** 에 사용할 **IP** 세트를 만들려면 **--option=family=inet6** 옵션을 추가합니다. 런타임 환경에서 새 설정을 적용하려면 **firewalld** 를 다시 로드합니다. 다음 명령을 **root** 로 사용하여 새 **IP** 세트를 나열합니다.

```
~]# firewall-cmd --permanent --get-ipsets
test
```

IP 세트에 대한 자세한 내용을 보려면 **root** 로 다음 명령을 사용하십시오.

```
~]# firewall-cmd --permanent --info-ipset=test
test
type: hash:net
options:
entries:
```

현재 **IP** 세트에는 항목이 없습니다. 테스트 **IP** 세트에 항목을 추가하려면 **root** 로 다음 명령을 사용하십시오.

```
~]# firewall-cmd --permanent --ipset=test --add-entry=192.168.0.1
success
```

이전 명령은 **IP** 주소 **192.168.0.1** 을 **IP** 세트에 추가합니다. **IP** 세트의 현재 항목 목록을 가져오려면 다음 명령을 **root** 로 사용합니다.

```
~]# firewall-cmd --permanent --ipset=test --get-entries
192.168.0.1
```

IP 주소 목록이 포함된 파일을 생성합니다. 예를 들면 다음과 같습니다.

```
~]# cat > iplist.txt <<EOL
192.168.0.2
192.168.0.3
192.168.1.0/24
192.168.2.254
EOL
```

IP 세트의 IP 주소 목록이 있는 파일에는 행당 항목이 포함되어야 합니다. 해시, 부분 또는 빈 줄로 시작하는 줄은 무시됩니다.

iplist.txt 파일에서 주소를 추가하려면 다음 명령을 root 로 사용하십시오.

```
~]# firewall-cmd --permanent --ipset=test --add-entries-from-file=iplist.txt
success
```

IP 세트의 확장 항목 목록을 보려면 다음 명령을 root 로 사용합니다.

```
~]# firewall-cmd --permanent --ipset=test --get-entries
192.168.0.1
192.168.0.2
192.168.0.3
192.168.1.0/24
192.168.2.254
```

IP 세트에서 주소를 제거하고 업데이트된 항목 목록을 확인하려면 root 로 다음 명령을 사용하십시오.

```
~]# firewall-cmd --permanent --ipset=test --remove-entries-from-file=iplist.txt
success
~]# firewall-cmd --permanent --ipset=test --get-entries 192.168.0.1
```

IP 세트를 영역에 소스로 추가하여 영역으로 IP 세트에 나열된 주소에서 들어오는 모든 트래픽을 처리할 수 있습니다. 예를 들어 테스트 IP 세트를 드롭다운 영역에 소스로 추가하려면 테스트 IP 세트에 나열된 모든 항목에서 들어오는 모든 패킷을 삭제하려면 root 로 다음 명령을 사용합니다.

```
~]# firewall-cmd --permanent --zone=drop --add-source=ipset:test
success
```

소스의 ipset: 접두사에는 소스가 IP 주소 또는 주소 범위가 아닌 firewalld 가 표시됩니다.

IP 세트 생성 및 제거만 영구 환경으로 제한되며, --permanent 옵션 없이 런타임 환경에서 다른 모든 IP 세트 옵션도 사용할 수 있습니다.

5.12.2. IP 세트에 대한 사용자 정의 서비스 구성

firewalld 를 시작하기 전에 IP 세트 구조를 생성하고 로드하도록 사용자 지정 서비스를 구성하려면 다음을 수행합니다.

1. ***root** 로 실행되는 편집기를 사용하여 다음과 같이 파일을 생성합니다.*

```
~]# vi /etc/systemd/system/ipset_name.service
[Unit]
Description=ipset_name
Before=firewalld.service

[Service]
Type=oneshot
RemainAfterExit=yes
ExecStart=/usr/local/bin/ipset_name.sh start
ExecStop=/usr/local/bin/ipset_name.sh stop

[Install]
WantedBy=basic.target
```

2. ***firewalld** 에서 IP 세트를 영구적으로 사용합니다.*

```
~]# vi /etc/firewalld/direct.xml
<?xml version="1.0" encoding="utf-8"?>
<direct>
 <rule ipv="ipv4" table="filter" chain="INPUT" priority="0">-m set --match-set
<replaceable>ipset_name</replaceable> src -j DROP</rule>
</direct>
```

3. *변경 사항을 활성화하려면 **firewalld** 재로드가 필요합니다.*

```
~]# firewall-cmd --reload
```

*이렇게 하면 상태 정보(**TCP** 세션이 종료되지 않음)가 방화벽을 다시 로드하지 않지만 다시 로드하는 동안 서비스 중단이 가능합니다.*

주의

firewalld 를 통해 관리되지 않는 **IP** 세트는 사용하지 않는 것이 좋습니다. 이러한 **IP** 세트를 사용하려면 세트를 참조하려면 영구 직접 규칙이 필요하며 이러한 **IP** 세트를 생성하려면 사용자 지정 서비스를 추가해야 합니다. **firewalld**를 시작하기 전에 이 서비스를 시작해야 합니다. 그러지 않으면 **firewalld** 가 이러한 세트를 사용하여 직접 규칙을 추가할 수 없습니다. **/etc/firewalld/direct.xml** 파일을 사용하여 영구적인 직접 규칙을 추가할 수 있습니다.

5.13. IPTABLES를 사용하여 IP 세트 설정 및 제어

firewalld 와 **iptables** (및 **ip6tables**) 서비스의 근본적인 차이점은 다음과 같습니다.

- **iptables** 서비스는 **/etc/sysconfig/iptables** 및 **/etc/sysconfig/ip6tables** 에 구성을 저장하지만 **firewalld** 는 이를 **/usr/lib/firewalld/** 및 **/etc/firewalld/** 의 다양한 **XML** 파일에 저장합니다. **firewalld** 는 **Red Hat Enterprise Linux**에 기본적으로 설치되므로 **/etc/sysconfig/iptables** 파일이 존재하지 않습니다.

- **iptables** 서비스에서는 모든 변경 사항을 통해 이전 규칙을 모두 플러시하고 **/etc/sysconfig/iptables** 에서 모든 새 규칙을 읽고, **firewalld** 를 사용하면 모든 규칙을 다시 생성하지 않습니다. 차이점만 적용됩니다. 결과적으로 **firewalld** 는 기존 연결이 손실되지 않고 런타임 중에 설정을 변경할 수 있습니다.

둘 다 **iptables** 툴 을 사용하여 커널 패킷 필터와 통신합니다.

firewalld 대신 **iptables** 및 **ip6tables** 서비스를 사용하려면 먼저 **root** 로 다음 명령을 실행하여 **firewalld** 를 비활성화합니다.

```
~]# systemctl disable firewalld
~]# systemctl stop firewalld
```

그런 다음 **root** 로 다음 명령을 입력하여 **iptables-services** 패키지를 설치합니다.

```
~]# yum install iptables-services
```

iptables-services 패키지에는 **iptables** 서비스와 **ip6tables** 서비스가 포함되어 있습니다.

그런 다음 **iptables** 및 **ip6tables** 서비스를 시작하려면 **root** 로 다음 명령을 입력합니다.

```
~]# systemctl start iptables
~]# systemctl start ip6tables
```

모든 시스템에서 서비스를 시작할 수 있도록 하려면 다음 명령을 입력합니다.

```
~]# systemctl enable iptables
~]# systemctl enable ip6tables
```

ipset 유틸리티는 **Linux** 커널의 **IP** 세트를 관리하는 데 사용됩니다. **IP** 세트는 **IP** 주소, 포트 번호, **IP** 및 **MAC** 주소 쌍 또는 **IP** 주소와 포트 번호 쌍을 저장하기 위한 프레임워크입니다. 집합이 매우 큰 경우에도 집합에 대해 빠르게 일치할 수 있는 방식으로 집합이 인덱싱됩니다. **IP** 세트를 사용하면 더 간단하고 관리가 용이한 구성으로 **iptables** 를 사용할 때 성능상의 이점을 제공할 수 있습니다. **iptables** 일치 및 대상을 참조하여 커널에서 지정된 세트를 보호하는 참조를 생성합니다. 세트는 이를 가리키는 단일 참조가 있는 동안에는 삭제할 수 없습니다.

ipset 을 사용하면 다음과 같은 **iptables** 명령을 세트로 교체할 수 있습니다.

```
~]# iptables -A INPUT -s 10.0.0.0/8 -j DROP
~]# iptables -A INPUT -s 172.16.0.0/12 -j DROP
~]# iptables -A INPUT -s 192.168.0.0/16 -j DROP
```

세트는 다음과 같이 생성됩니다.

```
~]# ipset create my-block-set hash:net
~]# ipset add my-block-set 10.0.0.0/8
~]# ipset add my-block-set 172.16.0.0/12
~]# ipset add my-block-set 192.168.0.0/16
```

그런 다음 **set**가 다음과 같이 **iptables** 명령에서 참조됩니다.

```
~]# iptables -A INPUT -m set --set my-block-set src -j DROP
```

세트가 한 번 이상 사용된 경우 구성 시간에 저장합니다. 집합에 많은 항목이 포함된 경우 처리 시간에 저장 시간이 생성됩니다.**If the set contains many entries a saving in processing time is made.**

5.14. 직접 인터페이스 사용

firewall-cmd 도구와 함께 **--direct** 옵션을 사용하여 런타임 중에 체인을 추가하고 제거할 수 있습니다. 여기에 몇 가지 예제가 나와 있습니다. 자세한 내용은 **firewall-cmd(1)** 매뉴얼 페이지를 참조하십시오.

iptables 에 익숙하지 않은 경우 실수로 방화벽에서 위반을 일으킬 수 있으므로 직접 인터페이스를 사용하는 것은 위험합니다.

직접 인터페이스 모드는 런타임 중에 서비스 또는 애플리케이션에서 특정 방화벽 규칙을 추가하기 위한 것입니다. **firewall-cmd --permanent --direct** 명령을 사용하거나 **/etc/firewalld/direct.xml** 을 수정하여 **--permanent** 옵션을 추가하여 규칙을 영구적으로 만들 수 있습니다. **/etc/firewalld/direct.xml** 파일에 대한 정보는 **man firewalld.direct(5)** 를 참조하십시오.

5.14.1. 직접 인터페이스를 사용하여 규칙 추가

"IN_public_allow" 체인에 규칙을 추가하려면 다음 명령을 **root** 로 입력합니다.

```
~]# firewall-cmd --direct --add-rule ipv4 filter IN_public_allow \
      0 -m tcp -p tcp --dport 666 -j ACCEPT
```

--permanent 옵션을 추가하여 설정을 영구적으로 설정합니다.

5.14.2. 직접 인터페이스를 사용하여 규칙 제거

"IN_public_allow" 체인에서 규칙을 제거하려면 **root** 로 다음 명령을 입력합니다.

```
~]# firewall-cmd --direct --remove-rule ipv4 filter IN_public_allow \
      0 -m tcp -p tcp --dport 666 -j ACCEPT
```

--permanent 옵션을 추가하여 설정을 영구적으로 설정합니다.

5.14.3. 직접 인터페이스를 사용하여 규칙 나열

"IN_public_allow" 체인의 규칙을 나열하려면 다음 명령을 **root** 로 입력합니다.

```
~]# firewall-cmd --direct --get-rules ipv4 filter IN_public_allow
```

이 명령(**-get-rules** 옵션)은 **-- add-rule** 옵션을 사용하여 이전에 추가한 규칙만 나열합니다. 다른 방법으로 추가한 기존 **iptables** 규칙은 나열되지 않습니다.

5.15. "RICH LANGUAGE" 구문을 사용하여 복잡한 방화벽 규칙 구성

"풍부한 언어" 구문을 사용하면 **direct-interface** 메서드보다 이해하기 쉬운 방식으로 복잡한 방화벽 규칙을 만들 수 있습니다. 또한 설정은 영구적으로 만들 수 있습니다. 언어에서는 값이 있는 키워드를 사용하며 **iptables** 규칙의 추상 표현입니다. 이 언어를 사용하여 영역을 구성할 수 있습니다. 현재 구성 방법은 계속 지원됩니다.

5.15.1. rich language 명령의 형식 지정

이 섹션의 모든 명령을 **root** 로 실행해야 합니다. 규칙을 추가하는 명령의 형식은 다음과 같습니다.

```
firewall-cmd [--zone=zone] --add-rich-rule='rule' [--timeout=timeval]
```

그러면 영역 영역에 대한 풍부한 언어 규칙 규칙이 추가됩니다. 이 옵션은 여러 번 지정할 수 있습니다. 영역을 생략하면 기본 영역이 사용됩니다. 시간 제한이 제공되면 규칙 또는 규칙은 지정된 시간 동안만 활성 상태를 유지하고 나중에 자동으로 제거됩니다. 시간 값 뒤에 **s** (초), **m** (분) 또는 **h** (시간)을 사용하여 시간 단위를 지정할 수 있습니다. 기본값은 초입니다.

규칙을 제거하려면 다음을 수행합니다.

```
firewall-cmd [--zone=zone] --remove-rich-rule='rule'
```

이렇게 하면 영역 의 다양한 언어 규칙 규칙이 제거됩니다. 이 옵션은 여러 번 지정할 수 있습니다. 영역을 생략하면 기본 영역이 사용됩니다.

규칙이 있는지 확인하려면 다음을 수행합니다.

```
firewall-cmd [--zone=zone] --query-rich-rule='rule'
```

이 명령은 영역 영역에 대해 풍부한 언어 규칙 규칙이 추가되었는지 여부를 반환합니다. 이 명령은 활성화된 경우 종료 상태 **0** 으로 **yes** 를 출력합니다. 그렇지 않으면 종료 상태 **1** 로 **no** 를 출력합니다. 영역을 생략하면 기본 영역이 사용됩니다.

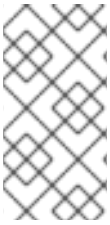
영역 구성 파일에서 사용되는 다양한 언어 표현에 대한 자세한 내용은 **firewalld.zone(5)** 도움말 페이

지를 참조하십시오.

5.15.2. rich Rule Structure 이해

풍부한 규칙 명령의 형식 또는 구조는 다음과 같습니다.

```
rule [family="rule family"]
  [ source [NOT] [address="address"] [mac="mac-address"] [ipset="ipset"] ]
  [ destination [NOT] address="address" ]
  [ element ]
  [ log [prefix="prefix text"] [level="log level"] [limit value="rate/duration"] ]
  [ audit ]
  [ action ]
```



참고

파일의 리치 규칙의 구조는 **NOT** 키워드를 사용하여 소스 및 대상 주소 명령의 감지를 반전하지만 명령줄에는 **invert="true"** 옵션을 사용합니다.

규칙은 특정 영역과 연결되어 있습니다. 하나의 영역에는 여러 규칙이 있을 수 있습니다. 일부 규칙이 상호 작용하거나 모순되는 경우 패킷과 일치하는 첫 번째 규칙이 적용됩니다.

5.15.3. rich Rule 명령 옵션 이해

제품군

규칙 제품군(**ipv4** 또는 **ipv 6**)이 제공되는 경우 규칙을 각각 **IPv4** 또는 **IPv6** 로 제한합니다. 규칙 제품군이 제공되지 않으면 **IPv4** 및 **IPv6** 모두에 대한 규칙이 추가됩니다. 규칙에서 소스 또는 대상 주소를 사용하는 경우 규칙 제품군을 제공해야 합니다. 이는 포트 전달의 경우에도 마찬가지입니다.

소스 및 대상 주소

소스

소스 주소를 지정하면 연결 시도의 원본을 소스 주소로 제한할 수 있습니다.**By specifying the source address, the origin of a connection attempt can be limited to the source address.** 소스 주소 또는 주소 범위는 **IPv4** 또는 **IPv6** 용 마스크가 있는 **IP** 주소 또는 네트워크 **IP** 주소입니다. **IPv4** 의 경우 마스크는 네트워크 마스크 또는 일반 번호일 수 있습니다. **IPv6** 의 경우 마스크는 일반 번호입니다. 호스트 이름 사용은 지원되지 않습니다. **NOT** 키워드를 추가하여 소스 주소 명령의 감각을 반전할 수 있습니다. 제공된 주소는 모두 일치합니다.

*규칙에 대해 지정되지 않은 경우 **MAC** 주소 및 **hash:mac** 유형의 **IP** 세트도 **IPv4** 및 **IPv6** 에 대해 추가할 수 있습니다. 다른 **IP** 세트는 규칙의 제품군 설정과 일치해야 합니다.*

대상

*대상 주소를 지정하면 대상을 대상 주소로 제한할 수 있습니다. 대상 주소는 **IP** 주소 또는 주소 범위의 소스 주소와 동일한 구문을 사용합니다. 소스 및 대상 주소 사용은 선택 사항이며 모든 요소에서 대상 주소를 사용할 수 없습니다. 이는 대상 주소 사용에 따라 달라집니다(예: 서비스 항목에서). 대상과 조치를 결합할 수 있습니다.*

elements

*요소는 서비스, 포트 ,프로토콜,**masquerade**,**icmp-block**,**forward-port** , **source-port** 등 요소 유형 중 하나일 수 있습니다.*

service

***service** 요소는 **firewalld** 제공 서비스 중 하나입니다. 사전 정의된 서비스 목록을 가져오려면 다음 명령을 입력합니다.*

```
~]$ firewall-cmd --get-services
```

서비스에서 대상 주소를 제공하는 경우 규칙의 대상 주소와 충돌하여 오류가 발생합니다. 내부적으로 대상 주소를 사용하는 서비스는 대부분 멀티 캐스트를 사용하는 서비스입니다. 명령은 다음 형식을 사용합니다.

```
service name=service_name
```

port

*포트 요소는 단일 포트 번호 또는 포트 범위 (예: 5060-50 62) 중 하나이거나 프로토콜을 **tcp** 또는 **udp** 일 수 있습니다. 명령은 다음 형식을 사용합니다.*

```
port port=number_or_range protocol=protocol
```

프로토콜

***protocol** 값은 프로토콜 **ID** 번호 또는 프로토콜 이름일 수 있습니다. 허용된 프로토콜 항목에 대해서는 **/etc/protocols** 를 참조하십시오. 명령은 다음 형식을 사용합니다.*

```
protocol value=protocol_name_or_ID
```

icmp-block

하나 이상의 **ICMP** 유형을 차단하려면 이 명령을 사용합니다. **ICMP** 유형은 **firewalld** 가 지원하는 **ICMP** 유형 중 하나입니다. 지원되는 **ICMP** 유형 목록을 가져오려면 다음 명령을 입력합니다.

```
~]$ firewall-cmd --get-icmptypes
```

여기서 작업을 지정하는 것은 허용되지 않습니다. **icmp-block** 은 내부적으로 거부되는 작업을 사용합니다. 명령은 다음 형식을 사용합니다.

```
icmp-block name=icmptype_name
```

masquerade

규칙에서 **IP** 마스커레이딩을 켭니다. 이 영역으로 마스커레이딩을 제한하기 위해 소스 주소를 제공할 수 있지만 대상 주소는 제공할 수 없습니다. 여기서 작업을 지정할 수 없습니다.

forward-port

tcp 또는 **udp** 로 지정된 프로토콜을 사용하여 로컬 포트에서 로컬 포트, 다른 시스템 또는 다른 시스템의 다른 포트로 패킷을 전달합니다. 포트 및 포트 는 단일 포트 번호 또는 포트 범위일 수 있습니다. 대상 주소는 간단한 **IP** 주소입니다. 여기서 작업을 지정할 수 없습니다. **forward-port** 명령은 내부적으로 수락하는 작업을 사용합니다. 명령은 다음 형식을 사용합니다.

```
forward-port port=number_or_range protocol=protocol /
        to-port=number_or_range to-addr=address
```

source-port

패킷의 소스 포트, 즉 연결 시도의 시작에 사용되는 포트와 일치합니다. 현재 머신의 포트를 일치시키려면 **port** 요소를 사용합니다. 소스 포트 요소는 단일 포트 번호 또는 포트 범위 **(예: 5060-5062)** 및 프로토콜을 **tcp** 또는 **udp** 일 수 있습니다. 명령은 다음 형식을 사용합니다.

```
source-port port=number_or_range protocol=protocol
```

로깅

log

syslog에서 커널 로깅을 사용하여 규칙을 새 연결 시도를 기록합니다. 로그 메시지에 접두사로 추가할 접두사 텍스트를 정의할 수 있습니다. 로그 수준은 **emerg,alert,crit,error,warning,notice,info, debug** 중 하나일 수 있습니다. 로그 사용은 선택 사항입니다. 다음과 같이 로깅을 제한할 수 있습니다.

```
log [prefix=prefix text] [level=log level] limit value=rate/duration
```

이 비율은 **s,m,h,d** 를 가진 자연 양의 양의 수 **[1, ..]**입니다.**s means seconds, m minutes, h means hours, d days.** 최대 제한 값은 **1/d** 이며, 이는 하루에 최대 **1**개의 로그 항목을 의미합니다.

audit

audit은 **service auditd** 로 전송된 감사 레코드를 사용하여 로깅하는 다른 방법을 제공합니다. 감사 유형은 **ACCEPT,REJECT** 또는 **DROP** 중 하나일 수 있지만, 감사 유형이 규칙 작업에서 자동으로 수집되므로 명령 감사 후에는 지정하지 않습니다. 감사에는 자체 매개 변수가 없지만 필요에 따라 제한을 추가할 수 있습니다. **audit**을 사용하는 것은 선택 사항입니다.

동작

accept|reject|drop|mark

작업은 수락,거부,삭제 또는 표시 중 하나일 수 있습니다. 규칙에는 요소 또는 소스만 포함할 수 있습니다. 규칙에 요소가 포함된 경우 요소와 일치하는 새 연결이 작업과 함께 처리됩니다. 규칙에 소스가 포함된 경우 지정된 작업을 사용하여 소스 주소의 모든 내용이 처리됩니다.

```
accept | reject [type=reject type] | drop | mark set="mark[/mask]"
```

accept 를 사용하면 모든 새 연결 시도가 부여됩니다. 거부 된 경우 해당 소스가 거부된 메시지를 받게 됩니다. 거부 유형은 다른 값을 사용하도록 설정할 수 있습니다. 드롭 다운 을 사용하면 모든 패킷이 즉시 삭제되고 정보가 소스로 전송되지 않습니다. 마크 를 모두 지정하면 모든 패킷이 지정된 마크 와 선택적 마스크 로 표시됩니다.

5.15.4. rich Rule Log 명령 사용

logging은 **Netfilter** 로그 대상 및 감사 대상을 사용하여 수행할 수 있습니다. 새 체인은 형식 영역 "_log에 이름이 있는 모든영역에" 추가됩니다. 여기서 **zone** 은 영역 이름입니다. 이 작업은 거부 체인이 올바른 순서를 가지기 전에 처리됩니다. 규칙 또는 부분은 다음과 같이 규칙의 동작에 따라 별도의 체인에 배치됩니다.

```
zone_log
  zone_deny
  zone_allow
```

모든 로깅 규칙은 “영역_log” 체인에 배치되고 먼저 구문 분석됩니다. 모든 거부 및 드롭 규칙이 “영역_deny” 체인에 배치되며 로그 체인 후에 구문 분석됩니다. 모든 수락 규칙은 “영역_allow” chain에 배치되며 거부 체인 후 구문 분석됩니다. 규칙에 로그 가 포함되어 있고 작업을 거부하거나 허용하는 경우 이러한 작업을 지정하는 규칙의 일부가 일치하는 체인에 배치됩니다.

5.15.4.1. Rich Rule Log 명령 예 1

인증 헤더 프로토콜 AH 에 대해 새 IPv4 및 IPv6 연결을 활성화합니다.

```
rule protocol value="ah" accept
```

5.15.4.2. rich Rule Log 명령 예 2 사용

감사를 사용하여 프로토콜 FTP 에 대한 새 IPv4 및 IPv6 연결을 허용하고 분당 1을 로그하십시오.

```
rule service name="ftp" log limit value="1/m" audit accept
```

5.15.4.3. Rich Rule Log 명령 예 3

address 192.168.0.0/24 의 새로운 IPv4 연결 프로토콜 TFTP 및 syslog를 사용하여 1분당 로그 1을 허용하십시오.

```
rule family="ipv4" source address="192.168.0.0/24" service name="tftp" log prefix="tftp"
level="info" limit value="1/m" accept
```

5.15.4.4. rich Rule Log 명령 예 4

프로토콜 RADIUS 의 경우 :2:3:4:6:: 의 새로운 IPv6 연결은 모두 거부되어 분당 3번의 속도로 기록됩니다. 다른 소스의 새 IPv6 연결이 허용됩니다.

```
rule family="ipv6" source address="1:2:3:4:6::" service name="radius" log prefix="dns"
level="info" limit value="3/m" reject
rule family="ipv6" service name="radius" accept
```

5.15.4.5. Rich Rule Log 명령 예 5

프로토콜 **TCP** 가 있는 **4011** 포트에서 **1:2:3:4:6::** 에서 받은 **IPv6** 패킷을 포트 **4012**의 **1::2:3:4:4:7** 로 전달합니다.

```
rule family="ipv6" source address="1:2:3:4:6::" forward-port to-addr="1::2:3:4:7" to-port="4012" protocol="tcp" port="4011"
```

5.15.4.6. Rich Rule Log 명령 예 6

이 소스의 모든 연결을 허용하도록 소스 주소를 허용 목록에 추가합니다.

```
rule family="ipv4" source address="192.168.2.2" accept
```

자세한 내용은 **firewalld.rich language(5)** 매뉴얼 페이지를 참조하십시오.

5.16. 방화벽 잠금 구성

로컬 애플리케이션 또는 서비스는 **root** 로 실행되는 경우 방화벽 구성을 변경할 수 있습니다(예: **libvirt**). 이 기능을 통해 관리자는 방화벽 구성을 잠글 수 있으므로 잠금 해제 화이트리스트에 추가된 애플리케이션이나 방화벽 변경을 요청할 수 있습니다. 잠금 설정은 기본적으로 비활성화되어 있습니다. 활성화하면 로컬 애플리케이션 또는 서비스에서 방화벽에 대한 원하지 않는 구성 변경이 없는지 확인할 수 있습니다.

5.16.1. 명령줄 클라이언트를 사용하여 잠금 구성

잠금이 활성화되었는지 쿼리하려면 다음 명령을 루트로 사용하십시오.

```
~]# firewall-cmd --query-lockdown
```

이 명령은 **lockdown**이 활성화된 경우 종료 상태 **0** 으로 **yes** 를 출력합니다. 그렇지 않으면 종료 상태 **1** 로 **no** 를 출력합니다.

잠금을 활성화하려면 **root** 로 다음 명령을 입력합니다.

```
~]# firewall-cmd --lockdown-on
```

잠금을 비활성화하려면 다음 명령을 **root** 로 사용하십시오.

```
~]# firewall-cmd --lockdown-off
```

5.16.2. 명령줄 클라이언트를 사용하여 잠금 해제 화이트리스트 옵션 구성

잠금 화이트리스트에는 명령, 보안 컨텍스트, 사용자 및 사용자 **ID**가 포함될 수 있습니다. 허용 목록의 명령 항목이 별표 "*" 로 종료되면 해당 명령으로 시작하는 모든 명령 행이 일치합니다. "*" 가 없으면 인수를 포함한 절대 명령이 일치해야 합니다.

컨텍스트는 실행 중인 애플리케이션 또는 서비스의 보안(**SELinux**) 컨텍스트입니다. 실행 중인 애플리케이션의 컨텍스트를 가져오려면 다음 명령을 사용합니다.

```
~]$ ps -e --context
```

이 명령은 실행 중인 모든 애플리케이션을 반환합니다. **grep** 툴을 통해 출력을 파이프하여 관심 있는 애플리케이션을 가져옵니다. 예를 들어 다음과 같습니다.

```
~]$ ps -e --context | grep example_program
```

허용 목록에 있는 모든 명령줄을 나열하려면 **root** 로 다음 명령을 입력합니다.

```
~]# firewall-cmd --list-lockdown-whitelist-commands
```

허용 목록에 명령 명령을 추가하려면 **root** 로 다음 명령을 입력합니다.

```
~]# firewall-cmd --add-lockdown-whitelist-command='/usr/bin/python -Es /usr/bin/command'
```

*허용 목록에서 명령 명령을 제거하려면 **root** 로 다음 명령을 입력합니다.*

```
~]# firewall-cmd --remove-lockdown-whitelist-command='/usr/bin/python -Es /usr/bin/command'
```

*명령 명령이 허용 목록에 있는지 쿼리하려면 다음 명령을 **root** 로 입력합니다.*

```
~]# firewall-cmd --query-lockdown-whitelist-command='/usr/bin/python -Es /usr/bin/command'
```

*이 명령은 **true**인 경우 종료 상태 **0** 으로 **yes** 를 출력합니다. 그렇지 않으면 종료 상태 **1** 로 **no** 를 출력합니다.*

*허용 목록에 있는 모든 보안 컨텍스트를 나열하려면 **root** 로 다음 명령을 입력합니다.*

```
~]# firewall-cmd --list-lockdown-whitelist-contexts
```

*허용 목록에 컨텍스트 컨텍스트 를 추가하려면 **root** 로 다음 명령을 입력합니다.*

```
~]# firewall-cmd --add-lockdown-whitelist-context=context
```

*허용 목록에서 컨텍스트 컨텍스트 를 제거하려면 **root** 로 다음 명령을 입력합니다.*

```
~]# firewall-cmd --remove-lockdown-whitelist-context=context
```

*컨텍스트 컨텍스트 가 허용 목록에 있는지 여부를 쿼리하려면 **root** 로 다음 명령을 입력합니다.*

```
~]# firewall-cmd --query-lockdown-whitelist-context=context
```

종료 상태 **0** 으로 **yes** 를 출력하고, **true**인 경우 종료 상태 **1** 로 **no** 를 출력합니다.

허용 목록에 있는 모든 사용자 **ID**를 나열하려면 **root** 로 다음 명령을 입력합니다.

```
~]# firewall-cmd --list-lockdown-whitelist-uids
```

허용 목록에 사용자 **ID** 를 추가하려면 **root** 로 다음 명령을 입력합니다.

```
~]# firewall-cmd --add-lockdown-whitelist-uid=uid
```

허용 목록에서 사용자 **ID uid** 를 제거하려면 **root** 로 다음 명령을 입력합니다.

```
~]# firewall-cmd --remove-lockdown-whitelist-uid=uid
```

사용자 **ID uid** 가 허용 목록에 있는지 여부를 쿼리하려면 다음 명령을 입력합니다.

```
~]$ firewall-cmd --query-lockdown-whitelist-uid=uid
```

종료 상태 **0** 으로 **yes** 를 출력하고, **true**인 경우 종료 상태 **1** 로 **no** 를 출력합니다.

허용 목록에 있는 모든 사용자 이름을 나열하려면 **root** 로 다음 명령을 입력합니다.

```
~]# firewall-cmd --list-lockdown-whitelist-users
```

허용 목록에 사용자 이름 사용자를 추가하려면 **root** 로 다음 명령을 입력합니다.

```
~]# firewall-cmd --add-lockdown-whitelist-user=user
```

화이트리스트에서 사용자 이름 사용자를 제거하려면 **root** 로 다음 명령을 입력합니다.

```
~]# firewall-cmd --remove-lockdown-whitelist-user=user
```

사용자 이름 사용자가 허용 목록에 있는지 여부를 쿼리하려면 다음 명령을 입력합니다.

```
~]$ firewall-cmd --query-lockdown-whitelist-user=user
```

종료 상태 **0** 으로 **yes** 를 출력하고, **true**인 경우 종료 상태 **1** 로 **no** 를 출력합니다.

5.16.3. 구성 파일을 사용하여 Lockdown Whitelist 옵션 구성

기본 허용 목록 구성 파일에는 **NetworkManager** 컨텍스트와 **libvirt** 의 기본 컨텍스트가 포함되어 있습니다. 사용자 **ID 0**도 목록에 있습니다.

```
<?xml version="1.0" encoding="utf-8"?>
<whitelist>
  <selinux context="system_u:system_r:NetworkManager_t:s0"/>
  <selinux context="system_u:system_r:virtd_t:s0-s0:c0.c1023"/>
  <user id="0"/>
</whitelist>
```

다음은 사용자 **ID**가 **815** 인 사용자의 경우 **firewall-cmd** 유틸리티에 대한 모든 명령을 활성화하는 화이트리스트 구성 파일의 예입니다.

```
<?xml version="1.0" encoding="utf-8"?>
<whitelist>
  <command name="/usr/bin/python -Es /bin/firewall-cmd*"/>
  <selinux context="system_u:system_r:NetworkManager_t:s0"/>
  <user id="815"/>
  <user name="user"/>
</whitelist>
```

이 예제에서는 사용자 **ID** 와 사용자 이름 을 모두 표시하지만 하나의 옵션만 필요합니다. **Python**은 인터프리터이며 명령줄에 앞에 추가됩니다. 특정 명령을 사용할 수도 있습니다. 예를 들면 다음과 같습니다.

```
/usr/bin/python /bin/firewall-cmd --lockdown-on
```

이 예에서는 **--lockdown-on** 명령만 허용됩니다.

참고

Red Hat Enterprise Linux 7에서 모든 유틸리티는 **/usr/bin/** 디렉토리에 배치되고 **/bin/** 디렉토리가 **/usr/bin/** 디렉토리에 자동으로 연결됩니다. 즉, **root** 로 실행할 때 **firewall-cmd** 의 경로가 **/bin/firewall-cmd** 로 확인될 수 있지만**/usr/bin/firewall-cmd** 를 사용할 수 있습니다. 모든 새 스크립트는 새 위치를 사용해야 합니다. 그러나 **root** 로 실행되는 스크립트가 **/bin/firewall-cmd** 경로를 사용하도록 작성된 경우, **/usr/bin/firewall-cmd** 경로 외에도루트 가 아닌 사용자에게만 사용된 명령 경로를 허용 목록에 추가해야 합니다.

명령의 **name** 속성 끝에 있는 "*" 는 이 문자열로 시작하는 모든 명령이 일치합니다. "*" 가 없으면 인수를 포함한 절대 명령이 일치해야 합니다.

5.17. 거부된 패킷에 대한 로깅 구성

firewalld 에서 **LogDenied** 옵션을 사용하면 거부된 패킷에 대한 간단한 로깅 메커니즘을 추가할 수 있습니다. 이는 거부되거나 삭제된 패킷입니다. 로깅 설정을 변경하려면 **/etc/firewalld/firewalld.conf** 파일을 편집하거나 명령줄 또는 **GUI** 구성 도구를 사용합니다.

LogDenied 가 활성화된 경우 기본 규칙에 대해 **INPUT, FORWARD** 및 **OUTPUT** 체인, 영역의 최종 거부 및 삭제 규칙 앞에 있는 로깅 규칙이 바로 추가됩니다. 이 설정에 사용할 수 있는 값은 **all,unicast,broadcast,multicast, off** 입니다. 기본 설정은 **OFF** 입니다. 유니캐스트,브로드캐스트, 멀티캐스트 설정을 사용하면 **pkttype** 이 링크 계층 패킷 유형과 일치하도록 사용됩니다. 모든 패킷이 기록되면 모든 패킷이 기록됩니다.

firewall-cmd를 사용하여 실제 **LogDenied** 설정을 나열하려면 다음 명령을 **root** 로 사용하십시오.

```
~]# firewall-cmd --get-log-denied
off
```

LogDenied 설정을 변경하려면 **root** 로 다음 명령을 사용하십시오.

```
~]# firewall-cmd --set-log-denied=all
success
```

firewalld GUI 구성 도구를 사용하여 **LogDenied** 설정을 변경하려면 **firewall-config** 를 시작합니다. 옵션 메뉴를 클릭하고 **Log Denied** 를 선택합니다. **LogDenied** 창이 나타납니다. 메뉴에서 새 **LogDenied** 설정을 선택하고 **OK**를 클릭합니다.

5.18. 추가 리소스

다음 정보 소스는 **firewalld** 와 관련된 추가 리소스를 제공합니다.

5.18.1. 설치된 문서

- **firewalld(1)** 도움말 페이지 - **firewalld** 에 대한 명령 옵션에 대해 설명합니다.
- **firewalld.conf(5)** 도움말 페이지 - **firewalld** 를 구성할 수 있는 정보가 포함되어 있습니다.
- **firewall-cmd(1)** 도움말 페이지 - **firewalld** 명령줄 클라이언트에 대한 명령 옵션에 대해 설명합니다.
- **firewall-config(1)** 도움말 페이지 - **firewall-config** 도구에 대한 설정 설명.
- **firewall-offline-cmd(1)** 매뉴얼 페이지 - **firewalld** 오프라인 명령줄 클라이언트에 대한 명령 옵션에 대해 설명합니다.
- **firewalld.icmptype(5)** 도움말 페이지 - **ICMP** 필터링을 위한 **XML** 구성 파일을 설명합니다.
- **firewalld.ipset(5)** 도움말 페이지 - **firewalld IP** 세트의 **XML** 구성 파일에 대해 설명합니다.
- **firewalld.service(5)** 도움말 페이지 - **firewalld** 서비스에 대한 **XML** 구성 파일에 대해 설명합니다.
- **firewalld.zone(5)** 도움말 페이지 - **firewalld** 영역 구성을 위한 **XML** 구성 파일에 대해 설명합니다.

- *firewalld.direct(5) 도움말 페이지 - firewalld 직접 인터페이스 구성 파일에 대해 설명합니다.*

- *firewalld.lockdown-whitelist(5) 도움말 페이지 - firewalld 잠금 목록 구성 파일에 대해 설명합니다.*

- *firewalld.rich language(5) 도움말 페이지 - firewalld 풍부한 언어 규칙 구문에 대해 설명합니다.*

- *firewalld.zones(5) 도움말 페이지 - 영역의 일반 설명과 구성 방법에 대한 설명입니다.*

- *firewalld.dbus(5) 도움말 페이지 - firewalld 의 D-Bus 인터페이스에 대해 설명합니다.*

5.18.2. 온라인 문서

- *http://www.firewalld.org/ — firewalld home page.*

6장. NFTABLES 시작하기

nftables 프레임워크는 패킷 분류 기능을 제공하며 **iptables,ip6tables,arptables,ebtables, ipset** 툴에 대해 지정된 후속 요소입니다. 이전의 패킷 필터링 툴에 비해 편의성, 기능 및 성능이 크게 향상되었으며 주요 개선 사항은 다음과 같습니다.

- 선형 처리 대신 내장 조회 테이블
- **IPv4** 및 **IPv6** 프로토콜 모두를 위한 단일 프레임워크
- 전체 규칙 세트를 가져오기, 업데이트 및 저장하는 대신 모두 원자적으로 적용되는 규칙
- 규칙 세트에서 디버깅 및 추적 지원(**nftrace**) 및 **nft** 툴에서 추적 이벤트 모니터링 지원
- 프로토콜별 확장 없이 보다 일관되고 컴팩트한 구문
- 타사 애플리케이션용 **Netlink API**

iptables과 마찬가지로 **nftables**는 체인을 저장하기 위해 테이블을 사용합니다. 체인에는 작업을 수행하기 위한 개별 규칙이 포함되어 있습니다. **nft** 툴은 이전 패킷 필터링 프레임워크의 모든 툴을 대체합니다. **libnftnl** 라이브러리는 **libmnl** 라이브러리를 통해 **nftables Netlink API**와 낮은 수준의 상호 작용에 사용할 수 있습니다.

규칙 세트 변경 효과를 표시하려면 **nft list ruleset** 명령을 사용합니다. 이러한 툴은 테이블, 체인, 규칙, 세트 및 기타 오브젝트를 **nftables** 규칙 세트에 추가하기 때문에 **nft flush ruleset** 명령과 같은 **nftables** 규칙 세트 작업이 이전에 별도의 기존 명령을 사용하여 설치된 규칙 세트에 영향을 줄 수 있습니다.

FIREWALLD 또는 **NFTABLES**를 사용하는 경우

- **firewalld:** 간단한 방화벽 사용 사례에 **firewalld** 유틸리티를 사용하십시오. 이 유틸리티는 사용하기 쉽고 이러한 시나리오의 일반적인 사용 사례를 다룹니다.
- **nftables: nftables** 유틸리티를 사용하여 전체 네트워크와 같이 복잡하고 성능에 중요한 방화벽을 설정합니다.

중요

다른 방화벽 서비스가 서로 영향을 미치지 않도록 하려면 **RHEL** 호스트에서만 실행하고 다른 서비스를 비활성화합니다.

6.1. NFTABLES 스크립트 작성 및 실행

nftables 프레임워크는 방화벽 규칙을 유지 관리하기 위해 쉘 스크립트를 사용하여 얻을 수 있는 기본 스크립팅 환경을 제공합니다. 스크립트 실행은 원자성입니다. 즉, 시스템이 전체 스크립트를 적용하거나 오류가 발생하는 경우 실행을 방지합니다. 이렇게 하면 방화벽이 항상 일관된 상태가 됩니다.

또한 **nftables** 스크립트 환경을 통해 관리자는 다음을 수행할 수 있습니다.

- 주석 추가
- 변수 정의
- 다른 규칙 세트 파일 포함

이 섹션에서는 이러한 기능을 사용하는 방법과 **nftables** 스크립트 생성 및 실행 방법을 설명합니다.

nftables 패키지를 설치하면 **Red Hat Enterprise Linux**가 **/etc/nftables/** 디렉터리에 ***.nft** 스크립트를 자동으로 생성합니다. 이러한 스크립트에는 서로 다른 용도로 테이블 및 빈 체인을 만드는 명령이 포함되어 있습니다.

6.1.1. 지원되는 nftables 스크립트 형식

nftables 스크립팅 환경은 다음 형식의 스크립트를 지원합니다.

- **nft list ruleset** 명령과 동일한 형식으로 스크립트를 작성할 수 있습니다. 규칙 세트를 표시합니다.

```
#!/usr/sbin/nft -f

# Flush the rule set
```

```
flush ruleset

table inet example_table {
  chain example_chain {
    # Chain for incoming packets that drops all packets that
    # are not explicitly allowed by any rule in this chain
    type filter hook input priority 0; policy drop;

    # Accept connections to port 22 (ssh)
    tcp dport ssh accept
  }
}
```

- **nft** 명령에서와 동일한 명령의 구문을 사용할 수 있습니다.

```
#!/usr/sbin/nft -f

# Flush the rule set
flush ruleset

# Create a table
add table inet example_table

# Create a chain for incoming packets that drops all packets
# that are not explicitly allowed by any rule in this chain
add chain inet example_table example_chain { type filter hook input priority 0 ; policy drop ; }

# Add a rule that accepts connections to port 22 (ssh)
add rule inet example_table example_chain tcp dport ssh accept
```

6.1.2. nftables 스크립트 실행

nft 유틸리티로 전달하거나 스크립트를 직접 실행하여 **nftables** 스크립트를 실행할 수 있습니다.

사전 요구 사항

- 이 섹션의 절차에서는 **nftables** 스크립트를 **/etc/nftables/example_firewall.nft** 파일에 저장했다고 가정합니다.

절차 **6.1. nft** 유틸리티를 사용하여 **nftables** 스크립트 실행

- **nft** 유틸리티로 전달하여 **nftables** 스크립트를 실행하려면 다음을 입력합니다.

```
# nft -f /etc/nftables/example_firewall.nft
```

절차 **6.2. nftables** 스크립트를 직접 실행합니다.

1. 필요한 단계는 한 번만 수행됩니다.

    1. 스크립트가 다음 **shebang** 시퀀스로 시작되는지 확인합니다.

    ```
    #!/usr/sbin/nft -f
    ```

    중요

    -f 매개변수를 생략하면 **nft** 유틸리티가 스크립트를 읽고 표시하지 않습니다. **error:** 구문 **error, unexpected newline, expecting string.**

    2. 선택 사항: 스크립트 소유자를 **root** 로 설정합니다.

    ```
    # chown root /etc/nftables/example_firewall.nft
    ```

    3. 소유자를 위한 스크립트를 실행 가능하게 만듭니다.

    ```
    # chmod u+x /etc/nftables/example_firewall.nft
    ```

2. 스크립트를 실행합니다.

```
# /etc/nftables/example_firewall.nft
```

출력이 표시되지 않으면 시스템에서 스크립트를 성공적으로 실행했습니다.

중요

nft 가 스크립트를 성공적으로 실행해도 규칙, 누락된 매개변수 또는 스크립트의 기타 문제로 인해 방화벽이 예상대로 작동하지 않을 수 있습니다.

추가 리소스

- 파일 소유자를 설정하는 방법에 대한 자세한 내용은 **chown(1)** 매뉴얼 페이지를 참조하십시오.

- 파일의 권한을 설정하는 방법에 대한 자세한 내용은 **chmod(1)** 매뉴얼 페이지를 참조하십시오.

- 시스템 부팅을 통한 **nftables** 규칙을 로드하는 방법에 대한 자세한 내용은 다음을 참조하십시오. **6.1.6절. "시스템이 부팅될 때 nftables 규칙 자동 로드"**

6.1.3. nftables 스크립트에서 주석 사용

nftables 스크립팅 환경은 **#** 문자 오른쪽에 있는 모든 항목을 주석으로 해석합니다.

> 예 **6.1. nftables** 스크립트의 주석
>
> 주석은 줄의 시작 부분 및 명령 옆에 있을 수 있습니다.
>
> ```
> ...
> # Flush the rule set
> flush ruleset
>
> add table inet example_table  # Create a table
> ...
> ```

6.1.4. nftables 스크립트에서 변수 사용

nftables 스크립트에서 변수를 정의하려면 **define** 키워드를 사용합니다. 단일 값 및 익명 세트를 변수에 저장할 수 있습니다. 더 복잡한 시나리오에는 이름이 지정된 세트 또는 **verdict** 맵을 사용합니다.

단일 값이 있는 변수

다음 예제에서는 **enp1s0** 값을 사용하여 **INET_DEV** 이라는 변수를 정의합니다.

```
define INET_DEV = enp1s0
```

스크립트에서 **$** 기호 뒤에 변수 이름을 입력하여 변수를 사용할 수 있습니다.

```
...
add rule inet example_table example_chain iifname $INET_DEV tcp dport ssh accept
...
```

익명 세트를 포함하는 변수

다음 예제에서는 익명 집합을 포함하는 변수를 정의합니다.

```
define DNS_SERVERS = { 192.0.2.1, 192.0.2.2 }
```

스크립트에서 **$** 기호 뒤에 변수 이름을 입력하여 변수를 사용할 수 있습니다.

```
add rule inet example_table example_chain ip daddr $DNS_SERVERS accept
```

참고

중괄호는 변수가 집합을 나타냄을 나타내기 때문에 규칙에서 사용할 때 특수한 의미를 갖습니다.

추가 리소스

- 세트에 대한 자세한 내용은 **6.4절. "nftables 명령에서 세트 사용"** 을 참조하십시오.
- **verdict** 맵에 대한 자세한 내용은 **6.5절. "nftables 명령에서 검증 맵 사용"** 을 참조하십시오.

6.1.5. nftables 스크립트에 파일 포함

nftables 스크립팅 환경을 사용하면 관리자가 **include** 문을 사용하여 다른 스크립트를 포함할 수 있습니다.

절대 또는 상대 경로가 없는 파일 이름만 지정하는 경우 **nftables** 에는 기본 검색 경로의 파일이 포함되어 있으며 이는 **Red Hat Enterprise Linux**에서 **/etc** 로 설정됩니다.

예 6.2. 기본 검색 디렉터리에서 파일 포함

기본 검색 디렉터리에서 파일을 포함하려면 다음을 수행합니다.

```
include "example.nft"
```

예 6.3. 디렉터리의 모든 ***.nft** 파일 포함

/etc/nftables/rulesets/ 디렉토리에 저장된 ***.nft*** 로 끝나는 모든 파일을 포함하려면:

```
include "/etc/nftables/rulesets/*.nft"
```

include 문은 점으로 시작하는 파일과 일치하지 않습니다.

추가 리소스

- *자세한 내용은* ***nft(8)*** *매뉴얼 페이지의 포함 파일 섹션을 참조하십시오.*

6.1.6. 시스템이 부팅될 때 ***nftables*** *규칙 자동 로드*

nftables systemd *서비스는* */etc/sysconfig/nftables.conf* *파일에 포함된 방화벽 스크립트를 로드합니다. 이 섹션에서는 시스템이 부팅될 때 방화벽 규칙을 로드하는 방법을 설명합니다.*

사전 요구 사항

- ***nftables*** *스크립트는* */etc/nftables/* *디렉터리에 저장됩니다.*

절차 ***6.3.*** *시스템이 부팅될 때* ***nftables*** *규칙 자동 로드*

1. */etc/sysconfig/nftables.conf* *파일을 편집합니다.*

   - */etc/nftables/* *에서 생성된* ****.nft*** *스크립트를 향상하면* ***nftables*** *패키지를 설치할 때 이러한 스크립트의* ***include*** *명령문의 주석을 제거합니다.*

   - *스크립트를 처음부터 작성하는 경우* ***include*** *문을 추가하여 이러한 스크립트를 포함합니다. 예를 들어* ***nftables*** *서비스가 시작될 때* */etc/nftables/example.nft* *스크립트를 로드하려면 다음을 추가합니다.*

     ```
     include "/etc/nftables/example.nft"
     ```

2. *필요한 경우 시스템을 재부팅하지 않고* ***nftables*** *서비스를 시작하여 방화벽 규칙을 로드합니다.*

```
# systemctl start nftables
```

3. **nftables** 서비스를 활성화합니다.

```
# systemctl enable nftables
```

추가 리소스

- 자세한 내용은 참조하십시오. 6.1.1절. "지원되는 nftables 스크립트 형식"

6.2. NFTABLES 테이블, 체인 및 규칙 생성 및 관리

이 섹션에서는 **nftables** 규칙 세트를 표시하는 방법과 이를 관리하는 방법을 설명합니다.

6.2.1. nftables 규칙 세트 표시

nftables 규칙 세트에는 테이블, 체인 및 규칙이 포함되어 있습니다. 이 섹션에서는 이 규칙 세트를 표시하는 방법에 대해 설명합니다.

모든 규칙 세트를 표시하려면 다음을 입력합니다.

```
# nft list ruleset
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
    tcp dport http accept
    tcp dport ssh accept
  }
}
```

참고

기본적으로 **nftables** 는 테이블을 사전 생성하지 않습니다. 결과적으로 테이블이 없는 호스트에 규칙 세트를 표시하는 경우 **nft list ruleset** 명령은 출력을 표시하지 않습니다.

6.2.2. nftables 테이블 생성

nftables 의 테이블은 체인, 규칙, 세트 및 기타 오브젝트의 컬렉션이 포함된 네임스페이스입니다. 이 섹션에서는 테이블을 만드는 방법에 대해 설명합니다.

각 테이블에는 주소 패밀리가 정의되어 있어야 합니다. 표의 주소 제품군은 테이블 프로세스에서 처리하는 주소 유형을 정의합니다. 테이블을 생성할 때 다음 주소 제품군 중 하나를 설정할 수 있습니다.

- **ip: IPv4** 패킷과만 일치합니다. 주소 제품군을 지정하지 않는 경우 기본값입니다.
- **ip6: IPv6** 패킷만 찾습니다.
- **inet: IPv4** 및 **IPv6** 패킷과 일치합니다.
- **ARP: IPv4 address resolution protocol(ARP)** 패킷과 일치합니다.
- 브릿지: 브리지 장치를 통과하는 패킷을 찾습니다.
- **netdev:** 인그레스의 패킷과 일치합니다.

절차 **6.4. nftables** 테이블 생성

1. 새 테이블을 생성하려면 **nft add table** 명령을 사용합니다. 예를 들어 **IPv4** 및 **IPv6** 패킷을 처리하는 **example_table** 라는 테이블을 생성하려면 다음을 수행합니다.

```
# nft add table inet example_table
```

2. 필요한 경우 규칙 세트의 모든 테이블을 나열합니다.

```
# nft list tables
table inet example_table
```

추가 리소스

- 주소 제품군에 대한 자세한 내용은 **nft(8)** 매뉴얼 페이지의 **Address families** 섹션을 참조하십시오.
-

테이블에서 실행할 수 있는 다른 작업에 대한 자세한 내용은 **nft(8)** 도움말 페이지의 테이블 섹션을 참조하십시오.

6.2.3. nftables 체인 생성

체인은 규칙에 대한 컨테이너입니다. 다음 두 가지 규칙 유형이 있습니다.

- 기본 체인: 기본 체인을 네트워킹 스택의 패킷 진입점으로 사용할 수 있습니다.
- 일반 체인: 정규 체인을 점프 대상으로 사용하고 더 나은 규칙을 구성할 수 있습니다.

이 절차에서는 기존 테이블에 기본 체인을 추가하는 방법을 설명합니다.

사전 요구 사항

- 새 체인을 추가하려는 테이블이 있습니다.

절차 **6.5. nftables** 체인 생성

1. **nft add chain** 명령을 사용하여 새 체인을 생성합니다. 예를 들어 **example_table** 에 **example_chain** 이라는 체인을 만들려면 다음을 수행합니다.

```
# nft add chain inet example_table example_chain '{ type filter hook input priority 0 ; policy accept ; }'
```

중요

쉘이 **endpoints**를 명령의 끝으로 해석하지 않도록 하려면 백슬래시를 사용하여 **Semicels**를 이스케이프해야 합니다. 또한 일부 쉘은 중괄호도 해석하므로 틱(')을 사용하여 중괄호 및 내부의 모든 내용을 따옴표로 묶습니다.

이 체인은 들어오는 패킷을 필터링합니다. **priority** 매개변수는 **nftables** 프로세스가 동일한 후크 값이 있는 순서를 지정합니다. 우선순위가 낮은 값이 더 높은 값보다 우선합니다. **policy** 매개변수는 이 체인의 규칙에 대한 기본 조치를 설정합니다. 서버에 원격으로 로그인한 경우 기본 정책을 삭제 하도록 설정한 경우 다른 규칙이 없는 경우 원격 액세스를 허용하지 않는 경우 즉시 연결이 끊어집니다.

2. 선택적으로 모든 체인을 표시합니다.

```
# nft list chains
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
  }
}
```

추가 리소스

- 주소 제품군에 대한 자세한 내용은 **nft(8)** 매뉴얼 페이지의 **Address families** 섹션을 참조하십시오.

- 체인에서 실행할 수 있는 다른 작업에 대한 자세한 내용은 **nft(8)** 도움말 페이지의 체인 섹션을 참조하십시오.

6.2.4. nftables 체인 끝에 규칙 추가

이 섹션에서는 기존 **nftables** 체인의 끝에 규칙을 추가하는 방법을 설명합니다.

사전 요구 사항

- 규칙을 추가하려는 체인이 있습니다.

절차 **6.6. nftables** 체인 끝에 규칙 추가

1. 새 규칙을 추가하려면 **nft add rule** 명령을 사용합니다. 예를 들어 포트 **22**에서 **TCP** 트래픽을 허용하는 **example_table** 의 **example_chain** 에 규칙을 추가하려면 다음을 수행합니다.

```
# nft add rule inet example_table example_chain tcp dport 22 accept
```

또는 포트 번호 대신 서비스 이름을 지정할 수 있습니다. 이 예제에서는 포트 번호 **22** 대신 **ssh** 를 사용할 수 있습니다. 서비스 이름은 **/etc/services** 파일의 해당 항목을 기반으로 하는 포트 번호로 확인됩니다.

2. 선택적으로 모든 체인과 규칙을 **example_table** 에 표시합니다.

```
# nft list table inet example_table
```

```
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
    ...
    tcp dport ssh accept
  }
}
```

추가 리소스

- 주소 제품군에 대한 자세한 내용은 **nft(8)** 매뉴얼 페이지의 **Address families** 섹션을 참조하십시오.
- 체인에서 실행할 수 있는 다른 작업에 대한 자세한 내용은 **nft(8)** 도움말 페이지의 규칙 섹션을 참조하십시오.

6.2.5. nftables 체인의 시작 부분에 규칙 삽입

이 섹션에서는 기존 **nftables** 체인의 시작 부분에 규칙을 삽입하는 방법을 설명합니다.

사전 요구 사항

- 규칙을 추가하려는 체인이 있습니다.

절차 **6.7. nftables** 체인의 시작 부분에 규칙 삽입

1. 새 규칙을 삽입하려면 **nft insert rule** 명령을 사용합니다. 예를 들어 포트 **22** 에서 **TCP** 트래픽을 허용하는 **example_table** 의 **example_chain** 에 규칙을 삽입하려면 다음을 수행합니다.

```
# nft insert rule inet example_table example_chain tcp dport 22 accept
```

또는 포트 번호 대신 서비스 이름을 지정할 수 있습니다. 이 예제에서는 포트 번호 **22** 대신 **ssh** 를 사용할 수 있습니다. 서비스 이름은 **/etc/services** 파일의 해당 항목을 기반으로 하는 포트 번호로 확인됩니다.

2. 선택적으로 모든 체인과 규칙을 **example_table** 에 표시합니다.

```
# nft list table inet example_table
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
```

```
    tcp dport ssh accept
    ...
  }
}
```

추가 리소스

- 주소 제품군에 대한 자세한 내용은 **nft(8)** 매뉴얼 페이지의 **Address families** 섹션을 참조하십시오.

- 체인에서 실행할 수 있는 다른 작업에 대한 자세한 내용은 **nft(8)** 도움말 페이지의 규칙 섹션을 참조하십시오.

6.2.6. nftables 체인의 특정 위치에 규칙 삽입

이 섹션에서는 **nftables** 체인의 기존 규칙 전후에 규칙을 삽입하는 방법을 설명합니다. 이렇게 하면 올바른 위치에 새 규칙을 배치할 수 있습니다.

사전 요구 사항

- 규칙을 추가하려는 체인이 있습니다.

절차 **6.8. nftables** 체인의 특정 위치에 규칙 삽입

1. **nft -a list ruleset** 명령을 사용하여 **handle**을 포함하여 **example_table** 의 모든 체인과 규칙을 표시합니다.

```
# nft -a list table inet example_table
table inet example_table { # handle 1
  chain example_chain { # handle 1
    type filter hook input priority filter; policy accept;
    tcp dport 22 accept # handle 2
    tcp dport 443 accept # handle 3
    tcp dport 389 accept # handle 4
  }
}
```

a를 사용하면 핸들을 표시합니다. 다음 단계에서 새 규칙을 배치하려면 이 정보가 필요합니다.

2. **example_table** 의 **example_chain** 체인에 새 규칙을 삽입합니다.

- *3 을 처리하기 전에 포트 636 에 TCP 트래픽을 허용하는 규칙을 삽입하려면 다음을 입력합니다.*

```
# nft insert rule inet example_table example_chain position 3 tcp dport 636 accept
```

- *3 을 처리한 후 포트 80 에서 TCP 트래픽을 허용하는 규칙을 추가하려면 다음을 입력합니다.*

```
# nft add rule inet example_table example_chain position 3 tcp dport 80 accept
```

3. *선택적으로 모든 체인과 규칙을 example_table 에 표시합니다.*

```
# nft -a list table inet example_table
table inet example_table { # handle 1
  chain example_chain { # handle 1
    type filter hook input priority filter; policy accept;
    tcp dport 22 accept # handle 2
    tcp dport 636 accept # handle 5
    tcp dport 443 accept # handle 3
    tcp dport 80 accept # handle 6
    tcp dport 389 accept # handle 4
  }
}
```

추가 리소스

- *주소 제품군에 대한 자세한 내용은 nft(8) 매뉴얼 페이지의 Address families 섹션을 참조하십시오.*

- *체인에서 실행할 수 있는 다른 작업에 대한 자세한 내용은 nft(8) 도움말 페이지의 규칙 섹션을 참조하십시오.*

6.3. NFTABLES를 사용하여 NAT 구성

nftables 를 사용하면 다음 네트워크 주소 변환(NAT) 유형을 구성할 수 있습니다.

- *마스커레이딩*

- 소스 NAT(SNAT)
- 대상 NAT (DNAT)
- 리디렉션

6.3.1. 다양한 NAT 유형: 마스커레이딩, 소스 NAT, 대상 NAT 및 리디렉션

NAT(네트워크 주소 변환) 유형은 다음과 같습니다.

마스커레이딩 및 소스 NAT(SNAT)

NAT 유형 중 하나를 사용하여 패킷의 소스 IP 주소를 변경합니다. 예를 들어 인터넷 서비스 공급자는 10.0.0.0/8 과 같은 개인 IP 범위를 라우팅하지 않습니다. 네트워크에서 개인 IP 범위를 사용하고 사용자가 인터넷의 서버에 도달할 수 있어야 하는 경우 이러한 범위의 패킷의 소스 IP 주소를 공용 IP 주소에 매핑합니다.

마스커레이딩과 SNAT 모두 매우 유사합니다. 차이점은 다음과 같습니다.

- 마스커레이딩은 발신 인터페이스의 IP 주소를 자동으로 사용합니다. 따라서 발신 인터페이스에서 동적 IP 주소를 사용하는 경우 마스커레이딩을 사용합니다.
- SNAT 는 패킷의 소스 IP 주소를 지정된 IP로 설정하고 발신 인터페이스의 IP를 동적으로 조회하지 않습니다. 따라서 SNAT 는 마스커레이딩보다 빠릅니다. 발신 인터페이스에서 고정 IP 주소를 사용하는 경우 SNAT 를 사용합니다.

대상 NAT (DNAT)

이 NAT 유형을 사용하여 들어오는 트래픽을 다른 호스트로 라우팅합니다. 예를 들어 웹 서버가 예약된 IP 범위의 IP 주소를 사용하므로 인터넷에서 직접 액세스할 수 없는 경우 라우터에서 DNAT 규칙을 설정하여 들어오는 트래픽을 이 서버로 리디렉션할 수 있습니다.

리디렉션

이 유형은 체인 후크에 따라 패킷을 로컬 시스템으로 리디렉션하는 DNAT의 특별한 경우입니다. 예를 들어 서비스가 표준 포트와 다른 포트에서 실행되는 경우 표준 포트에서 이 특정 포트로 들어오는 트래픽을 리디렉션할 수 있습니다.

6.3.2. nftables를 사용하여 마스커레이딩 구성

마스커레이딩을 사용하면 라우터에서 인터페이스를 통해 전송된 패킷의 소스 IP를 인터페이스의 IP 주소로 동적으로 변경할 수 있습니다. 즉, 인터페이스에 새 IP가 할당되면 nftables 가 소스 IP를 교체할 때 새 IP를 자동으로 사용합니다.

다음 절차에서는 ens3 인터페이스를 통해 호스트를 나가는 패킷의 소스 IP를 ens3 의 IP 세트로 교체하는 방법을 설명합니다.

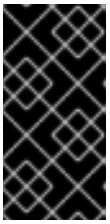
절차 6.9. nftables를 사용하여 마스커레이딩 구성

1. 테이블을 만듭니다.

```
# nft add table nat
```

2. 표에 prerouting 및 postrouting 체인을 추가합니다.

```
# nft -- add chain nat prerouting { type nat hook prerouting priority -100 \; }
# nft add chain nat postrouting { type nat hook postrouting priority 100 \; }
```



중요

사전 제한 체인에 규칙을 추가하지 않더라도 nftables 프레임워크에서 들어오는 패킷 응답과 일치하도록 이 체인이 필요합니다.

쉘이 음수 우선순위 값을 nft 명령의 옵션으로 해석하지 않도록 -- 옵션을 nft 명령에 전달해야 합니다.

3. ens3 인터페이스에서 나가는 패킷과 일치하는 postrouting 체인에 규칙을 추가합니다.

```
# nft add rule nat postrouting oifname "ens3" masquerade
```

6.3.3. nftables를 사용하여 소스 NAT 구성

라우터에서 소스 NAT(SNAT)를 사용하면 인터페이스를 통해 전송된 패킷의 IP를 특정 IP 주소로 변경할 수 있습니다.

다음 절차에서는 **ens3** 인터페이스를 통해 라우터에서 **192.0.2.1** 로 나가는 패킷의 소스 **IP**를 교체하는 방법을 설명합니다.

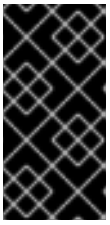
절차 6.10. nftables를 사용하여 소스 NAT 구성

1. 테이블을 만듭니다.

```
# nft add table nat
```

2. 표에 **prerouting** 및 **postrouting** 체인을 추가합니다.

```
# nft -- add chain nat prerouting { type nat hook prerouting priority -100 \; }
# nft add chain nat postrouting { type nat hook postrouting priority 100 \; }
```



중요

postrouting 체인에 규칙을 추가하는 경우에도 **nftables** 프레임워크에서 나가는 패킷 응답과 일치하도록 이 체인이 필요합니다.

쉘이 음수 우선순위 값을 **nft** 명령의 옵션으로 해석하지 않도록 **--** 옵션을 **nft** 명령에 전달해야 합니다.

3. **ens3** 을 통해 나가는 패킷의 소스 **IP**를 **192.0.2.1** 로 대체하는 후기 체인에 규칙을 추가합니다.

```
# nft add rule nat postrouting oifname "ens3" snat to 192.0.2.1
```

추가 리소스

- 자세한 내용은 참조하십시오. **6.6.2절. "특정 로컬 포트의 수신 패킷을 다른 호스트로 전달"**

6.3.4. nftables를 사용하여 대상 NAT 구성

대상 **NAT** 를 사용하면 라우터의 트래픽을 인터넷에서 직접 액세스할 수 없는 호스트로 리디렉션할 수 있습니다.

다음 절차에서는 라우터의 포트 **80** 및 **443** 으로 전송된 수신 트래픽을 **192.0.2.1 IP** 주소를 사용하여 호스트로 리디렉션하는 방법을 설명합니다.

절차 6.11. nftables를 사용하여 대상 NAT 구성

1. 테이블을 만듭니다.

```
# nft add table nat
```

2. 표에 **prerouting** 및 **postrouting** 체인을 추가합니다.

```
# nft -- add chain nat prerouting { type nat hook prerouting priority -100 \; }
# nft add chain nat postrouting { type nat hook postrouting priority 100 \; }
```

중요

postrouting 체인에 규칙을 추가하지 않아도 **nftables** 프레임워크에는 이 체인이 발신 패킷 응답과 일치해야 합니다.

쉘이 음수 우선순위 값을 **nft** 명령의 옵션으로 해석하지 않도록 **--** 옵션을 **nft** 명령에 전달해야 합니다.

3. 포트 **80** 및 **443** 으로 전송된 **ens3** 인터페이스에서 **192.0.2.1 IP**를 사용하여 호스트에 들어오는 트래픽을 리디렉션하는 사전 설정 체인에 규칙을 추가합니다.

```
# nft add rule nat prerouting iifname ens3 tcp dport { 80, 443 } dnat to 192.0.2.1
```

4. 환경에 따라 소스 주소를 변경하려면 **SNAT** 또는 마스커레이딩 규칙을 추가합니다.

    1. **ens3** 인터페이스가 동적 **IP** 주소를 사용하는 경우 마스커레이딩 규칙을 추가합니다.

```
# nft add rule nat postrouting oifname "ens3" masquerade
```

    2. **ens3** 인터페이스에서 고정 **IP** 주소를 사용하는 경우 **SNAT** 규칙을 추가합니다. 예를 들어 **ens3** 에서 **198.51.100.1 IP** 주소를 사용하는 경우 다음을 실행합니다.

```
# nft add rule nat postrouting oifname "ens3" snat to 198.51.100.1
```

추가 리소스

- 자세한 내용은 참조하십시오. 6.3.1절. “다양한 NAT 유형: 마스커레이딩, 소스 NAT, 대상 NAT 및 리디렉션”

6.3.5. nftables를 사용하여 리디렉션 구성

리디렉션 기능은 체인 후크에 따라 패킷을 로컬 시스템으로 리디렉션하는 대상 네트워크 주소 변환 (DNAT)의 특별한 경우입니다.

다음 절차에서는 로컬 호스트의 포트 22 로 전송된 수신 및 전달된 트래픽을 포트 2222 로 리디렉션하는 방법을 설명합니다.

절차 6.12. nftables를 사용하여 리디렉션 구성

1. 테이블을 만듭니다.

```
# nft add table nat
```

2. 표에 사전 체인을 추가합니다.

```
# nft -- add chain nat prerouting { type nat hook prerouting priority -100 \; }
```

쉘이 음수 우선순위 값을 nft 명령의 옵션으로 해석하지 않도록 -- 옵션을 nft 명령에 전달해야 합니다.

3. 포트 22 에서 들어오는 트래픽을 포트 2222 로 리디렉션하는 사전 제한 체인에 규칙을 추가합니다.

```
# nft add rule nat prerouting tcp dport 22 redirect to 2222
```

추가 리소스

- 자세한 내용은 참조하십시오. 6.3.1절. “다양한 NAT 유형: 마스커레이딩, 소스 NAT, 대상 NAT 및 리디렉션”

6.4. NFTABLES 명령에서 세트 사용

nftables 프레임워크는 기본적으로 세트를 지원합니다. 예를 들어 규칙이 여러 **IP** 주소, 포트 번호, 인터페이스 또는 기타 일치 조건과 일치해야 하는 경우 세트를 사용할 수 있습니다.

6.4.1. nftables에서 익명 세트 사용

익명 세트에는 규칙에서 직접 사용하는 **{ 22, 80, 443 }** 와 같이 중괄호로 묶인 쉼표로 구분된 값이 포함됩니다. **IP** 주소 또는 기타 일치 기준에도 익명 세트를 사용할 수 있습니다.

익명 세트의 단점은 세트를 변경하려면 규칙을 대체해야 한다는 것입니다. 동적 솔루션의 경우 **6.4.2 절. "nftables에서 명명된 세트 사용"** 에 설명된 대로 이름이 지정된 세트를 사용합니다.

사전 요구 사항

- **inet** 제품군의 **example_chain** 체인 및 **example_table** 테이블이 있습니다.

절차 **6.13. nftables에서 익명 세트 사용**

1. 예를 들어 포트 **22,80** 및 **443** 으로 들어오는 트래픽을 허용하는 **example_table** 에서 **example_chain** 에 규칙을 추가하려면 다음을 수행합니다.

```
# nft add rule inet example_table example_chain tcp dport { 22, 80, 443 } accept
```

2. 선택적으로 모든 체인과 규칙을 **example_table** 에 표시합니다.

```
# nft list table inet example_table
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
    tcp dport { ssh, http, https } accept
  }
}
```

6.4.2. nftables에서 명명된 세트 사용

nftables 프레임워크는 이름이 지정된 변경 집합을 지원합니다. 명명된 집합은 테이블 내의 여러 규칙에 사용할 수 있는 요소 목록 또는 범위입니다. 세트를 사용하는 규칙을 대체하지 않고 이름이 지정된 세트를 업데이트할 수 있다는 또 다른 이점은 세트를 사용하는 규칙을 대체하지 않고도 이름이 지정된 세트를 업데이트할 수 있다는 것입니다.

명명된 세트를 생성할 때 세트에 포함된 요소 유형을 지정해야 합니다. 다음 유형을 설정할 수 있습니다.

- IPv4 주소 또는 범위를 포함하는 세트의 ipv4_addr (예: 192.0. 2.1) 또는 192.0.2.0/24.
- IPv6 주소 또는 범위가 포함된 세트의 ipv6_addr (예: 2001:db8:1::1 또는 2001:db8:1::1/64) .
- 52:54:00:6b:66:42 와 같은 미디어 액세스 제어(MAC) 주소 목록이 포함된 세트의 ether_addr.
- inet_proto: tcp 와 같은 인터넷 프로토콜 유형 목록이 포함된 세트의 경우.
- inet_service for a set that contains a list of Internet services, such as ssh.
- 패킷 표시 목록을 포함하는 세트의 마크입니다. 패킷 표시는 모든 양의 32비트 정수 값(0 에서 2147483647)일 수 있습니다.

사전 요구 사항

- example_chain 체인 및 example_table 테이블이 있습니다.

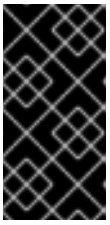
절차 6.14. nftables에서 명명된 세트 사용

1. 빈 세트를 생성합니다. 다음 예제에서는 IPv4 주소에 대한 세트를 생성합니다.
   a. 여러 개의 개별 IPv4 주소를 저장할 수 있는 세트를 생성하려면 다음을 수행하십시오.

```
# nft add set inet example_table example_set { type ipv4_addr \; }
```

   b. IPv4 주소 범위를 저장할 수 있는 세트를 생성하려면 다음을 수행합니다.

```
# nft add set inet example_table example_set { type ipv4_addr \; flags interval \; }
```



중요

셸이 **endpoints**를 명령의 끝으로 해석하지 않도록 하려면 백슬래시를 사용하여 **Semicels**를 이스케이프해야 합니다.

2. 선택적으로 세트를 사용하는 규칙을 만듭니다. 예를 들어 다음 명령은 **example_set** 의 **IPv4** 주소에서 모든 패킷을 삭제하는 **example_table** 의 **example_chain** 에 규칙을 추가합니다.

```
# nft add rule inet example_table example_chain ip saddr @example_set drop
```

example_set 는 여전히 비어 있기 때문에 현재 규칙에는 영향을 미치지 않습니다.

3. **example_set** 에 **IPv4** 주소를 추가합니다.

a. 개별 **IPv4** 주소를 저장하는 세트를 생성하는 경우 다음을 입력합니다.

```
# nft add element inet example_table example_set { 192.0.2.1, 192.0.2.2 }
```

b. **IPv4** 범위를 저장하는 세트를 생성하는 경우 다음을 입력합니다.

```
# nft add element inet example_table example_set { 192.0.2.0-192.0.2.255 }
```

IP 주소 범위를 지정하는 경우 위 예제에서 **192.0.2.0/24** 와 같은**CIDR(Classless Inter-Domain Routing)** 표기법을 사용할 수 있습니다.

6.4.3. 관련 정보

세트에 대한 자세한 내용은 **nft(8)** 매뉴얼 페이지의 **Sets** 섹션을 참조하십시오.

6.5. NFTABLES 명령에서 검증 맵 사용

사전이라고도 하는**verdict** 맵을 사용하면 **nft** 가 작업과 일치 기준을 매핑하여 패킷 정보를 기반으로 작업을 수행할 수 있습니다.

6.5.1. nftables에서 익명 맵 사용

*익명 맵은 규칙에서 직접 사용하는 **{ match_criteria : action }** 문입니다. 문에는 쉼표로 구분된 여러 매핑이 포함될 수 있습니다.*

*익명 맵의 단점은 맵을 변경하려면 규칙을 교체해야 합니다. 동적 솔루션의 경우 **6.5.2절. "nftables에서 이름이 지정된 맵 사용"** 에 설명된 대로 이름이 지정된 맵을 사용합니다.*

*이 예제에서는 익명 맵을 사용하여 **IPv4** 및 **IPv6** 프로토콜의 **TCP** 및 **UDP** 패킷을 서로 다른 체인으로 라우팅하여 수신되는 **TCP** 및 **UDP** 패킷을 개별적으로 계산하는 방법을 설명합니다.*

*절차 **6.15. nftables**에서 익명 맵 사용*

1. ***example_table** 을 생성합니다.*

```
# nft add table inet example_table
```

2. ***example_table** 에 **tcp_packets** 체인을 만듭니다.*

```
# nft add chain inet example_table tcp_packets
```

3. *이 체인의 트래픽 수를 계산하는 **tcp_packets** 에 규칙을 추가합니다.*

```
# nft add rule inet example_table tcp_packets counter
```

4. ***example_table** 에 **udp_packets** 체인을 생성합니다.*

```
# nft add chain inet example_table udp_packets
```

5. *이 체인의 트래픽을 계산하는 **udp_packets** 에 규칙을 추가합니다.*

```
# nft add rule inet example_table udp_packets counter
```

6. *들어오는 트래픽에 사용할 체인을 만듭니다. 예를 들어 들어오는 트래픽을 필터링하는 **example_table** 에서 **incoming_traffic** 라는 체인을 생성하려면 다음을 수행합니다.*

```
# nft add chain inet example_table incoming_traffic { type filter hook input priority 0 \; }
```

7. **incoming_traffic** 에 익명 맵이 있는 규칙을 추가합니다.

```
# nft add rule inet example_table incoming_traffic ip protocol vmap { tcp : jump tcp_packets, udp : jump udp_packets }
```

익명 맵은 패킷을 구분하여 프로토콜을 기반으로 다른 카운터 체인으로 보냅니다.

8. 트래픽 카운터를 나열하려면 **example_table** 을 표시합니다.

```
# nft list table inet example_table
table inet example_table {
  chain tcp_packets {
    counter packets 36379 bytes 2103816
  }

  chain udp_packets {
    counter packets 10 bytes 1559
  }

  chain incoming_traffic {
    type filter hook input priority filter; policy accept;
    ip protocol vmap { tcp : jump tcp_packets, udp : jump udp_packets }
  }
}
```

tcp_packets 및 **udp_packets** 체인의 카운터는 수신된 패킷 수와 바이트를 모두 표시합니다.

6.5.2. nftables에서 이름이 지정된 맵 사용

nftables 프레임워크는 이름이 지정된 **map**을 지원합니다. 이러한 맵은 테이블 내의 여러 규칙에 사용할 수 있습니다. 익명 맵의 또 다른 장점은 이름을 사용하는 규칙을 대체하지 않고 이름이 지정된 맵을 업데이트할 수 있다는 것입니다.

이름이 지정된 맵을 생성할 때 요소 유형을 지정해야 합니다.

- 일치하는 파트에 대한 **ipv4_addr** 에 **IPv4** 주소(예: **192.0.2.1**)가 포함되어 있습니다.
- 일치하는 맵의 **ipv6_addr** 에 **2001:db8:1::1** 과 같은 **IPv6** 주소가 포함되어 있습니다.

- *일치하는 부분이 있는 맵의 **ether_addr** 에는 **52:54:00:6b:66:42** 와 같은 미디어 액세스 제어 **(MAC)** 주소가 포함됩니다.*

- ***inet_proto for a map whose match part contains an Internet protocol type, such as tcp.***

- *일치하는 부분이 있는 맵의 **inet_service** 에는 **ssh** 또는 **22** 와 같은 인터넷 서비스 이름 포트 번호가 포함됩니다.*

- *패킷 마크 가 포함된 맵과 일치합니다. 패킷 마크는 임의의 양의 **32**비트 정수 값(**0**에서 **2147483647)**일 수 있습니다.**A packet mark can be any positive 32-bit integer value (0 to 2147483647).***

- ***part**에 카운터 값이 포함된 맵의 카운터를 나타냅니다.**Represents a counter for a map whose match part contains a counter value.** 카운터 값은 모든 양의 **64**비트 정수 값일 수 있습니다.*

- *바인딩 과 일치하는 맵의 할당량에 할당량 값이 포함됩니다. 할당량 값은 모든 양의 **64**비트 정수 값일 수 있습니다.*

*이 예제에서는 소스 **IP** 주소에 따라 들어오는 패킷을 허용하거나 삭제하는 방법을 설명합니다. 이름 지정된 맵을 사용하면 **IP** 주소 및 작업이 맵에 동적으로 저장되는 동안 이 시나리오를 구성하는 단일 규칙만 필요합니다. 이 절차에서는 맵에서 항목을 추가 및 제거하는 방법도 설명합니다.*

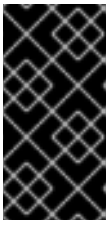
*절차 **6.16. nftables**에서 이름이 지정된 맵 사용*

1. *테이블을 만듭니다. 예를 들어 **IPv4** 패킷을 처리하는 **example_table** 라는 테이블을 생성하려면 다음을 수행합니다.*

```
# nft add table ip example_table
```

2. *체인을 만듭니다. 예를 들어 **example_table** 에 **example_chain** 이라는 체인을 만들려면 다음을 수행합니다.*

```
# nft add chain ip example_table example_chain { type filter hook input priority 0 \; }
```



중요

셸이 **endpoints**를 명령의 끝으로 해석하지 않도록 하려면 백슬래시를 사용하여 **Semicels**를 이스케이프해야 합니다.

3. 빈 맵을 생성합니다. 예를 들어 **IPv4** 주소에 대한 맵을 생성하려면 다음을 수행합니다.

```
# nft add map ip example_table example_map { type ipv4_addr : verdict \; }
```

4. 맵을 사용하는 규칙을 만듭니다. 예를 들어 다음 명령은 **example_map** 에 모두 정의된 **IPv4** 주소에 작업을 적용하는 **example_table** 의 **example_chain** 에 규칙을 추가합니다.

```
# nft add rule example_table example_chain ip saddr vmap @example_map
```

5. **IPv4** 주소 및 해당 작업을 **example_map:**에 추가합니다.

```
# nft add element ip example_table example_map { 192.0.2.1 : accept, 192.0.2.2 : drop }
```

이 예제에서는 작업에 대한 **IPv4** 주소 매핑을 정의합니다. 위에서 만든 규칙과 함께 방화벽은 **192.0.2.1** 에서 패킷을 수락하고 **192.0.2.2** 에서 패킷을 삭제합니다.

6. 선택적으로 다른 **IP** 주소 및 **action** 문을 추가하여 맵을 향상시킵니다.

```
# nft add element ip example_table example_map { 192.0.2.3 : accept }
```

7. 선택적으로 맵에서 항목을 제거합니다.

```
# nft delete element ip example_table example_map { 192.0.2.1 }
```

8. 필요한 경우 규칙 세트를 표시합니다.

```
# nft list ruleset
table ip example_table {
  map example_map {
    type ipv4_addr : verdict
    elements = { 192.0.2.2 : drop, 192.0.2.3 : accept }
  }
```

```
  chain example_chain {
    type filter hook input priority filter; policy accept;
    ip saddr vmap @example_map
  }
}
```

6.5.3. 관련 정보

verdict 맵에 대한 자세한 내용은 **nft(8)** 메뉴얼 페이지의 **Maps** 섹션을 참조하십시오.

6.6. NFTABLES를 사용하여 포트 전달 구성

포트 전달을 사용하면 관리자가 특정 대상 포트로 전송된 패킷을 다른 로컬 또는 원격 포트로 전달할 수 있습니다.

예를 들어 웹 서버에 공용 **IP** 주소가 없는 경우 방화벽에서 포트 전달 규칙을 설정하여 방화벽의 포트 **80** 및 **443** 을 웹 서버로 전달할 수 있습니다. 이 방화벽 규칙을 사용하여 인터넷 사용자는 방화벽의 **IP** 또는 호스트 이름을 사용하여 웹 서버에 액세스할 수 있습니다.

6.6.1. 들어오는 패킷을 다른 로컬 포트로 전달

이 섹션에서는 포트 **8022** 에서 들어오는 **IPv4** 패킷을 로컬 시스템의 포트 **22** 로 전달하는 방법에 대한 예를 설명합니다.

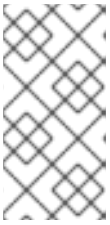
절차 **6.17.** 들어오는 패킷을 다른 로컬 포트로 전달

1. **IP** 주소 제품군을 사용하여 **nat** 라는 테이블을 만듭니다.

```
# nft add table ip nat
```

2. 표에 **prerouting** 및 **postrouting** 체인을 추가합니다.

```
# nft -- add chain ip nat prerouting { type nat hook prerouting priority -100 \; }
```



참고

쉘이 음수 우선 순위 값을 **nft** 명령의 옵션으로 해석하지 않도록 **--** 옵션을 **nft** 명령에 전달합니다.

3. 포트 **8022** 의 들어오는 패킷을 로컬 포트 **22** 로 리디렉션하는 사전 제한 체인에 규칙을 추가합니다.

```
# nft add rule ip nat prerouting tcp dport 8022 redirect to :22
```

6.6.2. 특정 로컬 포트의 수신 패킷을 다른 호스트로 전달

대상 네트워크 주소 변환(**DNAT**) 규칙을 사용하여 로컬 포트의 들어오는 패킷을 원격 호스트로 전달할 수 있습니다. 이를 통해 인터넷 사용자는 개인 **IP** 주소가 있는 호스트에서 실행되는 서비스에 액세스할 수 있습니다.

이 절차에서는 **192.0.2.1 IP** 주소를 사용하여 로컬 포트 **443** 의 수신 **IPv4** 패킷을 원격 시스템의 동일한 포트 번호로 전달하는 방법을 설명합니다.

사전 요구 사항

- 시스템에서 패킷을 전달해야 하는 **root** 사용자로 로그인했습니다.

절차 **6.18.** 특정 로컬 포트의 수신 패킷을 다른 호스트로 전달

1. **IP** 주소 제품군을 사용하여 **nat** 라는 테이블을 만듭니다.

```
# nft add table ip nat
```

2. 표에 **prerouting** 및 **postrouting** 체인을 추가합니다.

```
# nft -- add chain ip nat prerouting { type nat hook prerouting priority -100 \; }
# nft add chain ip nat postrouting { type nat hook postrouting priority 100 \; }
```

참고

쉘이 음수 우선 순위 값을 **nft** 명령의 옵션으로 해석하지 않도록 **--** 옵션을 **nft** 명령에 전달합니다.

3. 포트 **443** 의 들어오는 패킷을 **192.0.2.1** 의 동일한 포트로 리디렉션하는 사전 제한 체인에 규칙을 추가합니다.

```
# nft add rule ip nat prerouting tcp dport 443 dnat to 192.0.2.1
```

4. 발신 트래픽을 마스커레이팅하는 규칙을 **postrouting** 체인에 추가합니다.

```
# nft add rule ip daddr 192.0.2.1 masquerade
```

5. 패킷 전달을 활성화합니다.

```
# echo "net.ipv4.ip_forward=1" > /etc/sysctl.d/95-IPv4-forwarding.conf
# sysctl -p /etc/sysctl.d/95-IPv4-forwarding.conf
```

6.7. NFTABLES를 사용하여 연결 수 제한

nftables 를 사용하여 연결 수를 제한하거나 지정된 양의 연결을 설정하려고 시도하는 **IP** 주소를 차단하여 너무 많은 시스템 리소스를 사용하지 않도록 할 수 있습니다.

6.7.1. nftables를 사용하여 연결 수 제한

nft 유틸리티의 **ct count** 매개 변수를 사용하면 관리자가 연결 수를 제한할 수 있습니다. 절차에서는 들어오는 연결을 제한하는 방법에 대한 기본 예제를 설명합니다.

사전 요구 사항

- **example_table** 의 기본 **example_chain** 이 있습니다.

절차 **6.19. nftables**를 사용하여 연결 수 제한

1. **IPv4** 주소에서 **SSH** 포트**(22)**에 대한 동시 연결을 허용하는 규칙을 추가하고 동일한 **IP**에서 모든 추가 연결을 거부합니다.

```
# nft add rule ip example_table example_chain tcp dport ssh meter
example_meter { ip saddr ct count over 2 } counter reject
```

2. 필요한 경우 이전 단계에서 생성한 미터를 표시합니다.

```
# nft list meter ip example_table example_meter
table ip example_table {
  meter example_meter {
    type ipv4_addr
    size 65535
    elements = { 192.0.2.1 : ct count over 2 , 192.0.2.2 : ct count over 2  }
  }
}
```

elements 항목은 현재 규칙과 일치하는 주소가 표시됩니다. 이 예제에서 요소는 **SSH** 포트에 대한 활성 연결이 있는 **IP** 주소를 나열합니다. 출력은 활성 연결 수 또는 연결이 거부된 경우 표시되지 않습니다.

6.7.2. 1분 이내에 10개 이상의 새로 들어오는 TCP 연결을 시도하는 IP 주소 차단

nftables 프레임워크를 사용하면 관리자가 세트를 동적으로 업데이트할 수 있습니다. 이 섹션에서는 이 기능을 사용하여 **10**분 이내에 **10**개 이상의 **IPv4 TCP** 연결을 설정하는 호스트를 일시적으로 차단하는 방법을 설명합니다. **5**분 후에 **nftables** 가 거부 목록에서 **IP** 주소를 자동으로 제거합니다.

절차 **6.20. 1**분 이내에 **10**개 이상의 새로 들어오는 **TCP** 연결을 시도하는 **IP** 주소 차단

1. **IP** 주소 제품군을 사용하여 필터 테이블을 생성합니다.

```
# nft add table ip filter
```

2. 필터 테이블에 입력 체인을 추가합니다.

```
# nft add chain ip filter input { type filter hook input priority 0 \; }
```

3. 필터 테이블에 **denylist** 집합을 추가합니다.

```
# nft add set ip filter denylist { type ipv4_addr \; flags dynamic, timeout \; timeout 5m \; }
```

이 명령은 **IPv4** 주소에 대한 동적 세트를 생성합니다. **timeout 5m** 매개변수는 **nftables** 가 세트에서 **5**분 후에 항목을 자동으로 제거하도록 정의합니다.

4. 거부 목록 설정에 1분 내에 10개 이상의 새 TCP 연결을 설정하려고 하는 호스트의 소스 IP 주소를 자동으로 추가하는 규칙을 추가합니다.

```
# nft add rule ip filter input ip protocol tcp ct state new, untracked limit rate over 10/minute add @denylist { ip saddr }
```

5. denylist 세트의 IP 주소에서 모든 연결을 삭제하는 규칙을 추가합니다.

```
# nft add rule ip filter input ip saddr @denylist drop
```

6.7.3. 추가 리소스

- 자세한 내용은 참조하십시오. 6.4.2절. “nftables에서 명명된 세트 사용”

6.8. NFTABLES 규칙 디버깅

nftables 프레임워크는 관리자가 규칙을 디버그하고 패킷이 일치하는 경우 다양한 옵션을 제공합니다. 이 섹션에서는 이러한 옵션에 대해 설명합니다.

6.8.1. 카운터를 사용하여 규칙 생성

규칙이 일치되는지 확인하려면 카운터를 사용할 수 있습니다. 이 섹션에서는 카운터로 새 규칙을 만드는 방법을 설명합니다.

기존 규칙에 카운터를 추가하는 절차는 6.8.2절. “기존 규칙에 카운터 추가” 을 참조하십시오.

사전 요구 사항

- 규칙을 추가하려는 체인이 있습니다.

절차 6.21. 카운터를 사용하여 규칙 생성

1. counter 매개 변수가 있는 새 규칙을 체인에 추가합니다. 다음 예제에서는 포트 22 에서 TCP 트래픽을 허용하고 이 규칙과 일치하는 패킷 및 트래픽을 계산하는 카운터가 있는 규칙을 추가합니다.

```
# nft add rule inet example_table example_chain tcp dport 22 counter accept
```

2. 카운터 값을 표시하려면 다음을 수행합니다.

```
# nft list ruleset
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
    tcp dport ssh counter packets 6872 bytes 105448565 accept
  }
}
```

6.8.2. 기존 규칙에 카운터 추가

규칙이 일치되는지 확인하려면 카운터를 사용할 수 있습니다. 이 섹션에서는 기존 규칙에 카운터를 추가하는 방법을 설명합니다.

카운터를 사용하여 새 규칙을 추가하는 절차는 **6.8.1절. "카운터를 사용하여 규칙 생성"**을 참조하십시오.

사전 요구 사항

- 카운터를 추가하려는 규칙이 있습니다.

절차 **6.22.** 기존 규칙에 카운터 추가

1. 핸들을 포함하여 체인에 규칙을 표시합니다.

```
# nft --handle list chain inet example_table example_chain
table inet example_table {
  chain example_chain { # handle 1
    type filter hook input priority filter; policy accept;
    tcp dport ssh accept # handle 4
  }
}
```

2. 규칙을 교체하여 카운터를 **counter** 매개변수로 추가합니다. 다음 예제에서는 이전 단계에서 표시된 규칙을 교체하고 카운터를 추가합니다.

```
# nft replace rule inet example_table example_chain handle 4 tcp dport 22 counter accept
```

3. 카운터 값을 표시하려면 다음을 수행합니다.

```
# nft list ruleset
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
    tcp dport ssh counter packets 6872 bytes 105448565 accept
  }
}
```

6.8.3. 기존 규칙과 일치하는 패킷 모니터링

nftables 의 추적 기능을 **nft monitor** 명령과 함께 사용하면 관리자가 규칙과 일치하는 패킷을 표시할 수 있습니다. 이 절차에서는 규칙에 대한 추적을 활성화하고 이 규칙과 일치하는 패킷을 모니터링하는 방법에 대해 설명합니다.

사전 요구 사항

- 카운터를 추가하려는 규칙이 있습니다.

절차 **6.23.** 기존 규칙과 일치하는 패킷 모니터링

1. 핸들을 포함하여 체인에 규칙을 표시합니다.

```
# nft --handle list chain inet example_table example_chain
table inet example_table {
  chain example_chain { # handle 1
    type filter hook input priority filter; policy accept;
    tcp dport ssh accept # handle 4
  }
}
```

2. 규칙을 교체하지만 메타 **nftrace set 1** 매개변수로 추적 기능을 추가합니다. 다음 예제에서는 이전 단계에서 표시된 규칙을 교체하고 추적을 활성화합니다.

```
# nft replace rule inet example_table example_chain handle 4 tcp dport 22 meta nftrace set 1 accept
```

3. **nft monitor** 명령을 사용하여 추적을 표시합니다. 다음 예제에서는 명령의 출력을 필터링하여 **inet example_table example_chain** 이 포함된 항목만 표시합니다.

```
# nft monitor | grep "inet example_table example_chain"
```

```
trace id 3c5eb15e inet example_table example_chain packet: iif "enp1s0" ether saddr
52:54:00:17:ff:e4 ether daddr 52:54:00:72:2f:6e ip saddr 192.0.2.1 ip daddr 192.0.2.2 ip dscp
cs0 ip ecn not-ect ip ttl 64 ip id 49710 ip protocol tcp ip length 60 tcp sport 56728 tcp dport
ssh tcp flags == syn tcp window 64240
trace id 3c5eb15e inet example_table example_chain rule tcp dport ssh nftrace set 1 accept
(verdict accept)
    ...
```

주의

추적이 활성화된 규칙 수와 일치하는 트래픽의 양에 따라 ***nft monitor*** *명령은 많은 출력을 표시할 수 있습니다.* ***grep*** *또는 기타 유틸리티를 사용하여 출력을 필터링합니다.*

7장. 시스템 감사

Linux 감사 시스템은 시스템에서 보안 관련 정보를 추적하는 방법을 제공합니다. 사전 구성된 규칙에 따라 감사는 시스템에서 발생하는 이벤트에 대한 정보를 가능한 한 많이 기록하는 로그 항목을 생성합니다. 이 정보는 미션 크리티컬한 환경과 보안 정책의 위반 및 수행 조치를 결정하는 데 중요합니다. 감사는 시스템에 추가 보안을 제공하지 않습니다. 오히려 시스템에서 사용되는 보안 정책의 위반을 탐지하는 데 사용할 수 있습니다. 이러한 위반은 SELinux와 같은 추가 보안 조치로 인해 추가로 방지할 수 있습니다.

다음 목록에는 감사가 로그 파일에 기록할 수 있는 몇 가지 정보가 요약되어 있습니다.

- 날짜 및 시간, 유형 및 이벤트 결과.
- 제목 및 오브젝트의 민감도 레이블입니다.
- 이벤트를 트리거한 사용자의 ID와 이벤트 연결입니다.
- 감사 구성에 대한 모든 수정 및 감사 로그 파일에 액세스하려고 시도합니다.
- SSH, Kerberos 등의 인증 메커니즘의 모든 용도.
- 신뢰할 수 있는 데이터베이스(예: /etc/passwd)의 변경 사항.
- 시스템 안으로 또는 시스템에서 정보를 가져오거나 내보내려고 합니다.
- 사용자 ID, 주체 및 오브젝트 레이블 및 기타 속성을 기반으로 이벤트를 포함하거나 제외합니다.

감사 시스템을 사용하는 것도 여러 보안 관련 인증에 대한 요구 사항입니다. 감사는 다음 인증 또는 컴플라이언스 가이드의 요구사항을 충족하거나 초과하도록 설계되었습니다.

- Controled Access Protection Profile (CAPP)

- 레이블이 지정된 보안 보호 프로필(LSPP)
- 규칙 세트 기본 액세스 제어(RSBAC)
- 국제 산업 보안 프로그램 운영 설명서 (NISPOM)
- 연방 정보 보안 관리 법률 (FISMA)
- 결제 카드 산업 - PCI-DSS(Data Security Standard)
- STIG(Security Technical Implementation Guides)

감사 또한 다음과 같습니다.

- NIAP(National Information Assurance Partnership) 및 BSI(Best Security Industrys)에 의해 평가됨.
- Red Hat Enterprise Linux 5에서 LSPP/CAPP/RSBAC/EAL4+ 인증.
- Red Hat Enterprise Linux 6에서 운영 체제 보호 프로파일 / 평가 보장 수준 4 이상 (OSPP/EAL4+) 인증.

사용 사례

파일 액세스 감시

감사에서는 파일 또는 디렉터리에 액세스, 수정, 실행 또는 파일의 속성이 변경되었는지 추적할 수 있습니다. 예를 들어 중요한 파일에 대한 액세스를 감지하고 이러한 파일 중 하나가 손상된 경우 감사 추적을 사용할 수 있는 데 유용합니다.

시스템 호출 모니터링

특정 시스템 호출을 사용할 때마다 로그 항목을 생성하도록 감사를 구성할 수 있습니다. 예를 들어 settimeofday,clock_adjtime 및 기타 시간 관련 시스템 호출을 모니터링하여 시스템 시간 변경 사항

을 추적하는 데 사용할 수 있습니다.

사용자가 실행한 명령 기록

감사는 파일이 실행되었는지 여부를 추적하므로 특정 명령의 모든 실행을 기록하도록 규칙을 정의할 수 있습니다. 예를 들어 /bin 디렉터리의 모든 실행 파일에 대해 규칙을 정의할 수 있습니다. 그런 다음 결과 로그 항목을 사용자 ID로 검색하여 사용자당 실행된 명령의 감사 추적을 생성할 수 있습니다.

시스템 경로의 실행 기록

규칙 호출 시 경로를 inode로 변환하는 파일 액세스를 감시하는 것 외에도 감사에서는 규칙 호출 시 존재하지 않는 경로 실행 또는 규칙 호출 후 파일을 교체한 경우에도 경로 실행을 조사할 수 있습니다. 이를 통해 프로그램 실행 파일을 업그레이드하거나 설치되기 전에 규칙이 계속 작동할 수 있습니다.

보안 이벤트 기록

pam_ceilometer 인증 모듈은 실패한 로그인 시도를 기록할 수 있습니다. 감사는 실패한 로그인 시도를 기록하도록 설정할 수 있으며 로그인을 시도한 사용자에 대한 추가 정보를 제공합니다.

이벤트 검색

감사에서는 로그 항목을 필터링하고 여러 조건에 따라 전체 감사 추적을 제공하는 ausearch 유틸리티를 제공합니다.

요약 보고서 실행

aureport 유틸리티는 기록된 이벤트의 일일 보고서를 생성하는 데 사용할 수 있습니다. 그런 다음 시스템 관리자는 이러한 보고서를 분석하고 의심스런 활동을 추가로 조사할 수 있습니다.

네트워크 액세스 모니터링

시스템 관리자가 네트워크 액세스를 모니터링할 수 있도록 iptables 및 ebtables 유틸리티를 구성하여 감사 이벤트를 트리거할 수 있습니다.

참고

감사에서 수집하는 정보의 양에 따라 시스템 성능이 영향을 받을 수 있습니다.

7.1. 감사 시스템 아키텍처

감사 시스템은 사용자 공간 애플리케이션과 유틸리티와 커널 측 시스템 호출 처리의 두 가지 주요 부분으로 구성됩니다. 커널 구성 요소는 사용자 공간 애플리케이션에서 시스템 호출을 수신하고 사용자,작업,**fstype** 또는 종료 필터 중 하나를 통해 필터링합니다.

시스템 호출이 제외 필터를 통과하면 앞서 언급한 필터 중 하나를 통해 전송됩니다. 감사 규칙 구성에 따라 추가 처리를 위해 감사 데몬으로 전송합니다.

사용자 공간 감사 데몬은 커널에서 정보를 수집하고 로그 파일에 항목을 생성합니다. 기타 감사 사용자 공간 유틸리티는 감사 데몬, 커널 감사 구성 요소 또는 감사 로그 파일과 상호 작용합니다.

- **Audi sp -** 감사 디스패처 데몬은 감사 데몬과 상호 작용하고 추가 처리를 위해 이벤트를 다른 애플리케이션에 보냅니다. 이 데몬의 목적은 실시간 분석 프로그램이 감사 이벤트와 상호 작용할 수 있도록 플러그인 메커니즘을 제공하는 것입니다.

- **auditctl -** 감사 제어 유틸리티는 커널 감사 구성 요소와 상호 작용하여 규칙을 관리하고 이벤트 생성 프로세스의 여러 설정 및 매개 변수를 제어합니다.

- 나머지 감사 유틸리티는 감사 로그 파일의 내용을 입력으로 사용하고 사용자 요구 사항에 따라 출력을 생성합니다. 예를 들어 **aureport** 유틸리티는 기록된 모든 이벤트에 대한 보고서를 생성합니다.

7.2. 감사 패키지 설치

감사 시스템을 사용하려면 시스템에 감사 패키지가 설치되어 있어야 합니다. 감사 패키지(**audit** 및 **audit-libs**)는 기본적으로 **Red Hat Enterprise Linux 7**에 설치됩니다. 이러한 패키지가 설치되어 있지 않은 경우 **root** 사용자로 다음 명령을 실행하여 감사 및 종속 항목을 설치합니다.

```
~]# yum install audit
```

7.3. 감사 서비스 구성

감사 데몬은 **/etc/audit/auditd.conf** 파일에서 구성할 수 있습니다. 이 파일은 감사 데몬의 동작을 수정하는 구성 매개변수로 구성됩니다. 해시 기호(**#**) 뒤에 오는 빈 줄과 텍스트는 무시됩니다. 자세한 내용은 **auditd.conf(5)** 도움말 페이지를 참조하십시오.

7.3.1. 보안 환경을 위한 auditd 구성

*기본 **auditd** 구성은 대부분의 환경에 적합해야 합니다. 그러나 환경에서 엄격한 보안 정책을 충족해야 하는 경우 **/etc/audit/auditd.conf** 파일의 감사 데몬 구성에 대해 다음 설정이 권장됩니다.*

log_file

*감사 로그 파일(일반적으로 **/var/log/audit/**)이 있는 디렉터리는 별도의 마운트 지점에 있어야 합니다. 이렇게 하면 다른 프로세스가 이 디렉터리에서 공간을 소비하지 않으며 감사 데몬의 나머지 공간을 정확하게 감지할 수 있습니다.*

max_log_file

감사 로그 파일이 있는 파티션에서 사용 가능한 공간을 완전히 사용하려면 단일 감사 로그 파일의 최대 크기를 지정해야 합니다.

max_log_file_action

***max_log_file** 에 설정된 제한에 도달하면 감사 로그 파일을 덮어쓰지 않도록 **keep_logs** 로 설정해야 하는 작업을 결정합니다.*

space_left

***space_left_action** 매개 변수에 설정된 작업이 트리거되는 디스크에 남은 여유 공간의 양을 지정합니다. 관리자에게 충분한 시간을 제공하고 디스크 공간을 확보할 수 있는 번호로 설정해야 합니다. **space_left** 값은 감사 로그 파일이 생성되는 비율에 따라 달라집니다.*

space_left_action

***space_left_action** 매개 변수를 이메일 또는 적절한 알림 방법을 사용하여 **exec** 로 설정하는 것이 좋습니다.*

admin_space_left

***admin_space_left_action** 매개변수에 설정된 작업이 트리거되는 절대 최소 공간 크기를 관리자가 수행하는 작업을 로깅할 충분한 공간을 남겨 두는 값으로 설정해야 합니다.*

admin_space_left_action

*단일 사용자 모드로 시스템을 배치하려면 **single -user**로 설정하고 관리자가 일부 디스크 공간을 확보할 수 있도록 해야 합니다.*

disk_full_action

감사 로그 파일을 보유한 파티션에서 사용 가능한 공간이 없는 경우 트리거되는 작업을 중지 또는

단일 로 설정해야 합니다. 이렇게 하면 감사가 더 이상 이벤트를 로깅할 수 없는 경우 시스템이 단일 사용자 모드로 종료되거나 작동합니다.

disk_error_action

감사 로그 파일을 보유하는 파티션에서 오류가 감지되는 경우, 하드웨어 오작동의 처리와 관련된 로컬 보안 정책에 따라 **syslog,**단일 또는 중단 으로 설정해야 하는 경우 트리거되는 작업을 지정합니다.

flush

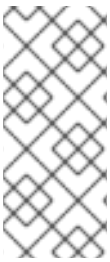
incremental_async 로 설정해야 합니다. 하드 드라이브와 하드 동기화를 강제하기 전에 디스크에 보낼 수 있는 레코드 수를 결정하는 **freq** 매개변수와 함께 작동합니다. **freq** 매개변수는 **100** 으로 설정해야 합니다. 이러한 매개 변수를 사용하면 활동 버스트에 적합한 성능을 유지하면서 감사 이벤트 데이터가 디스크의 로그 파일과 동기화됩니다.

나머지 구성 옵션은 로컬 보안 정책에 따라 설정해야 합니다.

7.4. 감사 서비스 시작

auditd 가 구성되면 서비스를 시작하여 감사 정보를 수집하여 로그 파일에 저장합니다. **auditd** 를 시작하려면 **root** 사용자로 다음 명령을 사용합니다.

```
~]# service auditd start
```



참고

service 명령은 **auditd** 데몬과 올바르게 상호 작용하는 유일한 방법입니다. **auid** 값이 올바르게 기록되도록 **service** 명령을 사용해야 합니다. **systemctl** 명령은 두 가지 작업인 **enable** 및 **status** 에만 사용할 수 있습니다.

부팅 시 시작하도록 **auditd** 를 구성하려면 다음을 수행합니다.

```
~]# systemctl enable auditd
```

service auditd action 명령을 사용하여 **auditd** 에서 다른 여러 작업을 수행할 수 있습니다. 여기서 작업은 다음 중 하나일 수 있습니다.

중지

auditd *를 중지합니다.*

재시작

auditd *를 다시 시작합니다.*

다시 로드 또는 강제 로드

/etc/audit/auditd.conf *파일에서* **auditd** *구성을 다시 로드합니다.*

rotate

/var/log/audit/ *디렉터리에서 로그 파일을 순환합니다.*

resume

감사 로그 파일을 보유하는 디스크 파티션에 사용 가능한 공간이 충분하지 않은 경우와 같이 이전에 일시 중지된 후 감사 이벤트의 로깅을 재개합니다.

condrestart *또는* ***try-restart***

auditd *가 이미 실행 중인 경우에만 다시 시작합니다.*

status

auditd *의 실행 상태를 표시합니다.*

7.5. 감사 규칙 정의

감사 시스템은 로그 파일에서 캡처할 항목을 정의하는 일련의 규칙에서 작동합니다. 다음 유형의 감사 규칙을 지정할 수 있습니다.

제어 규칙

감사 시스템의 동작과 일부 구성을 수정할 수 있도록 허용합니다.

파일 시스템 규칙

파일 감시라고도 하는 경우 특정 파일 또는 디렉터리에 대한 액세스 감사를 허용합니다.

시스템 호출 규칙

지정된 프로그램에서 수행하는 시스템 호출을 로깅할 수 있습니다.

감사 규칙을 설정할 수 있습니다.

- **auditctl** 유틸리티를 사용하는 명령줄에서 다음을 수행합니다. 이러한 규칙은 재부팅 후에도 유지되지 않습니다. 자세한 내용은 참조하십시오. 7.5.1절. "auditctl을 사용하여 감사 규칙 정의"

- **/etc/audit/audit.rules** 파일에서 자세한 내용은 참조하십시오. 7.5.3절. "/etc/audit/audit.rules 파일에서 영구 감사 규칙 및 제어 정의"

7.5.1. auditctl을 사용하여 감사 규칙 정의

auditctl 명령을 사용하면 감사 시스템의 기본 기능을 제어하고 기록된 감사 이벤트를 결정하는 규칙을 정의할 수 있습니다.

참고

감사 서비스 및 감사 로그 파일과 상호 작용하는 모든 명령에는 **root** 권한이 필요합니다. **root** 사용자로 이러한 명령을 실행해야 합니다. 또한 사용자 메시지를 기록하려면 감사 서비스와 **CAP_AUDIT_WRITE**를 설정하려면 **CAP_AUDIT_CONTROL**이 필요합니다.

제어 규칙 정의

다음은 감사 시스템의 동작을 수정할 수 있는 일부 제어 규칙입니다.

-b

커널에서 기존 감사 버퍼의 최대 크기를 설정합니다. 예를 들면 다음과 같습니다.

```
~]# auditctl -b 8192
```

-f

다음과 같이 심각한 오류가 감지될 때 수행되는 작업을 설정합니다.

```
~]# auditctl -f 2
```

위의 구성에서는 심각한 오류가 발생하는 경우 커널 패닉을 트리거합니다.

-e

감사 시스템을 활성화하고 비활성화하거나 해당 구성을 잠급니다. 예를 들면 다음과 같습니다.

```
~]# auditctl -e 2
```

위의 명령은 감사 구성을 잠급니다.

-r

생성된 메시지의 속도를 초당 설정합니다. 예를 들면 다음과 같습니다.

```
~]# auditctl -r 0
```

위의 구성은 생성된 메시지에 대한 속도 제한을 설정하지 않습니다.

-s

감사 시스템의 상태를 보고합니다. 예를 들면 다음과 같습니다.

```
~]# auditctl -s
AUDIT_STATUS: enabled=1 flag=2 pid=0 rate_limit=0 backlog_limit=8192 lost=259 backlog=0
```

-l

현재 로드된 모든 감사 규칙을 나열합니다. 예를 들면 다음과 같습니다.

```
~]# auditctl -l
-w /etc/passwd -p wa -k passwd_changes
-w /etc/selinux -p wa -k selinux_changes
-w /sbin/insmod -p x -k module_insertion
⋮
```

-D

다음과 같이 현재 로드된 모든 감사 규칙을 삭제합니다.

```
~]# auditctl -D
No rules
```

파일 시스템 규칙 정의

파일 시스템 규칙을 정의하려면 다음 구문을 사용합니다.

```
auditctl -w path_to_file -p permissions -k key_name
```

다음과 같습니다.

- **path_to_file** 은 감사되는 파일 또는 디렉터리입니다.
- 권한은 로깅된 권한입니다.
  - **R -** 파일 또는 디렉토리에 대한 읽기 액세스.
  - **w -** 파일 또는 디렉토리에 대한 쓰기 액세스입니다.
  - **X -** 파일 또는 디렉토리에 대한 액세스를 실행합니다.
  - **a -** 파일 또는 디렉터리의 특성을 변경합니다.
- **key_name** 은 특정 로그 항목을 생성한 규칙 또는 규칙 집합을 식별하는 데 도움이 되는 선택적 문자열입니다.

예 7.1. 파일 시스템 규칙

/etc/passwd 파일의 모든 쓰기 액세스 권한을 로깅하는 규칙과 **/etc/passwd** 파일을 변경하는 규칙을 정의하려면 다음 명령을 실행합니다.

```
~]# auditctl -w /etc/passwd -p wa -k passwd_changes
```

-k 옵션 뒤에 있는 문자열은 임의적입니다.

/etc/selinux/ 디렉터리에 있는 모든 파일에 대한 쓰기 액세스 권한을 로깅하는 규칙을 정의하려면 다음 명령을 실행합니다.

```
~]# auditctl -w /etc/selinux/ -p wa -k selinux_changes
```

모듈을 **Linux** 커널에 삽입하는 **/sbin/insmod** 명령의 실행을 기록하는 규칙을 정의하려면 다음 명령을 실행합니다.

```
~]# auditctl -w /sbin/insmod -p x -k module_insertion
```

시스템 호출 규칙 정의

시스템 호출 규칙을 정의하려면 다음 구문을 사용합니다.

```
auditctl -a action,filter -S system_call -F field=value -k key_name
```

다음과 같습니다.

- **action** 및 **filter** 는 특정 이벤트가 기록될 때를 지정합니다. 작업을 항상 또는 사용하지 않을 수 있습니다.**filter** 는 이벤트에 적용되는 커널 규칙 일치 필터를 지정합니다. **rule-matching** 필터는 작업,**exit,user, exclude** 중 하나일 수 있습니다. 이러한 필터에 대한 자세한 내용은 **7.1**절. “감사 시스템 아키텍처” 시작을 참조하십시오.
- **system_call** 은 시스템 호출을 해당 이름으로 지정합니다. 모든 시스템 호출 목록은 **/usr/include/asm/unistd_64.h** 파일에서 확인할 수 있습니다. 여러 시스템 호출을 자체 **-S** 옵션 다음에 지정된 하나의 규칙으로 그룹화할 수 있습니다.
- **field=value** 지정된 아키텍처, 그룹 **ID**, 프로세스 **ID** 등을 기반으로 이벤트를 일치시키도록 규칙을 추가로 수정하는 추가 옵션을 지정합니다. 사용 가능한 모든 필드 유형 및 해당 값의 전체 목록은 **auditctl(8)** 도움말 페이지를 참조하십시오.
- **key_name** 은 특정 로그 항목을 생성한 규칙 또는 규칙 집합을 식별하는 데 도움이 되는 선택적 문자열입니다.

예 7.2. 시스템 호출 규칙

adjtimex 또는 **settimeofofday** 시스템 호출이 프로그램에서 사용될 때마다 로그 항목을 생성하는 규칙을 정의하기 위해 **64**비트 아키텍처를 사용하는 시스템은 다음 명령을 실행합니다.

```
~]# auditctl -a always,exit -F arch=b64 -S adjtimex -S settimeofday -k time_change
```

파일이 삭제될 때마다 로그 항목을 만드는 규칙을 정의하거나 **ID**가 **1000** 이상인 시스템 사용자가 이름을 변경하려면 다음 명령을 실행합니다.

```
~]# auditctl -a always,exit -S unlink -S unlinkat -S rename -S renameat -F auid>=1000 -F
auid!=4294967295 -k delete
```

-F auid!=4294967295 옵션은 로그인 **UID**가 설정되지 않은 사용자를 제외하는 데 사용됩니다.

시스템 호출 규칙 구문을 사용하여 파일 시스템 규칙을 정의할 수도 있습니다. 다음 명령은 **-w /etc/shadow -p wa** 파일 시스템 규칙과 유사한 시스템 호출 규칙을 생성합니다.

```
~]# auditctl -a always,exit -F path=/etc/shadow -F perm=wa
```

7.5.2. 실행 가능한 파일 규칙 정의

실행 가능한 파일 규칙을 정의하려면 다음 구문을 사용합니다.

```
auditctl  -a action,filter [ -F arch=cpu -S system_call] -F exe=path_to_executable_file -k key_name
```

다음과 같습니다.

- **action** 및 **filter** 는 특정 이벤트가 기록될 때를 지정합니다. 작업을 항상 또는 사용하지 않을 수 있습니다.**filter** 는 이벤트에 적용되는 커널 규칙 일치 필터를 지정합니다. **rule-matching** 필터는 작업**,exit,user, exclude** 중 하나일 수 있습니다. 이러한 필터에 대한 자세한 내용은 **7.1**절. “감사 시스템 아키텍처” 시작을 참조하십시오.

- **system_call** 은 시스템 호출을 해당 이름으로 지정합니다. 모든 시스템 호출 목록은 **/usr/include/asm/unistd_64.h** 파일에서 확인할 수 있습니다. 여러 시스템 호출을 자체 **-S** 옵션

다음에 지정된 하나의 규칙으로 그룹화할 수 있습니다.

- **path_to_executable_file** 은 감사된 실행 파일의 절대 경로입니다.
- **key_name** 은 특정 로그 항목을 생성한 규칙 또는 규칙 집합을 식별하는 데 도움이 되는 선택적 문자열입니다.

> 예 7.3. 실행 파일 규칙
>
> **/bin/id** 프로그램의 모든 실행을 로깅하는 규칙을 정의하려면 다음 명령을 실행합니다.
>
> ```
> ~]# auditctl -a always,exit -F exe=/bin/id -F arch=b64 -S execve -k execution_bin_id
> ```

7.5.3. /etc/audit/audit.rules 파일에서 영구 감사 규칙 및 제어 정의

재부팅 후에도 지속되는 감사 규칙을 정의하려면 **/etc/audit/audit.rules** 파일에 직접 포함하거나 **/etc/audit/rules. d/** 디렉터리에 있는 규칙을 읽는 **augenrules** 프로그램을 사용해야 합니다. **/etc/audit/audit.rules** 파일은 동일한 **auditctl** 명령줄 구문을 사용하여 규칙을 지정합니다. 해시 기호(**#**) 뒤에 오는 빈 줄과 텍스트는 무시됩니다.

auditctl 명령을 사용하여 **-R** 옵션을 사용하여 지정된 파일에서 규칙을 읽을 수도 있습니다. 예를 들면 다음과 같습니다.

```
~]# auditctl -R /usr/share/doc/audit/rules/30-stig.rules
```

제어 규칙 정의

파일에는 감사 시스템의 동작을 수정하는 다음 제어 규칙만 포함할 수 있습니다(**-b,-D,-e,-f,-r,--loginuid-immutable, --backlog_wait_time**). 이러한 옵션에 대한 자세한 내용은 "제어 규칙 정의" 을 참조하십시오.

> 예 7.4. audit.rules의 제어 규칙
>
> ```
> # Delete all previous rules
> -D
>
> # Set buffer size
> -b 8192
>
> # Make the configuration immutable -- reboot is required to change audit rules
> -e 2
> ```

```
# Panic when a failure occurs
-f 2

# Generate at most 100 audit messages per second
-r 100

# Make login UID immutable once it is set (may break containers)
--loginuid-immutable 1
```

파일 시스템 및 시스템 호출 규칙 정의

파일 시스템 및 시스템 호출 규칙은 **auditctl** *구문을 사용하여 정의합니다.* ***7.5.1절. "auditctl을 사용하여 감사 규칙 정의"*** *의 예제는 다음 규칙 파일을 사용하여 나타낼 수 있습니다.*

예 7.5. audit.rules의 파일 시스템 및 시스템 호출 규칙

```
-w /etc/passwd -p wa -k passwd_changes
-w /etc/selinux/ -p wa -k selinux_changes
-w /sbin/insmod -p x -k module_insertion

-a always,exit -F arch=b64 -S adjtimex -S settimeofday -k time_change
-a always,exit -S unlink -S unlinkat -S rename -S renameat -F auid>=1000 -F auid!=4294967295 -k delete
```

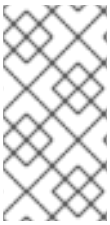
사전 구성된 규칙 파일

/usr/share/doc/audit/rules/ *디렉토리에서* **audit** *패키지는 다양한 인증 표준에 따라 사전 구성된 규칙 파일 세트를 제공합니다.*

- ***30-NISPOM.rules -*** *국가 산업 보안 운영 설명서의 정보 시스템 보안 장에 지정된 요구 사항을 충족하는 감사 규칙 구성입니다.*
- ***30- PCI-dss-v31.rules - PCI DSS(Payment Card Industry Data Security Standard) v3.1*** *로 설정된 요구 사항을 충족하는 감사 규칙 구성입니다.*
- ***30-STIG.rules -STIG(Security Technical Implementation Guides)에*** *의해 설정된 요구사항을 충족하는 감사 규칙 구성.*

이러한 구성 파일을 사용하려면 원본 ***/etc/audit/audit.rules*** *파일의 백업을 생성하고* ***/etc/audit/audit.rules*** *파일을 통해 선택한 구성 파일을 복사합니다.*

```
~]# cp /etc/audit/audit.rules /etc/audit/audit.rules_backup
~]# cp /usr/share/doc/audit/rules/30-stig.rules /etc/audit/audit.rules
```



참고

*감사 규칙에는 번호를 정렬할 수 있는 번호가 매겨져 있습니다. 이름 지정 체계에 대한 자세한 내용은 **/usr/share/doc/audit/rules/README-rules** 파일을 참조하십시오.*

***augenrules** 를 사용하여 영구 규칙 정의*

***augenrules** 스크립트는 **/etc/audit/rules.d/** 디렉터리에 있는 규칙을 읽고 **audit.rules** 파일로 컴파일 합니다. 이 스크립트는 자연적인 정렬 순서에 따라 특정 순서로 **.rules** 로 끝나는 모든 파일을 처리합니다. 이 디렉터리의 파일은 다음과 같은 의미로 그룹으로 구성됩니다.*

- ***10 - kernel 및 auditctl 구성***
- ***20 - 일반 규칙과 일치 할 수 있지만 다른 일치를 원하는 규칙***
- ***30 - 주요 규칙***
- ***40 - 선택적 규칙***
- ***50 - 서버 특정 규칙***
- ***70 - 시스템 로컬 규칙***
- ***90 - 종료(Mmutable)***

*이 규칙은 한 번에 모두 사용할 수 없습니다. 이러한 정책은 간주해야 하는 정책의 일부이며, 개별 파일을 **/etc/audit/rules.d/** 로 복사합니다. 예를 들어 **STIG** 구성에서 시스템을 설정하려면 규칙 **10-base-config, 30-stig, 31-privileged, 99-finalize**를 복사합니다.*

*/etc/audit/rules.d/ 디렉터리에 규칙이 있으면 **--load** 지시문을 사용하여 **augenrules** 스크립트를 실행하여 로드합니다.*

```
~]# augenrules --load
augenrules --load No rules
enabled 1
failure 1
pid 634
rate_limit 0
backlog_limit 8192
lost 0
backlog 0
enabled 1
failure 1
pid 634
rate_limit 0
backlog_limit 8192
lost 0
backlog 1
```

*감사 규칙 및 **augenrules** 스크립트에 대한 자세한 내용은 **audit.rules(8)** 및 **augenrules(8)** 매뉴얼 페이지를 참조하십시오.*

7.6. 감사 로그 파일 이해

*기본적으로 감사 시스템은 **/var/log/audit/audit.log** 파일에 로그 항목을 저장합니다. 로그 순환이 활성화된 경우 순환된 **audit.log** 파일이 동일한 디렉터리에 저장됩니다.*

*다음 감사 규칙은 **/etc/ssh/sshd_config** 파일을 읽거나 수정하려고 할 때마다 기록됩니다.*

```
-w /etc/ssh/sshd_config -p warx -k sshd_config
```

*예를 들어 다음 명령을 사용하여 **auditd** 데몬이 실행 중인 경우 감사 로그 파일에 새 이벤트가 생성됩니다.*

```
~]$ cat /etc/ssh/sshd_config
```

***audit.log** 파일의 이 이벤트는 다음과 같습니다.*

```
type=SYSCALL msg=audit(1364481363.243:24287): arch=c000003e syscall=2 success=no exit=-13
a0=7fffd19c5592 a1=0 a2=7fffd19c4b50 a3=a items=1 ppid=2686 pid=3538 auid=1000 uid=1000
gid=1000 euid=1000 suid=1000 fsuid=1000 egid=1000 sgid=1000 fsgid=1000 tty=pts0 ses=1
```

```
comm="cat" exe="/bin/cat" subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
key="sshd_config"
type=CWD msg=audit(1364481363.243:24287):  cwd="/home/shadowman"
type=PATH msg=audit(1364481363.243:24287): item=0 name="/etc/ssh/sshd_config" inode=409248
dev=fd:00 mode=0100600 ouid=0 ogid=0 rdev=00:00 obj=system_u:object_r:etc_t:s0
objtype=NORMAL cap_fp=none cap_fi=none cap_fe=0 cap_fver=0
type=PROCTITLE msg=audit(1364481363.243:24287) :
proctitle=636174002F6574632F7373682F737368645F636F6E666967
```

위의 이벤트는 동일한 타임스탬프와 일련 번호를 공유하는 네 개의 레코드로 구성됩니다. 레코드는 항상 **type=** 키워드로 시작합니다. 각 레코드는 공백이나 쉼표로 구분된 여러 **name=**값 쌍으로 구성됩니다. 위 이벤트의 자세한 분석은 다음과 같습니다.

첫 번째 레코드

type=SYSCALL

type 필드에는 레코드의 유형이 포함됩니다. 이 예에서 **SYSCALL** 값은 이 레코드가 커널에 대한 시스템 호출에 의해 트리거되었음을 지정합니다.

가능한 모든 유형 값 및 해당 설명 목록은 감사 레코드 유형 항목을 참조하십시오.

msg=audit(1364481363.243:24287):

msg 필드는 다음과 같이 기록합니다.

- 감사 양식에서 타임스탬프 및 레코드의 고유 **ID(time_stamp:ID)**. 동일한 감사 이벤트의 일부로 생성된 경우 여러 레코드가 동일한 타임스탬프 및 **ID**를 공유할 수 있습니다. 타임 스탬프는 **1970**년 **1**월 **1**일 **00:00:00**부터 **UTC** 이후의 **Unix** 시간 형식을 사용합니다.
- 다양한 이벤트별 이름**=** 커널 또는 사용자 공간 애플리케이션에서 제공하는값 쌍입니다.

arch=c000003e

arch 필드에는 시스템의 **CPU** 아키텍처에 대한 정보가 포함되어 있습니다. 값 **c000003e** 은 **16**진수 표기법으로 인코딩됩니다. **ausearch** 명령으로 감사 레코드를 검색할 때 **-i** 또는 **--interpret** 옵션을 사용하여 **16**진수 값을 사람이 읽을 수 있는 동등한 값으로 자동 변환합니다. **c000003e** 값은 **x86_64** 로 해석됩니다.

syscall=2

syscall 필드는 커널에 전송된 시스템 호출 유형을 기록합니다. 값 **2** 는 **/usr/include/asm/unistd_64.h** 파일에서 사람이 읽을 수 있는 것과 일치시킬 수 있습니다. 이 경우 **2**

는 공개 시스템 호출입니다. **ausyscall** 유틸리티를 사용하면 시스템 호출 번호를 사람이 읽을 수 있는 해당 번호로 변환할 수 있습니다. **ausyscall --dump** 명령을 사용하여 숫자와 함께 모든 시스템 호출 목록을 표시합니다. 자세한 내용은 **ausyscall(8)** 도움말 페이지를 참조하십시오.

success=no

success 필드는 특정 이벤트에 기록된 시스템 호출이 성공 또는 실패인지를 기록합니다. 이 경우 전화가 성공하지 못했습니다.

exit=-13

exit 필드에는 시스템 호출에서 반환된 종료 코드를 지정하는 값이 포함되어 있습니다. 이 값은 시스템 호출마다 다릅니다. 다음 명령을 사용하여 사람이 읽을 수 있는 해당 값을 해석할 수 있습니다.

```
~]# ausearch --interpret --exit -13
```

이전 예에서는 감사 로그에 종료 코드 **-13** 로 실패한 이벤트가 포함되어 있다고 가정합니다.

a0=7fffd19c5592, a1=0, a2=7fffd19c5592, a3=a

a0 필드의 **a 0** 필드는 이 이벤트에서 시스템 호출의 **16**진수 표기법으로 인코딩된 처음 네 개의 인수를 기록합니다. 이러한 인수는 사용되는 시스템 호출에 따라 다릅니다. **ausearch** 유틸리티에서 해석할 수 있습니다.

items=1

items 필드에는 **syscall** 레코드를 따르는 **PATH** 보조 레코드 수가 포함되어 있습니다.

ppid=2686

ppid 필드는 **PPID(Parent Process ID)**를 기록합니다. 이 경우 **2686** 은 상위 프로세스의 **PPID(예: bash)**였습니다.

pid=3538

pid 필드는 **PID(프로세스 ID)**를 기록합니다. 이 경우 **3538** 은 **cat** 프로세스의 **PID**입니다.

auid=1000

auid 필드는 **loginuid**인 감사 사용자 **ID**를 기록합니다. 이 **ID**는 로그인 시 사용자에게 할당되며, 예를 들어 사용자 계정을 **su - john** 명령으로 전환하여 모든 프로세스에서 상속됩니다.

uid=1000

uid 필드는 분석 프로세스를 시작한 사용자의 사용자 ID를 기록합니다. 사용자 ID는 다음 명령을 사용하여 사용자 이름으로 해석될 수 있습니다. ausearch -i --uid UID.

gid=1000

gid 필드는 분석 프로세스를 시작한 사용자의 그룹 ID를 기록합니다.

euid=1000

euid 필드는 분석 프로세스를 시작한 사용자의 유효한 사용자 ID를 기록합니다.

suid=1000

suid 필드는 분석 프로세스를 시작한 사용자의 설정된 사용자 ID를 기록합니다.

fsuid=1000

fsuid 필드는 분석 프로세스를 시작한 사용자의 파일 시스템 사용자 ID를 기록합니다.

egid=1000

egid 필드는 분석 프로세스를 시작한 사용자의 유효한 그룹 ID를 기록합니다.

sgid=1000

sgid 필드는 분석 프로세스를 시작한 사용자의 세트 그룹 ID를 기록합니다.

fsgid=1000

fsgid 필드는 분석 프로세스를 시작한 사용자의 파일 시스템 그룹 ID를 기록합니다.

tty=pts0

tty 필드는 분석 프로세스가 호출된 터미널을 기록합니다.

ses=1

ses 필드는 분석 프로세스가 호출된 세션의 세션 ID를 기록합니다.

comm="cat"

comm 필드는 분석 프로세스를 호출하는 데 사용된 명령의 명령줄 이름을 기록합니다. 이 경우 **cat** 명령을 사용하여 이 감사 이벤트를 트리거했습니다.

exe="/bin/cat"

exe 필드는 분석 프로세스를 호출하는 데 사용된 실행 파일의 경로를 기록합니다.

subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023

subj 필드는 분석 프로세스가 실행 시 레이블이 지정된 **SELinux** 컨텍스트를 기록합니다.

key="sshd_config"

key 필드는 감사 로그에서 이 이벤트를 생성한 규칙과 연관된 관리자 정의 문자열을 기록합니다.

두 번째 레코드

type=CWD

두 번째 레코드에서 **type** 필드 값은 **CWD** - 현재 작업 디렉터리입니다. 이 유형은 첫 번째 레코드에 지정된 시스템 호출을 호출한 프로세스가 실행된 작업 디렉터리를 기록하는 데 사용됩니다.

이 레코드의 목적은 상대 경로가 연결된 **PATH** 레코드에 캡처되는 경우 현재 프로세스의 위치를 기록하는 것입니다. 이렇게 하면 절대 경로를 재구성할 수 있습니다.

msg=audit(1364481363.243:24287)

msg 필드는 첫 번째 레코드의 값과 동일한 타임스탬프 및 **ID** 값을 보유합니다. 타임 스탬프는 **1970**년 **1**월 **1**일 **00:00:00**부터 **UTC** 이후의 **Unix** 시간 형식을 사용합니다.

cwd="/home/user_name"

cwd 필드에는 시스템 호출이 호출된 디렉터리의 경로가 포함됩니다.

세 번째 레코드

type=PATH

세 번째 레코드에서 **type** 필드 값은 **PATH** 입니다. 감사 이벤트에는 시스템 호출에 인수로 전달되

는 모든 경로에 대한 **PATH-type** 레코드가 포함됩니다. 이 감사 이벤트에서는 하나의 경로 (**/etc/ssh/sshd_config**)만 인수로 사용되었습니다.

msg=audit(1364481363.243:24287):

msg 필드는 첫 번째 및 두 번째 레코드의 값과 동일한 타임스탬프 및 **ID** 값을 보유합니다.

item=0

item 필드는 **SYSCALL** 유형 레코드에서 참조되는 총 항목 수, 현재 레코드를 나타냅니다. 이 숫자는 **0**을 기반으로 하며, 값이 **0** 이면 첫 번째 항목임을 의미합니다.

name="/etc/ssh/sshd_config"

name 필드는 시스템 호출에 전달된 파일 또는 디렉터리의 경로를 인수로 기록합니다. 이 경우 **/etc/ssh/sshd_config** 파일이었습니다.

inode=409248

inode 필드에는 이 이벤트에 기록된 파일 또는 디렉터리와 연결된 **inode** 번호가 포함됩니다. 다음 명령은 **409248 inode** 번호와 연결된 파일 또는 디렉터리를 표시합니다.

```
~]# find / -inum 409248 -print
/etc/ssh/sshd_config
```

dev=fd:00

dev 필드는 이 이벤트에 기록된 파일 또는 디렉터리가 포함된 장치의 마이너 및 주요 **ID**를 지정합니다. 이 경우 값은 **/dev/fd/0** 장치를 나타냅니다.

mode=0100600

mode 필드는 **st_mode** 필드의 **stat** 명령에서 반환된 대로 숫자 표기법으로 인코딩된 파일 또는 디렉터리 권한을 기록합니다. 자세한 내용은 **stat(2)** 매뉴얼 페이지를 참조하십시오. 이 경우 **0100600**은 **-rw-------** 로 해석될 수 있습니다. 즉, **root** 사용자만 **/etc/ssh/sshd_config** 파일에 대한 읽기 및 쓰기 권한을 갖습니다.

ouid=0

O uid 필드는 오브젝트 소유자의 사용자 **ID**를 기록합니다.

ogid=0

ogid 필드는 오브젝트 소유자의 그룹 ID를 기록합니다.

rdev=00:00

rdev 필드에는 특수 파일에 대한 기록된 장치 식별자만 포함됩니다. 이 경우 기록된 파일이 일반 파일이므로 사용되지 않습니다.

obj=system_u:object_r:etc_t:s0

obj 필드는 실행 시 기록된 파일 또는 디렉터리의 레이블이 지정된 SELinux 컨텍스트를 기록합니다.

objtype=NORMAL

objtype 필드는 지정된 syscall의 컨텍스트에서 각 경로 레코드의 작업의 의도를 기록합니다.

cap_fp=none

cap_fp 필드는 파일 또는 디렉터리 오브젝트의 허용된 파일 시스템 기반 기능의 설정과 관련된 데이터를 기록합니다.

cap_fi=none

cap_fi 필드는 파일 또는 디렉터리 오브젝트의 상속된 파일 시스템 기반 기능의 설정과 관련된 데이터를 기록합니다.

cap_fe=0

cap_fe 필드는 파일 또는 디렉터리 오브젝트의 유효 시스템 기반 기능의 설정을 기록합니다.

cap_fver=0

cap_fver 필드는 파일 또는 디렉터리 오브젝트의 파일 시스템 기반 기능의 버전을 기록합니다.

네 번째 레코드

type=PROCTITLE

type 필드에는 레코드의 유형이 포함됩니다. 이 예제에서 PROCTITLE 값은 이 레코드가 커널에 대한 시스템 호출에 의해 트리거되는 이 감사 이벤트를 트리거한 전체 명령줄을 제공하도록 지정합니

다.

proctitle=636174002F6574632F7373682F737368645F636F6E666967

proctitle 필드는 분석 프로세스를 호출하는 데 사용된 명령의 전체 명령줄을 기록합니다. 이 필드는 사용자가 감사 로그 구문 분석기에 영향을 미치지 않도록 **16**진수 표기법으로 인코딩됩니다. 텍스트는 이 감사 이벤트를 트리거한 명령에 대해 디코딩합니다. **ausearch** 명령으로 감사 레코드를 검색할 때 **-i** 또는 **--interpret** 옵션을 사용하여 **16**진수 값을 사람이 읽을 수 있는 동등한 값으로 자동 변환합니다. **636174002F6574632F7373682F736F636F6E666967** 값은 **cat /etc/ssh/sshd_config** 로 해석됩니다.

위에서 분석한 감사 이벤트는 이벤트에 포함될 수 있는 가능한 모든 필드의 하위 집합만 포함합니다. 모든 이벤트 필드 및 해당 설명 목록은 *감사 이벤트 필드를 참조하십시오*. 모든 이벤트 유형 및 해당 설명 목록은 *감사 레코드 유형을 참조하십시오*.

예 7.6. 추가 audit.log 이벤트

다음 감사 이벤트는 **auditd** 데몬의 시작을 기록합니다. **ver** 필드에는 시작된 감사 데몬의 버전이 표시됩니다.

```
type=DAEMON_START msg=audit(1363713609.192:5426): auditd start, ver=2.2 format=raw
kernel=2.6.32-358.2.1.el6.x86_64 auid=1000 pid=4979 subj=unconfined_u:system_r:auditd_t:s0
res=success
```

다음 감사 이벤트는 **UID**가 **1000**인 사용자의 실패한 시도를 기록하여 **root** 사용자로 로그인합니다.

```
type=USER_AUTH msg=audit(1364475353.159:24270): user pid=3280 uid=1000 auid=1000
ses=1 subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='op=PAM:authentication acct="root" exe="/bin/su" hostname=? addr=? terminal=pts/0
res=failed'
```

7.7. 감사 로그 파일 검색

ausearch 유틸리티를 사용하면 특정 이벤트에 대한 감사 로그 파일을 검색할 수 있습니다. 기본적으로 **ausearch** 는 **/var/log/audit/audit.log** 파일을 검색합니다. **ausearch options -if file_name** 명령을 사용하여 다른 파일을 지정할 수 있습니다. 하나의 **ausearch** 명령에 여러 옵션을 제공하는 것은 동일한 필드 유형의 여러 인스턴스 간에 **AND** 연산자를 사용하는 것과 동일합니다.

예 7.7. ausearch 를 사용하여 로그 파일 감사 검색

/var/log/audit/audit.log 파일에서 실패한 로그인 시도를 검색하려면 다음 명령을 사용하십시오.

```
~]# ausearch --message USER_LOGIN --success no --interpret
```

모든 계정, 그룹 및 역할 변경 사항을 검색하려면 다음 명령을 사용합니다.

```
~]# ausearch -m ADD_USER -m DEL_USER -m ADD_GROUP -m USER_CHAUTHTOK -m
DEL_GROUP -m CHGRP_ID -m ROLE_ASSIGN -m ROLE_REMOVE -i
```

사용자의 로그인 ID(auid)를 사용하여 특정 사용자가 수행한 모든 작업을 검색하려면 다음 명령을 사용합니다.

```
~]# ausearch -ua 1000 -i
```

토요일부터 지금까지 실패한 모든 시스템 호출을 검색하려면 다음 명령을 사용합니다.

```
~]# ausearch --start yesterday --end now -m SYSCALL -sv no -i
```

모든 ausearch 옵션의 전체 목록은 ausearch(8) 도움말 페이지를 참조하십시오.

7.8. 감사 보고서 생성

aureport 유틸리티를 사용하면 감사 로그 파일에 기록된 이벤트에 대한 요약 및 columnar 보고서를 생성할 수 있습니다. 기본적으로 /var/log/audit/ 디렉터리의 모든 audit.log 파일을 쿼리하여 보고서를 작성합니다. aureport options -if file_name 명령을 사용하여 보고서를 실행하도록 다른 파일을 지정할 수 있습니다.

예 7.8. aureport 를 사용하여 감사 보고서 생성

현재 예제를 제외하고 지난 3일 동안 기록된 이벤트에 대한 보고서를 생성하려면 다음 명령을 사용합니다.

```
~]# aureport --start 04/08/2013 00:00:00 --end 04/11/2013 00:00:00
```

실행 가능한 모든 파일 이벤트의 보고서를 생성하려면 다음 명령을 사용합니다.

```
~]# aureport -x
```

위의 실행 파일 이벤트 보고서에 대한 요약을 생성하려면 다음 명령을 사용합니다.

```
~]# aureport -x --summary
```

모든 사용자에 대해 실패한 이벤트에 대한 요약 보고서를 생성하려면 다음 명령을 사용합니다.

```
~]# aureport -u --failed --summary -i
```

각 시스템 사용자별로 실패한 모든 로그인 시도에 대한 요약 보고서를 생성하려면 다음 명령을 사용합니다.

```
~]# aureport --login --summary -i
```

사용자 **ID 1000** 의 모든 파일 액세스 이벤트를 검색하는 **ausearch** 쿼리에서 보고서를 생성하려면 다음 명령을 사용합니다.

```
~]# ausearch --start today --loginuid 1000 --raw | aureport -f --summary
```

쿼리되는 모든 감사 파일의 보고서 및 포함된 이벤트의 시간 범위를 생성하려면 다음 명령을 사용합니다.

```
~]# aureport -t
```

모든 **aureport** 옵션의 전체 목록은 **aureport(8)** 도움말 페이지를 참조하십시오.

7.9. 추가 리소스

감사 시스템에 대한 자세한 내용은 다음 소스를 참조하십시오.

온라인 소스

- *RHEL 감사 시스템 참조: https://access.redhat.com/articles/4409591.*
- *Linux 감사 설명서 프로젝트 페이지: https://github.com/linux-audit/audit-documentation/wiki.*

설치된 문서

audit 패키지에서 제공하는 문서는 /usr/share/doc/audit/ 디렉터리에서 확인할 수 있습니다.

수동 페이지

- *audispd.conf(5)*
- *auditd.conf(5)*
- *ausearch-expression(5)*
- *audit.rules(7)*
- *audispd(8)*
- *auditctl(8)*
- *auditd(8)*
- *aulast(8)*
- *aulastlog(8)*

- ***aureport(8)***
- ***ausearch(8)***
- ***ausyscall(8)***
- ***autrace(8)***
- ***auvirt(8)***

8장. 구성 규정 준수 및 취약점에 대해 시스템 스캔

규정 준수 감사는 지정된 오브젝트가 규정 준수 정책에 지정된 모든 규칙을 따르는지 여부를 결정하는 프로세스입니다. 규정 준수 정책은 컴퓨팅 환경에서 사용해야 하는 체크리스트 형태로 필요한 설정을 지정하는 보안 전문가가 정의합니다.

규정 준수 정책은 조직마다, 심지어 동일한 조직 내의 여러 시스템에서도 크게 달라질 수 있습니다. 이러한 정책의 차이는 각 시스템의 목적과 조직의 중요성을 기반으로 합니다. 사용자 지정 소프트웨어 설정 및 배포 특성으로 인해 사용자 지정 정책 체크리스트가 필요합니다.

8.1. RHEL의 구성 규정 준수 툴

Red Hat Enterprise Linux는 완전 자동화된 컴플라이언스 감사를 수행할 수 있는 도구를 제공합니다. 이러한 툴은 **SCAP(Security Content Automation Protocol)** 표준을 기반으로 하며 규정 준수 정책의 자동화된 조정을 위해 설계되었습니다.

- **SCAP Workbench - scap-workbench** 그래픽 유틸리티는 단일 로컬 또는 원격 시스템에서 구성 및 취약점 검사를 수행하도록 설계되었습니다. 또한 이러한 스캔 및 평가를 기반으로 보안 보고서를 생성하는 데 사용할 수도 있습니다.

- **OpenSCAP - oscap** 명령줄 유틸리티를 포함하는 **OpenSCAP** 라이브러리는 로컬 시스템에서 구성 및 취약점 스캔을 수행하고 구성 규정 준수 콘텐츠의 유효성을 검사하고 이러한 스캔 및 평가를 기반으로 보고서 및 가이드를 생성하도록 설계되었습니다.

- **SCAP Security Guide(SSG) - scap-security-guide** 패키지는 **Linux** 시스템에 대한 최신 보안 정책 컬렉션을 제공합니다. 이 지침은 적용 가능한 경우 정부 요구 사항과 연결된 실용적인 강화 조언 카탈로그로 구성됩니다. 이 프로젝트는 일반화된 정책 요구 사항과 특정 구현 지침 간의 격차를 해소합니다.

- **SCE (Script Check Engine; 스크립트 검사 엔진) - SCE**는 관리자가 **Bash, Python, Ruby**와 같은 스크립팅 언어를 사용하여 보안 콘텐츠를 작성할 수 있는 **SCAP** 프로토콜의 확장입니다. **SCE** 확장은 **openscap-engine-sce** 패키지에 제공됩니다. **SCE** 자체는 **SCAP** 환경의 일부가 아닙니다.

여러 시스템에서 원격으로 자동화된 규정 준수 감사를 수행하려면 **Red Hat Satellite**에 **OpenSCAP** 솔루션을 사용할 수 있습니다.

추가 리소스

- *oscap(8) - oscap 명령줄 유틸리티의 도움말 페이지에서 사용 가능한 옵션과 사용법에 대한 전체 목록을 제공합니다.*

- *Red Hat 보안 데모: 보안 규정 준수를 자동화할 수 있는 사용자 지정 보안 정책 콘텐츠 생성 - 업계 표준 보안 정책 및 사용자 지정 보안 정책을 모두 준수하기 위해 Red Hat Enterprise Linux에 포함된 툴을 사용하여 보안 규정 준수 자동화에 초기 환경을 제공하는 실습 랩입니다. 팀을 위해 이러한 랩 연습을 교육하거나 액세스하려면 자세한 내용은 Red Hat 계정 팀에 문의하십시오.*

- *Red Hat 보안 데모: RHEL Security Technologies - 실습 랩을 통해 OpenSCAP을 포함한 Red Hat Enterprise Linux에서 사용할 수 있는 주요 보안 기술을 사용하여 RHEL 시스템의 모든 수준에서 보안을 구현하는 방법을 배울 수 있습니다. 팀을 위해 이러한 랩 연습을 교육하거나 액세스하려면 자세한 내용은 Red Hat 계정 팀에 문의하십시오.*

- *SCAP-workbench(8) - SCAP Workbench 애플리케이션의 매뉴얼 페이지는 애플리케이션에 대한 기본 정보와 SCAP 콘텐츠의 잠재적인 소스에 대한 링크를 제공합니다.*

- *scap-security-guide(8) - scap-security-guide 프로젝트의 도움말 페이지에서 사용 가능한 다양한 SCAP 보안 프로필에 대한 추가 문서를 제공합니다. 또한 OpenSCAP 유틸리티를 사용하여 제공된 벤치마크를 사용하는 예제도 포함되어 있습니다.*

- *Red Hat Satellite 관리 가이드의 보안 규정 준수 관리에서는 Red Hat Satellite에서 OpenSCAP 사용에 대한 자세한 내용을 제공합니다.*

8.2. 취약점 검사

8.2.1. Red Hat Security Advisories OVAL Feed

Red Hat Enterprise Linux 보안 감사 기능은 SCAP(Security Content Automation Protocol) 표준을 기반으로 합니다. SCAP는 자동화된 구성, 취약점 및 패치 검사, 기술 제어 준수 활동 및 보안 측정을 지원하는 다용도 사양 프레임워크입니다.

SCAP 사양은 스캐너 또는 정책 편집기를 구현하지 않아도 보안 콘텐츠 형식이 잘 알려져 표준화되어 있는 에코시스템을 생성합니다. 이를 통해 조직은 채택한 보안 벤더 수에 관계없이 SCC(보안 정책)를 한 번 구축할 수 있습니다.

OVAL(Open Vulnerability Assessment Language)은 SCAP에서 필수적이고 오래된 구성 요소입니다. 다른 툴 및 사용자 지정 스크립트와 달리 OVAL은 선언적 방식으로 필요한 리소스 상태를 설명합니다.

OVAL 코드는 직접 실행되지 않지만 scanner라는 OVAL 인터프리터 툴을 사용합니다. OVAL의 선언적 특성은 평가된 시스템의 상태가 실수로 수정되지 않도록 합니다.

다른 모든 SCAP 구성 요소와 마찬가지로, OVAL은 XML을 기반으로 합니다. SCAP 표준은 여러 문서 형식을 정의합니다. 각각 다른 유형의 정보를 포함하며 다른 목적을 제공합니다.

Red Hat 제품 보안 팀은 Red Hat 고객에게 영향을 미치는 모든 보안 문제를 추적하고 조사하여 고객이 위험을 평가하고 관리할 수 있도록 지원합니다. Red Hat 고객 포털에서 적시에 간결한 패치 및 보안 공지를 제공합니다. Red Hat은 OVAL 패치 정의를 생성하고 지원하므로 시스템에서 읽을 수 있는 보안 권고 버전을 제공합니다.

플랫폼, 버전 및 기타 요인 간의 차이로 인해 취약점의 Red Hat 제품 보안 정성 등급 은 타사가 제공하는 CVSS(Common Vulnerability Scoring System) 기준 평가 평가와 직접적으로 일치하지 않습니다. 따라서 타사가 제공하는 정의 대신 RHSA OVAL 정의를 사용하는 것이 좋습니다.

RHSA OVAL 정의는 개별적으로 사용할 수 있으며 전체 패키지로 제공되며 Red Hat 고객 포털에서 사용 가능한 새로운 보안 권고를 1시간 이내에 업데이트할 수 있습니다.

각 OVAL 패치 정의는 one-to-one을 RHSA(Red Hat Security Advisory)에 매핑합니다. RHSA에는 여러 취약점에 대한 수정 사항이 포함될 수 있으므로 각 취약점은 CVE(Common Vulnerabilities and Exposures) 이름으로 별도로 나열되며 공개 버그 데이터베이스에 해당 항목에 대한 링크가 있습니다.

RHSA OVAL 정의는 시스템에 설치된 취약한 버전의 RPM 패키지를 확인하도록 설계되었습니다. 예를 들어 이러한 정의를 확장하여 추가 검사를 포함하여 패키지가 취약한 구성에서 사용 중인지 확인할 수 있습니다. 이러한 정의는 Red Hat에서 제공하는 소프트웨어 및 업데이트를 포함하도록 설계되었습니다. 타사 소프트웨어의 패치 상태를 감지하려면 추가 정의가 필요합니다.

참고

컨테이너 또는 컨테이너 이미지에서 보안 취약점을 스캔하려면 8.9절. “컨테이너 및 컨테이너 이미지에서 취약점 스캔” 을 참조하십시오.

추가 리소스

- Red Hat 및 OVAL 호환성
- Red Hat 및 CVE 호환성

- *제품 보안 개요의 알림 및 공지*
- *보안 데이터 지표*
- *8.9절. “컨테이너 및 컨테이너 이미지에서 취약점 스캔”*

8.2.2. 취약점 스캔

***oscap** 명령줄 유틸리티를 사용하면 로컬 시스템을 스캔하고, 구성 준수 콘텐츠를 검증하고, 이러한 스캔 및 평가를 기반으로 보고서 및 가이드를 생성할 수 있습니다. 이 유틸리티는 **OpenSCAP** 라이브러리의 프런트엔드 역할을 하며 처리하는 **SCAP** 콘텐츠 유형에 따라 해당 기능을 모듈(하위 명령)에 그룹화합니다.*

절차

1. ***openscap-scanner** 및 **EgressIP 2** 패키지를 설치합니다.*

```
~]# yum install openscap-scanner bzip2
```

2. *시스템의 최신 **RHSA OVAL** 정의를 다운로드합니다. 예를 들면 다음과 같습니다.*

```
~]# wget -O - https://www.redhat.com/security/data/oval/v2/RHEL7/rhel-7.oval.xml.bz2 |
bzip2 --decompress > rhel-7.oval.xml
```

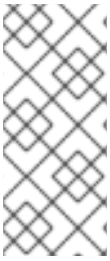
3. *시스템에서 취약점을 스캔하고 결과를 **vulnerability.html** 파일에 저장합니다.*

```
~]# oscap oval eval --report vulnerability.html rhel-7.oval.xml
```

검증

1. *선택한 브라우저의 결과를 확인합니다. 예를 들면 다음과 같습니다.*

```
~]$ firefox vulnerability.html &
```



참고

CVE OVAL 검사에서 취약점을 검색합니다. 따라서 결과 "True"는 시스템이 취약하다는 것을 의미하지만 "False"는 검사에서 취약점을 발견하지 않았음을 의미합니다. HTML 보고서에서 이 값은 결과 행의 색상으로 더욱 구분됩니다.

추가 리소스

- oscap(8) 도움말 페이지.
- Red Hat OVAL 정의 목록.

8.2.3. 원격 시스템에서 취약점 스캔

SSH 프로토콜을 통해 oscap-ssh 도구를 사용하여 OpenSCAP 스캐너가 있는 취약점이 원격 시스템에 있는지 확인할 수도 있습니다.

사전 요구 사항

- openscap-scanner 패키지는 원격 시스템에 설치됩니다.
- SSH 서버는 원격 시스템에서 실행되고 있습니다.

절차

1. openscap-utils 및 bzip2 패키지를 설치합니다.

```
~]# yum install openscap-utils bzip2
```

2. 시스템에 대한 최신 RHSA OVAL 정의를 다운로드합니다.

```
~]# wget -O - https://www.redhat.com/security/data/oval/v2/RHEL7/rhel-7.oval.xml.bz2 |
bzip2 --decompress > rhel-7.oval.xml
```

3. machine1 호스트 이름으로 원격 시스템을 스캔하고, 포트 22에서 SSH를 실행하고, joesec 사용자 이름을 사용하여 취약점을 검사하고 결과를 remote-vulnerability.html 파일에 저장합니

다.

```
~]# oscap-ssh joesec@machine1 22 oval eval --report remote-vulnerability.html rhel-7.oval.xml
```

추가 리소스

- **oscap-ssh(8)** 도움말 페이지.
- **Red Hat OVAL** 정의 목록.

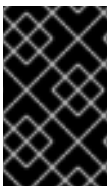
8.3. 구성 규정 준수 검사

8.3.1. RHEL 7의 구성 규정 준수

구성 규정 준수 스캔을 사용하여 특정 조직에서 정의한 기준을 준수할 수 있습니다. 예를 들어, 미국 정부와 협력하는 경우 **OSPP(**운영 체제 보호 프로필**)**를 준수해야 할 수 있으며 결제 프로세서인 경우 **PCI-DSS(Payment Card Industry Data Security Standard)**를 준수해야 할 수 있습니다. 구성 준수 스캔을 수행하여 시스템 보안을 강화할 수도 있습니다.

Red Hat은 영향을 받는 구성 요소에 대한 **Red Hat** 모범 사례가 있기 때문에 **SCAP** 보안 가이드 패키지에 제공된 **SCAP(Security Content Automation Protocol)** 콘텐츠를 따르는 것이 좋습니다.

SCAP 보안 가이드 패키지는 **SCAP 1.2** 및 **SCAP 1.3** 표준을 준수하는 콘텐츠를 제공합니다. **openscap** 스캐너 유틸리티는 **SCAP** 보안 가이드 패키지에 제공된 **SCAP 1.2** 및 **SCAP 1.3** 콘텐츠와 호환됩니다.



중요

구성 규정 준수 스캔을 수행해도 시스템이 규정을 준수하는 것은 아닙니다.

SCAP 보안 가이드 제품군은 여러 플랫폼의 프로필을 데이터 스트림 문서 형태로 제공합니다. 데이터 스트림은 정의, 벤치마크, 프로필 및 개별 규칙이 포함된 파일입니다. 각 규칙은 규정 준수에 대한 적용 가능성 및 요구 사항을 지정합니다. **RHEL 7**은 보안 정책을 준수하기 위해 여러 프로필을 제공합니다. 업계 표준 외에도 **Red Hat** 데이터 스트림에는 실패한 규칙 수정 정보도 포함되어 있습니다.

규정 준수 검사 리소스 구조

```
Data stream
  ├── xccdf
  /     ├── benchmark
  /         ├── profile
  /             ├──rule
  /                 ├── xccdf
  /                     ├── oval reference
  ├── oval                  ├── ocil reference
  ├── ocil                  ├── cpe reference
  └── cpe                   └── remediation
```

*프로필은 **OSPP(운영 체제 보호 프로필)** 또는 **PCI-DSS(Payment Card Industry Data Security Standard)**와 같은 보안 정책을 기반으로 하는 규칙 집합입니다. 이를 통해 보안 표준을 준수하는 자동화된 방식으로 시스템을 감사할 수 있습니다.*

프로필을 수정하여 특정 규칙(예: 암호 길이)을 사용자 지정할 수 있습니다. 프로필 조정에 대한 자세한 내용은 을 참조하십시오. 8.7.2절. "SCAP Workbench를 사용하여 보안 프로필 사용자 정의"

참고

컨테이너 또는 컨테이너 이미지에서 구성 컴플라이언스를 스캔하려면 다음을 참조하십시오. 8.9절. "컨테이너 및 컨테이너 이미지에서 취약점 스캔"

8.3.2. OpenSCAP 스캔의 가능한 결과

***OpenSCAP** 스캔에 적용되는 데이터 스트림 및 프로필과 시스템의 다양한 속성에 따라 각 규칙에서 특정 결과를 생성할 수 있습니다. 이 목록은 가능한 결과의 목록으로 의미에 대한 간략한 설명과 함께 제공됩니다.*

표 8.1. OpenSCAP 스캔의 가능한 결과

결과	설명
통과	검사에서 이 규칙과의 충돌을 찾지 못했습니다.
실패	검사에서 이 규칙과 충돌하는 것을 발견했습니다.
확인되지 않음	OpenSCAP에서는 이 규칙을 자동으로 평가하지 않습니다. 시스템이 이 규칙을 수동으로 준수하는지 확인합니다.
해당 없음	이 규칙은 현재 구성에 적용되지 않습니다.

결과	설명
선택되지 않음	이 규칙은 프로필에 포함되지 않습니다. OpenSCAP은 이 규칙을 평가하지 않으며 결과에 이러한 규칙을 표시하지 않습니다.
오류	검사에 오류가 발생했습니다. 자세한 내용은 **--verboseDEVEL** 옵션을 사용하여 **oscap-scanner** 명령을 입력할 수 있습니다. 버그 보고서를 작성하는 것이 좋습니다.
알 수 없음	검사에 예기치 않은 상황이 발생했습니다. 자세한 내용은 **--verboseDEVEL** 옵션을 사용하여 **oscap-scanner** 명령을 입력할 수 있습니다. 버그 보고서를 작성하는 것이 좋습니다.

8.3.3. 구성 규정 준수 프로필 보기

*검사 또는 해결을 위해 프로필을 사용하기 전에 나열하고 **oscap info** 하위 명령을 사용하여 자세한 설명을 확인할 수 있습니다.*

사전 요구 사항

- ***openscap-scanner** 및 **scap-security-guide** 패키지가 설치됩니다.*

절차

1. ***SCAP** 보안 가이드 프로젝트에서 제공하는 구성 규정 준수 프로필로 사용 가능한 모든 파일을 나열합니다.*

```
~]$ ls /usr/share/xml/scap/ssg/content/
ssg-firefox-cpe-dictionary.xml  ssg-rhel6-ocil.xml
ssg-firefox-cpe-oval.xml        ssg-rhel6-oval.xml
...
ssg-rhel6-ds-1.2.xml            ssg-rhel8-xccdf.xml
ssg-rhel6-ds.xml
...
```

2. ***oscap info** 하위 명령을 사용하여 선택한 데이터 스트림에 대한 세부 정보를 표시합니다. 데이터 스트림을 포함하는 **XML** 파일은 이름에 **-ds** 문자열로 표시됩니다. **Profiles** 섹션에서 사용 가능한 프로필 및 해당 **ID** 목록을 찾을 수 있습니다.*

```
~]$ oscap info /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
...
```

```
Profiles:
 Title: PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 7
  Id: xccdf_org.ssgproject.content_profile_pci-dss
 Title: OSPP - Protection Profile for General Purpose Operating Systems v. 4.2.1
  Id: xccdf_org.ssgproject.content_profile_ospp
...
```

3. 데이터 스트림 파일에서 프로필을 선택하고 선택한 프로필에 대한 추가 세부 정보를 표시합니다. 이렇게 하려면 이전 명령의 출력에 표시된 **ID** 접미사 뒤에 **--profile** 옵션과 함께 **oscap info** 를 사용합니다. 예를 들어 **PCI-DSS** 프로필의 **ID**는 **xccdf_org.ssgproject.content_profile_pci-dss** 이며 **--profile** 옵션의 값은 **_pci-dss** 일 수 있습니다.

```
~]$ oscap info --profile _pci-dss /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
...
Profile
 Title: PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 7
 Id: xccdf_org.ssgproject.content_profile_pci-dss

 Description: Ensures PCI-DSS v3.2.1 related security configuration settings are applied.
...
```

4. 또는 **GUI**를 사용하는 경우 **scap-security-guide-doc** 패키지를 설치하고 웹 브라우저에서 **file:///usr/share/doc/scap-security-guide-doc-0.1.46/sg-rhel7-guide-index.html** 파일을 엽니다. 가이드의 오른쪽 상단에 있는 **Red Hat Enterprise Linux 7** 문서의 보안 구성 필드에 있는 필수 프로필을 선택하면 해당 명령에 이미 포함된 **ID**가 후속 평가를 위해 해당 명령에 포함되어 있는 것을 확인할 수 있습니다.

추가 리소스

- **scap-security-guide(8)** 도움말 페이지에는 프로필 목록도 포함되어 있습니다.

8.3.4. 특정 기준선을 사용하여 구성 규정 준수 평가

시스템이 특정 기준을 준수하는지 확인하려면 다음 단계를 따르십시오.

사전 요구 사항

- **openscap-scanner** 및 **scap-security-guide** 패키지가 설치됩니다.

- 시스템이 준수해야 하는 기준 내에서 프로필의 **ID**를 알고 있습니다. **ID**를 찾으려면 8.3.3절. “구성 규정 준수 프로필 보기” 을 참조하십시오.

절차

1. *선택한 프로파일로 시스템의 규정 준수를 평가하고 검사 결과를 **report.html HTML** 파일에 저장합니다. 예를 들면 다음과 같습니다.*

```
~]$ sudo oscap xccdf eval --report report.html --profile ospp
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

2. *선택 사항: **machine1** 호스트 이름, 포트 **22**에서 실행되는 **SSH, joesec** 사용자 이름을 사용하여 원격 시스템을 스캔하고 결과를 **remote-report.html** 파일에 저장합니다.*

```
~]$ oscap-ssh joesec@machine1 22 xccdf eval --report remote_report.html --profile ospp
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

추가 리소스

- ***scap-security-guide(8)** 도움말 페이지*
- ***file:///usr/share/doc/scap-security-guide-doc-0.1.46/** 디렉터리에 설치된 **SCAP** 보안 가이드 설명서입니다.*
- ***scap- security-guide-doc** 패키지와 함께 설치된 **Red Hat Enterprise Linux 7**의 보안 구성 가이드.*

8.4. 특정 기준선을 사용하여 시스템을 ALIGN으로 수정

*이 절차를 사용하여 특정 기준선과 일치하도록 **RHEL 7** 시스템을 개선하십시오. 이 예제에서는 **Protection Profile for General Purpose Operating Systems (OSPP)**를 사용합니다.*

주의

*신중하게 사용하지 않는 경우 **Remediate** 옵션을 활성화하여 시스템 평가를 실행하면 시스템에 작동하지 않을 수 있습니다. **Red Hat**은 보안 강화 수정으로 인한 변경 사항을 되돌리는 자동화된 방법을 제공하지 않습니다. 수정은 기본 구성의 **RHEL** 시스템에서 지원됩니다. 설치 후 시스템이 변경된 경우 수정을 실행하여 필요한 보안 프로필을 준수하지 못할 수 있습니다.*

사전 요구 사항

- ***scap-security-guide** 패키지가 **RHEL 7** 시스템에 설치되어 있습니다.*

절차

1. ***--remediate** 옵션과 함께 **oscap** 명령을 사용합니다.*

```
~]$ sudo oscap xccdf eval --profile ospp --remediate /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

2. *시스템을 다시 시작합니다.*

검증

1. ***OSPP** 프로필로 시스템 준수를 평가하고 검사 결과를 **ospp_report.html** 파일에 저장합니다.*

```
~]$ oscap xccdf eval --report ospp_report.html --profile ospp
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

추가 리소스

- ***scap-security-guide(8)** 및 **oscap(8)** 도움말 페이지*

8.5. SSG ANSIBLE 플레이북을 사용하여 특정 기준선으로 시스템 수정

*SCAP 보안 가이드 프로젝트의 **Ansible** 플레이북 파일을 사용하여 특정 기준선으로 시스템을 해결하려면 다음 절차를 사용하십시오. 이 예제에서는 **Protection Profile for General Purpose Operating***

Systems (OSPP)를 사용합니다.

주의

신중하게 사용하지 않는 경우 **Remediate** 옵션을 활성화하여 시스템 평가를 실행하면 시스템에 작동하지 않을 수 있습니다. **Red Hat**은 보안 강화 수정으로 인한 변경 사항을 되돌리는 자동화된 방법을 제공하지 않습니다. 수정은 기본 구성의 **RHEL** 시스템에서 지원됩니다. 설치 후 시스템이 변경된 경우 수정을 실행하여 필요한 보안 프로필을 준수하지 못할 수 있습니다.

사전 요구 사항

- **scap-security-guide** 패키지가 **RHEL 7** 시스템에 설치되어 있습니다.
- **ansible** 패키지가 설치되어 있어야 합니다. 자세한 내용은 **Ansible** 설치 가이드를 참조하십시오.

절차

1. **Ansible**을 사용하여 **OSPP**에 맞게 시스템을 교정합니다.

```
~]# ansible-playbook -i localhost, -c local /usr/share/scap-security-guide/ansible/ssg-rhel7-role-ospp.yml
```

2. 시스템을 다시 시작합니다.

검증

1. **OSPP** 프로필로 시스템 준수를 평가하고 검사 결과를 **ospp_report.html** 파일에 저장합니다.

```
~]# oscap xccdf eval --profile ospp --report ospp_report.html /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

추가 리소스

- ***scap-security-guide(8)*** *및* ***oscap(8)*** *도움말 페이지*

- ***Ansible*** *설명서*

8.6. 특정 기준선을 사용하여 시스템을 조정하기 위해 해결 ***ANSIBLE*** *플레이북 생성*

이 절차를 사용하여 시스템을 특정 기준선과 조정하는 데 필요한 수정 사항만 포함하는 ***Ansible*** *플레이북을 생성합니다. 이 예제에서는* ***Protection Profile for General Purpose Operating Systems (OSPP)*** *를 사용합니다. 이 절차를 통해 이미 충족된 요구 사항을 다루지 않는 작은 플레이북을 생성합니다. 이러한 단계를 수행하면 시스템을 어떤 식으로든 수정하지 않으며 이후 애플리케이션을 위한 파일만 준비합니다.*

사전 요구 사항

- ***scap-security-guide*** *패키지가 시스템에 설치되어 있습니다.*

절차

1. *시스템을 스캔하고 결과를 저장합니다.*

```
~]# oscap xccdf eval --profile ospp --results ospp-results.xml
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

2. *이전 단계에서 생성한 파일을 기반으로* ***Ansible*** *플레이북을 생성합니다.*

```
~]# oscap xccdf generate fix --fix-type ansible --profile ospp --output ospp-remediations.yml
ospp-results.xml
```

3. ***ospp-remediations.yml*** *파일에는* ***1****단계에서 수행한 검사 중에 실패한 규칙에 대한* ***Ansible*** *수정이 포함되어 있습니다. 생성된 파일을 검토한 후* ***ansible-playbook ospp-remediations.yml*** *명령을 사용하여 적용할 수 있습니다.*

검증

1. *선택한 텍스트 편집기에서* ***ospp-remediations.yml*** *파일에* ***1****단계에서 수행된 검사에 실패한 규칙이 포함되어 있는지 검토합니다.*

추가 리소스

- *scap-security-guide(8) 및 oscap(8) 도움말 페이지*
- *Ansible 설명서*

8.7. SCAP WORKBENCH를 사용하여 사용자 지정된 프로필로 시스템 스캔

SCAP Workbench 는 단일 로컬 또는 원격 시스템에서 구성 스캔을 수행하고 시스템 수정을 수행하고 검사 평가를 기반으로 보고서를 생성할 수 있는 그래픽 유틸리티입니다. **SCAP Workbench**에는 **oscap** 명령줄 유틸리티에 비해 기능이 제한되어 있습니다. **SCAP Workbench** 는 데이터 스트림 파일 형식으로 보안 콘텐츠를 처리합니다.

8.7.1. SCAP Workbench를 사용하여 시스템을 스캔 및 해결

선택한 보안 정책에 대해 시스템을 평가하려면 다음 절차를 사용하십시오.

사전 요구 사항

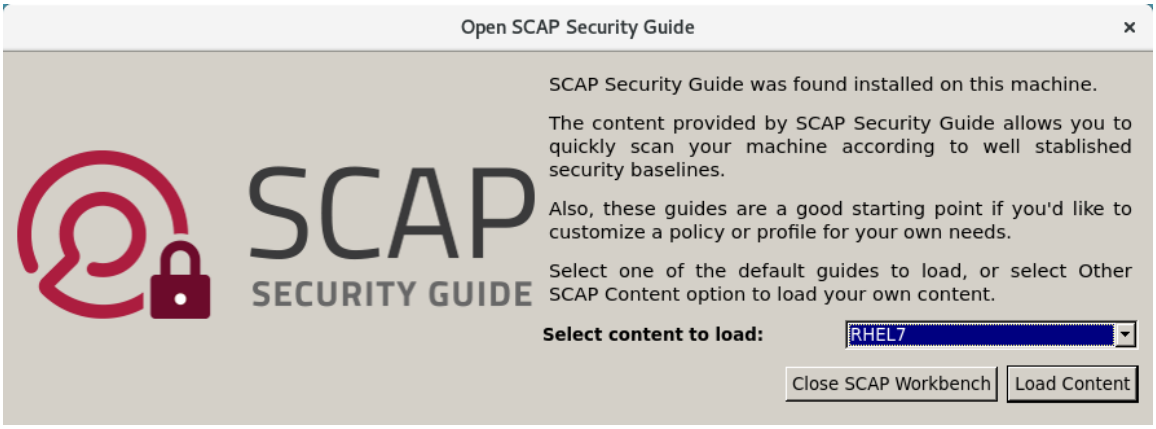
- **scap-workbench** 패키지가 시스템에 설치되어 있습니다.

절차

1. **GNOME Classic** 데스크탑 환경에서 **SCAP Workbench**를 실행하려면 **Super** 키를 눌러 **Activities Overview**를 입력하고 **scap-workbench**를 입력한 다음 **Enter** 키를 누릅니다. 또는 다음을 사용합니다.

```
~]$ scap-workbench &
```

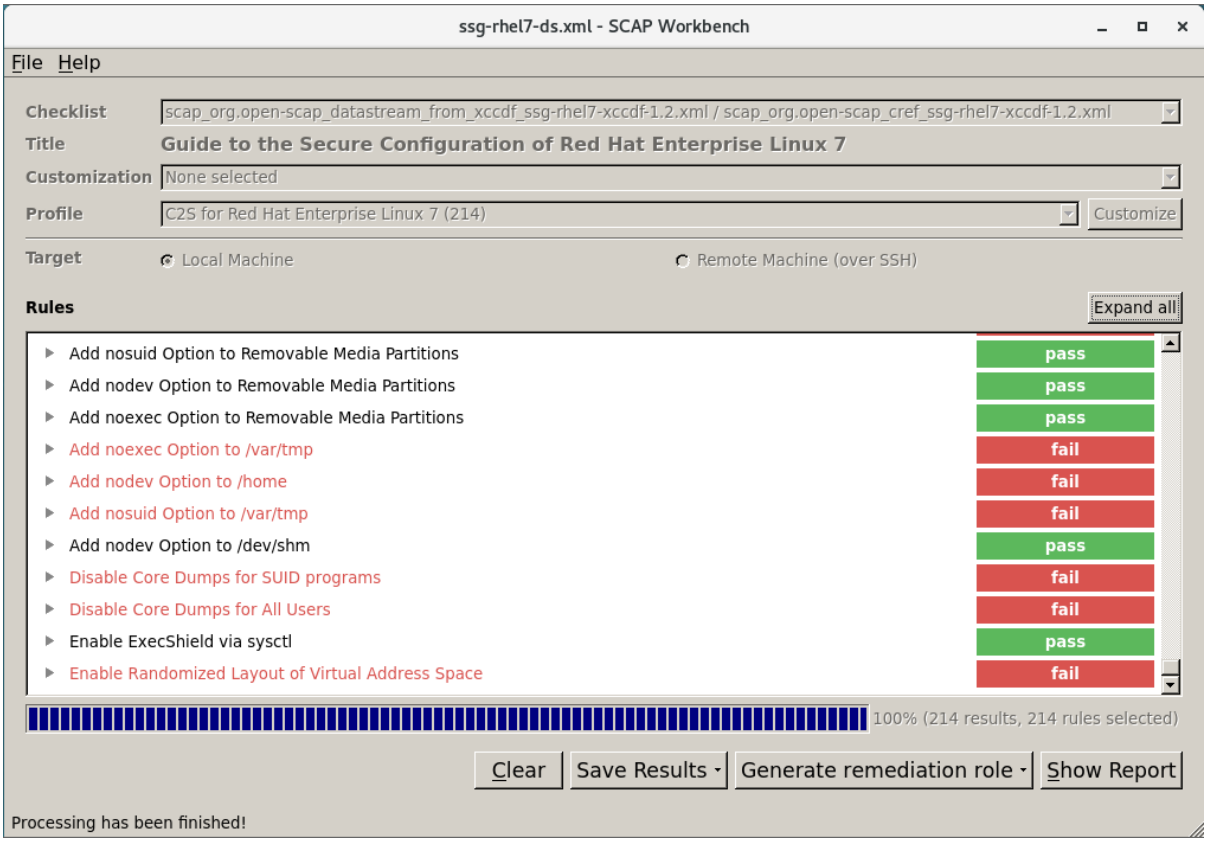
2. 다음 옵션 중 하나를 사용하여 보안 정책을 선택합니다.
   - 시작 창에서 콘텐츠 로드 버튼
   - **SCAP** 보안 가이드에서 콘텐츠 열기
   - **File** (파일) 메뉴에서 기타 콘텐츠를 열고 해당 **XCCDF, SCAP RPM** 또는 데이터 스트림 파일을 검색합니다.



3. ***Remediate** 확인란을 선택하여 시스템 구성을 자동 수정할 수 있습니다. 이 옵션을 활성화하면 **SCAP Workbench**는 정책에서 적용하는 보안 규칙에 따라 시스템 구성을 변경합니다. 이 프로세스에서는 시스템 검사 중에 실패하는 관련 검사를 수정하려고 합니다.*

> *주의*
>
> *신중하게 사용하지 않는 경우 **Remediate** 옵션을 활성화하여 시스템 평가를 실행하면 시스템에 작동하지 않을 수 있습니다. **Red Hat**은 보안 강화 수정으로 인한 변경 사항을 되돌리는 자동화된 방법을 제공하지 않습니다. 수정은 기본 구성의 **RHEL** 시스템에서 지원됩니다. 설치 후 시스템이 변경된 경우 수정을 실행하여 필요한 보안 프로필을 준수하지 못할 수 있습니다.*

4. ***Scan** 버튼을 클릭하여 선택한 프로필로 시스템을 스캔합니다.*



5. *검사 결과를 **XCCDF, ARF** 또는 **HTML** 파일의 형식으로 저장하려면 **Save Results** 콤보 상자를 클릭합니다. **HTML Report** 옵션을 선택하여 사람이 읽을 수 있는 형식으로 스캔 보고서를 생성합니다. **XCCDF** 및 **ARF**(데이터 스트림) 형식은 추가 자동 처리에 적합합니다. 세 가지 옵션을 모두 반복적으로 선택할 수 있습니다.*

6. *결과 기반 수정을 파일로 내보내려면 **Generate remediation** 역할 팝업 메뉴를 사용합니다.*

8.7.2. SCAP Workbench를 사용하여 보안 프로필 사용자 정의

특정 규칙(예: 최소 암호 길이)에서 매개 변수를 변경하고, 다른 방식으로 적용되는 규칙을 제거하고 추가 규칙을 선택하여 내부 정책을 구현하여 보안 프로필을 사용자 지정할 수 있습니다. 프로필을 사용자 지정하여 새 규칙을 정의할 수 없습니다.

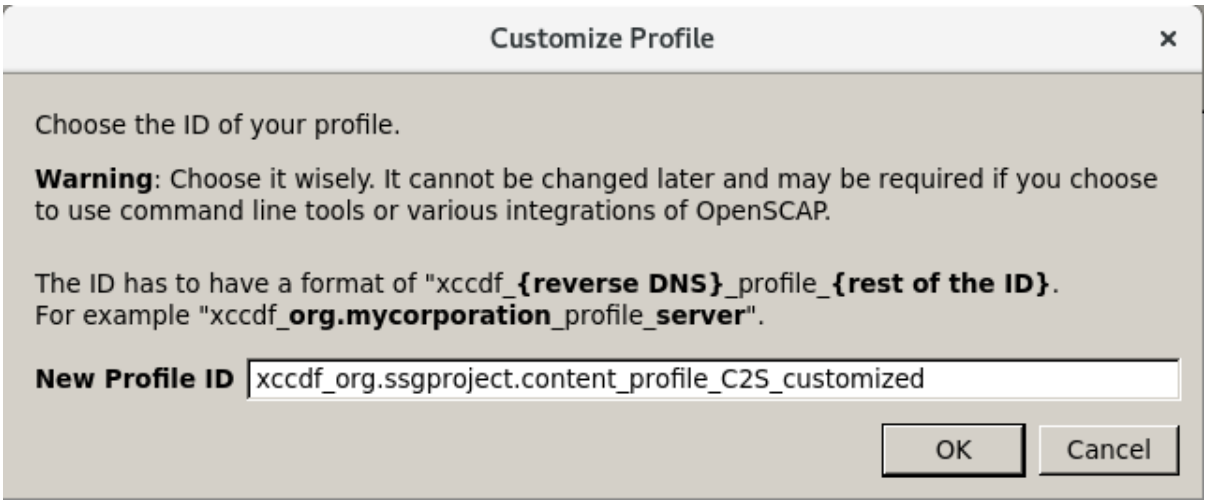
*다음 절차에서는 프로필을 사용자 지정하는 데 **SCAP Workbench**를 사용하는 방법을 보여줍니다. **oscap** 명령줄 유틸리티와 함께 사용할 맞춤형 프로필을 저장할 수도 있습니다.*

절차

1. ***SCAP Workbench** 를 실행하고 **SCAP** 보안 가이드에서 콘텐츠 열기 또는 파일 메뉴에서 기타 콘텐츠 를 열어 사용자 지정할 프로필을 선택합니다.*

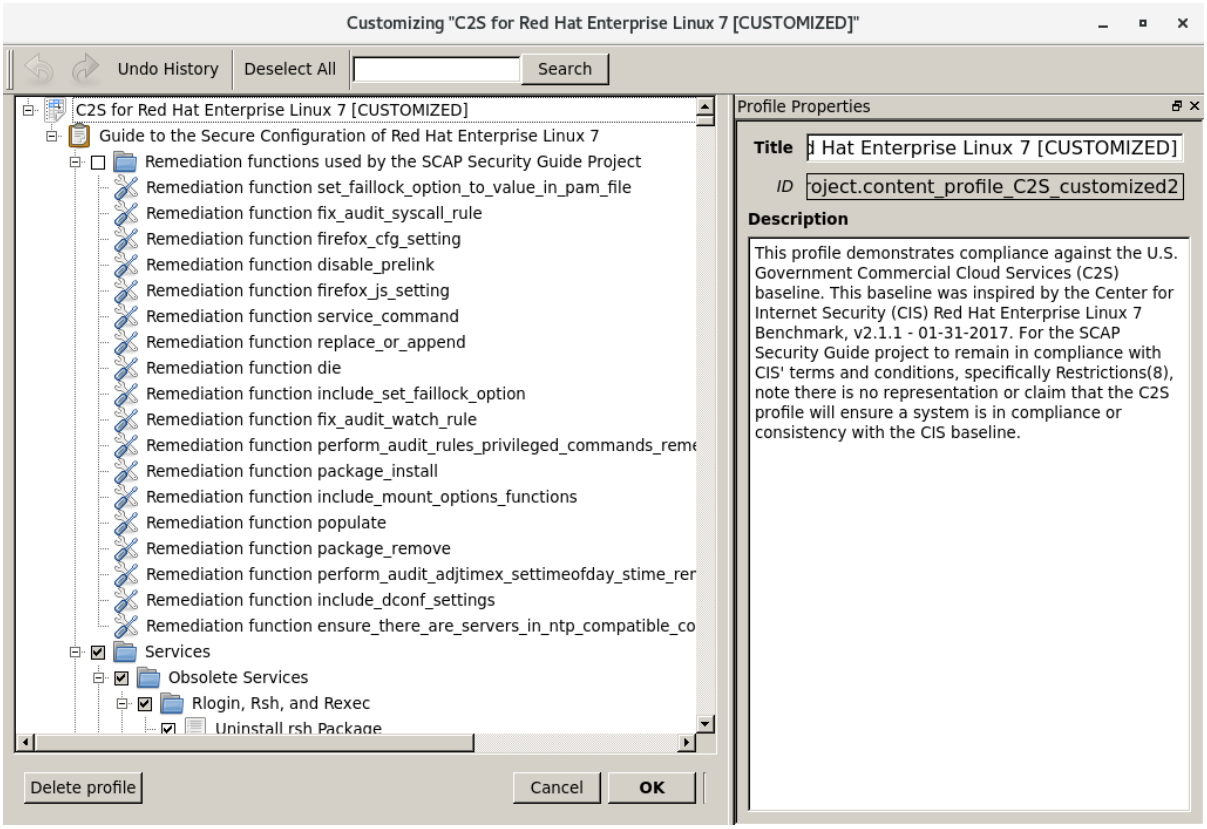
2. *요구 사항에 따라 선택한 보안 프로필을 조정하려면 사용자 지정 버튼을 클릭합니다.*

*그러면 원래 **XCCDF** 파일을 변경하지 않고 현재 선택한 **XCCDF** 프로파일을 수정할 수 있는 새 사용자 지정 창이 열립니다. 새 프로필 **ID**를 선택합니다.*



3. *논리 그룹 또는 **Search** (검색) 필드로 구성된 규칙과 함께 트리 구조를 사용하여 수정하는 규칙을 찾습니다.*

4. *트리 구조의 확인란을 사용하여 규칙을 포함하거나 제외하거나 해당하는 규칙의 값을 수정합니다.*



5. ***OK (확인) 버튼을 클릭하여 변경 사항을 확인합니다.***

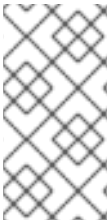
6. *변경 사항을 영구적으로 저장하려면 다음 옵션 중 하나를 사용합니다.*

- ***File** 메뉴에서 **Save Customization only**을 사용하여 사용자 지정 파일을 별도로 저장합니다.*

- *파일 메뉴에서 **Save All** 을 사용하여 모든 콘텐츠를 한 번에 저장합니다.*

***Into a directory** 옵션을 선택하면 **SCAP Workbench** 는 **XCCDF** 또는 데이터 스트림 파일과 사용자 지정 파일을 지정된 위치에 모두 저장합니다. 이를 백업 솔루션으로 사용할 수 있습니다.*

***As RPM** 옵션을 선택하면 **SCAP Workbench**에 데이터 스트림 파일과 사용자 지정 파일이 포함된 **RPM** 패키지를 생성하도록 지시할 수 있습니다. 이 기능은 원격으로 스캔할 수 없는 시스템에 보안 콘텐츠를 배포하고 추가 처리를 위해 콘텐츠를 전달하는 데 유용합니다.*



참고

SCAP Workbench 는 맞춤형 프로필의 결과 기반 수정을 지원하지 않으므로 oscap 명령줄 유틸리티와 함께 내보낸 수정을 사용합니다.

8.7.3. 관련 정보

- scap-workbench(8) 도움말 페이지
- SCAP Workbench 사용자 설명서
- Satellite 6.x를 사용하여 사용자 지정된 SCAP 정책 배포 - 맞춤형 스크립트에 대한 기술 자료 문서

8.8. 설치 후 보안 프로필과 일치하는 시스템 배포 IMMEDIATELY

OpenSCAP 제품군을 사용하여 설치 프로세스 직후 OSPP 또는 PCI-DSS와 같은 보안 프로필을 준수하는 RHEL 시스템을 배포할 수 있습니다. 이 배포 방법을 사용하여 나중에 해결 스크립트를 사용하여 적용할 수 없는 특정 규칙을 적용할 수 있습니다 (예: 암호 보안 강도 및 파티셔닝 규칙).

8.8.1. 그래픽 설치를 사용하여 기본 RHEL 시스템 배포

특정 기준과 일치하는 RHEL 시스템을 배포하려면 다음 절차를 사용하십시오. 이 예에서는 OSDPP(General Purpose Operating System)에 보안 프로필을 사용합니다.

사전 요구 사항

- graphical 설치 프로그램으로 부팅되었습니다. OSCAP Anaconda 애드온은 텍스트 전용 설치를 지원하지 않습니다.
- Installation Summary 창에 액세스했습니다.

절차

1. Installation Summary 창에서 Software Selection을 클릭합니다. Software Selection 창이 열립니다.

2. **Base Environment** 창에서 **Server** 환경을 선택합니다. 기본 환경 하나만 선택할 수 있습니다.

3. **Done**을 클릭하여 설정을 적용하고 **Installation Summary** 창으로 돌아갑니다.

4. **Security Policy**를 클릭합니다. **Security Policy** 창이 열립니다.

5. 시스템에서 보안 정책을 활성화하려면 **Apply security policy**을 **ON**으로 전환합니다.

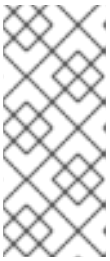
6. 프로필 창에서 **General Purpose Operating Systems**의 **Protection Profile for General Purpose Operating Systems**를 선택합니다.

7. **Select Profile**을 클릭하여 선택을 확인합니다.

8. **Changes that were done or need to be done**을 확인합니다. 나머지 수동 변경 사항을 완료합니다.

9. **OSPP**에는 충족해야 하는 엄격한 파티셔닝 요구 사항이 있으므로 **/boot, /home, /var, /var/log, /var/tmp, /var/log/audit**에 대한 별도의 파티션을 만듭니다.

10. 그래픽 설치 프로세스를 완료합니다.



참고

성공적인 설치 후 그래픽 설치 프로그램은 해당 **Kickstart** 파일을 자동으로 생성합니다. **/root/anaconda-ks.cfg** 파일을 사용하여 **OSPP** 호환 시스템을 자동으로 설치할 수 있습니다.

검증

1. 설치가 완료된 후 시스템의 현재 상태를 확인하려면 시스템을 재부팅하고 새 검사를 시작합니다.

```
~]# oscap xccdf eval --profile ospp --report eval_postinstall_report.html
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

추가 리소스

- 파티셔닝에 대한 자세한 내용은 수동 파티션 구성을 참조하십시오.

8.8.2. Kickstart를 사용하여 Baseline-Compliant RHEL Systems 배포

특정 기준과 일치하는 **RHEL** 시스템을 배포하려면 다음 절차를 사용하십시오. 이 예에서는 **OSDPP(General Purpose Operating System)**에 보안 프로필을 사용합니다.

사전 요구 사항

- **scap-security-guide** 패키지가 시스템에 설치되어 있습니다.

절차

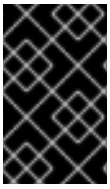
1. 선택한 편집기에서 **/usr/share/scap-security-guide/kickstart/ssg-rhel7-ospp-ks.cfg Kickstart** 파일을 엽니다.

2. 구성 요구 사항에 맞게 파티션 구성표를 업데이트합니다. **OSPP** 규정 준수의 경우 **/boot,/home,/var /log,/var/tmp, /var/log/audit** 에 대한 별도의 파티션을 유지해야 하지만 이러한 파티션의 크기를 변경할 수 있습니다.

주의

OSCAP Anaconda 애드온은 텍스트 전용 설치를 지원하지 않으므로 **Kickstart** 파일에서 텍스트 옵션을 사용하지 마십시오. 자세한 내용은 **RHBZ#1674001**에서 참조하십시오.

3. **Kickstart**를 사용하여 자동 설치를 수행하는 방법에 설명된 대로 **Kickstart** 설치를 시작합니다.



중요

해시 형식의 암호는 **OSPP** 요구 사항에 대해 확인할 수 없습니다.

검증

1. 설치가 완료된 후 시스템의 현재 상태를 확인하려면 시스템을 재부팅하고 새 검사를 시작합니다.

```
~]# oscap xccdf eval --profile ospp --report eval_postinstall_report.html
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

추가 리소스

- 자세한 내용은 **OSCAP Anaconda Add-on** 프로젝트 페이지를 참조하십시오.

8.9. 컨테이너 및 컨테이너 이미지에서 취약점 스캔

이 절차를 사용하여 컨테이너 또는 컨테이너 이미지의 보안 취약점을 찾습니다.

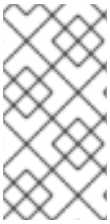
oscap-docker 명령줄 유틸리티 또는 **atomic** 검사 명령줄 유틸리티를 사용하여 컨테이너 또는 컨테이너 이미지에서 보안 취약점을 찾을 수 있습니다.

oscap-docker 를 사용하면 **oscap** 프로그램을 사용하여 컨테이너 이미지 및 컨테이너를 스캔할 수 있습니다.

atomic 스캔 기능을 사용하면 **OpenSCAP** 스캔 기능을 사용하여 시스템에서 컨테이너 이미지 및 컨테이너를 스캔할 수 있습니다. 알려진 **CVE** 취약점 및 구성 준수 여부를 스캔할 수 있습니다. 컨테이너 이미지를 지정된 정책에 수정할 수도 있습니다.

8.9.1. oscap-docker를 사용하여 컨테이너 이미지 및 컨테이너 스캔

oscap-docker 유틸리티를 사용하여 컨테이너 및 컨테이너 이미지를 스캔할 수 있습니다.



참고

oscap-docker 명령을 실행하려면 루트 권한이 필요하며 컨테이너 **ID**는 두 번째 인수입니다.

사전 요구 사항

- **openscap-containers** 패키지가 설치되어 있습니다.

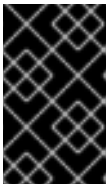
절차

1. 컨테이너 또는 컨테이너 이미지의 **ID**를 찾습니다. 예를 들면 다음과 같습니다.

```
~]# docker images
REPOSITORY                         TAG      IMAGE ID       CREATED       SIZE
registry.access.redhat.com/ubi7/ubi   latest   096cae65a207   7 weeks ago   239 MB
```

2. 컨테이너 또는 컨테이너 이미지에서 취약점을 검사하고 결과를 **vulnerability.html** 파일에 저장합니다.

```
~]# oscap-docker image-cve 096cae65a207 --report vulnerability.html
```



중요

컨테이너를 검사하려면 **image-cve** 인수를 **container-cve** 로 교체합니다.

검증

1. 선택한 브라우저에서 결과를 검사합니다. 예를 들면 다음과 같습니다.

```
~]$ firefox vulnerability.html &
```

추가 리소스

- 자세한 내용은 **oscap-docker(8)** 및 **oscap(8)** 매뉴얼 페이지를 참조하십시오.

8.9.2. 원자 검사를 사용하여 컨테이너 이미지 및 컨테이너 스캔에서 취약점 검색

atomic 검사 유틸리티를 사용하면 컨테이너 및 컨테이너 이미지에서 **Red Hat에서 릴리스한 CVE OVAL 정의에 정의된 알려진 보안 취약점에** 대해 스캔할 수 있습니다. **atomic scan** 명령에는 다음과 같은 형식이 있습니다.

```
~]# atomic scan [OPTIONS] [ID]
```

여기서 **ID** 는 검사할 컨테이너 이미지 또는 컨테이너의 **ID**입니다.

사용 사례

- 모든 컨테이너 이미지를 검사하려면 **--images** 지시문을 사용합니다.
- 모든 컨테이너를 스캔하려면 **--containers** 지시문을 사용합니다.
- 두 유형을 모두 검사하려면 **--all** 지시문을 사용합니다.
- 사용 가능한 모든 명령줄 옵션을 나열하려면 **atomic scan --help** 명령을 사용합니다.

atomic 검사 명령의 기본 검사 유형은 **CVE** 검사 입니다. **Red Hat에서 제공하는 CVE OVAL 정의에 정의된 알려진 보안 취약점의 타겟을 확인하려면** 이를 사용합니다.

사전 요구 사항

- **atomic install rhel7/openscap** 명령을 사용하여 **RHCC(Red Hat Container Catalog)** 에서 **OpenSCAP** 컨테이너 이미지를 다운로드하여 설치했습니다.

절차

1. 정의가 최신 상태인지 확인하기 위해 최신 **OpenSCAP** 컨테이너 이미지가 있는지 확인합니다.

```
~]# atomic help registry.access.redhat.com/rhel7/openscap | grep version
```

중요

Red Hat은 컨테이너 이미지의 매주 업데이트를 제공합니다. 항상 최신 **OpenSCAP** 컨테이너 이미지를 사용하여 **CVE** 스캔 유형에서 사용하는 **OVAL** 정의가 최신 상태인지 확인합니다.

2. 여러 알려진 보안 취약점이 있는 **RHEL 7.2** 컨테이너 이미지를 스캔합니다.

```
~]# atomic scan registry.access.redhat.com/rhel7:7.2
docker run -t --rm -v /etc/localtime:/etc/localtime -v /run/atomic/2017-11-01-14-49-36-614281:/scanin -v /var/lib/atomic/openscap/2017-11-01-14-49-36-614281:/scanout:rw,Z -v /etc/oscapd:/etc/oscapd:ro registry.access.redhat.com/rhel7/openscap oscapd-evaluate scan --no-standard-compliance --targets chroots-in-dir:///scanin --output /scanout

registry.access.redhat.com/rhel7:7.2 (98a88a8b722a718)

The following issues were found:

 RHSA-2017:2832: nss security update (Important)
 Severity: Important
  RHSA URL: https://access.redhat.com/errata/RHSA-2017:2832
  RHSA ID: RHSA-2017:2832-01
  Associated CVEs:
    CVE ID: CVE-2017-7805
    CVE URL: https://access.redhat.com/security/cve/CVE-2017-7805
...
```

추가 리소스

- **Red Hat Enterprise Linux Atomic Host** 제품 문서에 는 **atomic** 명령 사용과 컨테이너에 대한 자세한 설명이 포함되어 있습니다.
- **Red Hat** 고객 포털은 **Atomic CLI(**명령줄 인터페이스**)**에 대한 가이드 를 제공합니다.

8.10. 특정 기준선을 사용하여 컨테이너 또는 컨테이너 이미지의 구성 규정 준수 평가

단계에 따라 컨테이너 또는 컨테이너 이미지의 규정 준수 여부에 따라 특정 보안 기준(예: **Operating System Protection Profile**) 또는 **PCI-DSS(Payment Card Industry Data Security Standard)**와 같은 특정 보안 기준을 평가합니다.

사전 요구 사항

- ***openscap-utils** 및 **scap-security-guide** 패키지가 설치되어 있습니다.*

절차

1. *컨테이너 또는 컨테이너 이미지의 **ID**를 찾습니다. 예를 들면 다음과 같습니다.*

```
~]# docker images
REPOSITORY                         TAG      IMAGE ID       CREATED       SIZE
registry.access.redhat.com/ubi7/ubi   latest   096cae65a207   7 weeks ago   239 MB
```

2. ***OSPP** 프로필로 컨테이너 이미지의 규정 준수를 평가하고 검사 결과를 **report.html HTML** 파일에 저장합니다.*

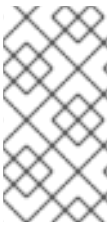
```
~]$ sudo oscap-docker 096cae65a207 xccdf eval --report report.html --profile ospp
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

*설정 준수 여부를 **PCI-DSS** 기준으로 평가하면 **096cae65a207** 을 컨테이너 이미지의 **ID**로, **ospp** 값을 **pci-dss** 로 바꿉니다.*

검증

1. *선택한 브라우저의 결과를 확인합니다. 예를 들면 다음과 같습니다.*

```
~]$ firefox report.html &
```



참고

적용되지 않음으로 표시된 규칙은 컨테이너화된 시스템에 적용되지 않는 규칙입니다. 이러한 규칙은 베어 메탈 또는 가상화 시스템에만 적용됩니다.

추가 리소스

- *자세한 내용은 **oscap-docker(8)** 및 **scap-security-guide(8)** 매뉴얼 페이지를 참조하십시오.*

- ***file:///usr/share/doc/scap-security-guide-doc-0.1.46/** 디렉터리에 설치된 **SCAP** 보안 가이드 설명서입니다.*

8.11. 원자 검사를 사용하여 컨테이너 이미지 및 컨테이너의 구성 검사및 수정

8.11.1. 원자 검사를 사용하여 컨테이너 이미지 및 컨테이너의 구성 규정 준수 검사

이러한 유형의 스캔을 사용하여 **OpenSCAP** 컨테이너 이미지 내에 번들로 제공된 **SCAP** 콘텐츠로 **Red Hat Enterprise Linux** 기반 컨테이너 이미지와 컨테이너를 평가합니다. 이를 통해 **SCAP** 보안 가이드에서 제공하는 모든 프로필과의 스캔이 가능합니다.

참고

atomic 명령 및 컨테이너 사용에 대한 자세한 내용은 **Red Hat Enterprise Linux Atomic Host** 제품 설명서를 참조하십시오. **Red Hat** 고객 포털은 **Atomic CLI(**명령줄 인터페이스**)**에 대한 가이드 도 제공합니다.

사전 요구 사항

- **atomic install rhel7/openscap** 명령을 사용하여 **RHCC(Red Hat Container Catalog)** 에서 **OpenSCAP** 컨테이너 이미지를 다운로드하여 설치했습니다.

절차

1. **configuration_compliance** 검사에 대해 **OpenSCAP** 이미지에서 제공하는 **SCAP** 콘텐츠를 나열합니다.

```
~]# atomic help registry.access.redhat.com/rhel7/openscap
```

Defense 정보 시스템국 보안 기술 구현 가이드**(DISA container Guide)** 정책을 사용하여 최신 **Red Hat Enterprise Linux 7** 컨테이너 이미지의 규정 준수를 확인하고 검사에서 **HTML** 보고서를 생성합니다.

```
~]# atomic scan --scan_type configuration_compliance --scanner_args xccdf-
id=scap_org.open-scap_cref_ssg-rhel7-xccdf-
1.2.xml,profile=xccdf_org.ssgproject.content_profile_stig-rhel7-disa,report
registry.access.redhat.com/rhel7:latest
```

이전 명령의 출력에는 마지막에 검사와 연결된 파일에 대한 정보가 포함되어 있습니다.

```
.....
```

```
Files associated with this scan are in /var/lib/atomic/openscap/2017-11-03-13-35-34-296606.

~]# tree /var/lib/atomic/openscap/2017-11-03-13-35-34-296606
/var/lib/atomic/openscap/2017-11-03-13-35-34-296606
├── db7a70a0414e589d7c8c162712b329d4fc670fa47ddde721250fb9fcdbed9cc2
│   ├── arf.xml
│   ├── fix.sh
│   ├── json
│   └── report.html
└── environment.json

1 directory, 5 files
```

atomic 검사 에서는 모든 결과가 포함된 하위 디렉터리를 생성하고 **/var/lib/atomic/openscap/** 디렉터리의 검사에서 보고합니다. 결과가 포함된 **arf.xml** 파일은 구성 준수에 대한 모든 스캔에 생성됩니다. 사람이 읽을 수 있는 **HTML** 보고서 파일을 생성하려면 **--scanner_args** 옵션에 **report suboption**을 추가합니다.

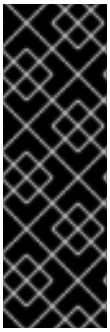
2. 선택 사항: **DISAProgress Viewer** 가 읽을 수 있는 **XCCDF** 결과를 생성하려면 **--scanner_args** 옵션에 **stig-viewer** 하위 옵션을 추가합니다. 결과는 **stig.xml** 에 표시됩니다.

참고

--scanner_args 옵션의 **xccdf-id** 하위 옵션이 생략되면 스캐너는 선택한 데이터 스트림 파일의 첫 번째 **XCCDF** 구성 요소에서 프로필을 검색합니다. 데이터 스트림 파일에 대한 자세한 내용은 **8.3.1절. "RHEL 7의 구성 규정 준수"** 을 참조하십시오.

8.11.2. 원자 검사를사용하여 컨테이너 이미지 및 컨테이너의 구성 규정 준수 수정

원래 컨테이너 이미지에 대해 구성 컴플라이언스 검사를 실행하여 **DISAonnectionFactory** 정책 준수를 확인할 수 있습니다. 검사 결과에 따라 실패한 검사 결과에 대한 **bash** 수정이 포함된 수정 스크립트가 생성됩니다. 그러면 수정 스크립트가 원래 컨테이너 이미지에 적용됩니다. 이를 수정이라고 합니다. 수정으로 인해 변경된 구성이 포함된 컨테이너 이미지가 원본 컨테이너 이미지 위에 새 계층으로 추가됩니다.



중요

원래 컨테이너 이미지는 변경되지 않고 새 계층만 생성됩니다. 수정 프로세스는 모든 구성 개선 사항이 포함된 새 컨테이너 이미지를 빌드합니다. 이 계층의 콘텐츠는 스캔의 보안 정책에 따라 정의되며 이전의 경우 **DISA STIG** 정책입니다. 또한 수정된 컨테이너 이미지가 포함된 **Red Hat**에서 더 이상 서명하지 않습니다. 이는 수정 계층을 포함하여 원래 컨테이너 이미지와 다르기 때문에 예상되는 것입니다.

사전 요구 사항

- ***atomic install rhel7/openscap** 명령을 사용하여 RHCC(Red Hat Container Catalog) 에서 **OpenSCAP** 컨테이너 이미지를 다운로드하여 설치했습니다.*

절차

1. ***configuration_compliance** 검사에 대해 **OpenSCAP** 이미지에서 제공하는 **SCAP** 콘텐츠를 나열합니다.*

```
~]# atomic help registry.access.redhat.com/rhel7/openscap
```

2. *컨테이너 이미지를 지정된 정책에 해결하려면 구성 컴플라이언스를 스캔할 때 **--remediate** 옵션을 **atomic** 검사 명령에 추가합니다. 다음 명령은 **Red Hat Enterprise Linux 7** 컨테이너 이미지에서 **DISA810** 정책과 호환되는 새로운 수정 컨테이너 이미지를 빌드합니다.*

```
~]# atomic scan --remediate --scan_type configuration_compliance --scanner_args
profile=xccdf_org.ssgproject.content_profile_stig-rhel7-disa,report
registry.access.redhat.com/rhel7:latest

registry.access.redhat.com/rhel7:latest (db7a70a0414e589)

The following issues were found:
.............
  Configure Time Service Maxpoll Interval
  Severity: Low
   XCCDF result: fail

  Configure LDAP Client to Use TLS For All Transactions
  Severity: Moderate
   XCCDF result: fail
.............
Remediating rule 43/44: 'xccdf_org.ssgproject.content_rule_chronyd_or_ntpd_set_maxpoll'
Remediating rule 44/44: 'xccdf_org.ssgproject.content_rule_ldap_client_start_tls'

Successfully built 9bbc7083760e
Successfully built remediated image 9bbc7083760e from
db7a70a0414e589d7c8c162712b329d4fc670fa47ddde721250fb9fcdbed9cc2.

Files associated with this scan are in /var/lib/atomic/openscap/2017-11-06-13-01-42-785000.
```

3. *선택 사항: **atomic** 검사 명령의 출력에는 수정된 이미지 **ID**가 보고됩니다. 이미지를 쉽게 기억할 수 있도록 하려면 다음과 같이 일부 이름으로 태그를 지정합니다.*

```
~]# docker tag 9bbc7083760e rhel7_disa_stig
```

8.12. RHEL 7에서 지원되는 SCAP 보안 가이드 프로필

RHEL의 특정 마이너 릴리스에서 제공된 SCAP 콘텐츠만 사용합니다. 강화에 참여하는 구성 요소가 새 기능으로 정기적으로 업데이트되기 때문입니다. 이러한 업데이트를 반영하기 위해 SCAP 콘텐츠 변경이 필요하지만 이전 버전과 항상 호환되지는 않습니다.

다음 표에서는 프로필이 정렬되는 정책 버전과 함께 RHEL의 각 마이너 버전에서 제공되는 프로필을 찾을 수 있습니다.

표 8.2. RHEL 7.9에서 지원되는 SCAP 보안 가이드 프로필

프로파일 이름	프로파일 ID	정책 버전
레벨 2의 CIS Red Hat Enterprise Linux 7 벤치마크 - 서버	**xccdf_org.ssgproject.content_profile_cis**	RHEL 7.9.9 이상:2.2.0 RHEL 7.9.10 이상:3.1.1
CIS Red Hat Enterprise Linux 7 벤치마크 수준 1 - 서버	**xccdf_org.ssgproject.content_profile_cis_server_l1**	RHEL 7.9.10 이상:3.1.1
Red Hat Enterprise Linux 7 벤치마크 수준 1 - 워크스테이션	**xccdf_org.ssgproject.content_profile_cis_workstation_l1**	RHEL 7.9.10 이상:3.1.1
레벨 2 - 워크스테이션의 CIS Red Hat Enterprise Linux 7 벤치마크	**xccdf_org.ssgproject.content_profile_cis_workstation_l2**	RHEL 7.9.10 이상:3.1.1
정보 시스템 (ANSSI) BP-028의 보안 수준을 위한 프랑스어 국가국	**xccdf_org.ssgproject.content_profile_anssi_nt28_enhanced**	RHEL 7.9.4 이상:draft RHEL 7.9.5 이상:1.2
정보 시스템(ANSSI) BP-028 고가용성을 위한 프랑스어 국가국	**xccdf_org.ssgproject.content_profile_anssi_nt28_high**	RHEL 7.9.6 이상:draft RHEL 7.9.7 이상:1.2
정보 시스템 (ANSSI) BP-028 인테리어 수준을 위한 프랑스 국가국	**xccdf_org.ssgproject.content_profile_anssi_nt28_intermediary**	RHEL 7.9.4 이상: 프로젝트 RHEL 7.9.5 이상:1.2
정보 시스템 (ANSSI) BP-028 Minimal Level용 프랑스 국가국	**xccdf_org.ssgproject.content_profile_anssi_nt28_minimal**	RHEL 7.9.4 이상:draft RHEL 7.9.5 이상:1.2
C2S for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_C2S**	버전이 지정되지 않음
CJIS(Cributor Justice Information Services) 보안 정책	**xccdf_org.ssgproject.content_profile_cjis**	5.4
비정형 정보 시스템 및 조직(NIST 800-171)에서 분류되지 않은 정보	**xccdf_org.ssgproject.content_profile_cui**	r1

프로파일 이름	프로파일 ID	정책 버전
ACS(Austricial Security Center) Essential Eight	**xccdf_org.ssgproject.content_profile_e8**	버전이 지정되지 않음
HIPAA(Health Insurance Portability and Accountability Act)	**xccdf_org.ssgproject.content_profile_hipaa**	버전이 지정되지 않음
NIST 국가 체크리스트 프로그램 보안 가이드	**xccdf_org.ssgproject.content_profile_ncp**	버전이 지정되지 않음
OSPP - 범용 운영 체제 v4.2.1용 프로필 보호	**xccdf_org.ssgproject.content_profile_ospp**	4.2.1
PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_pci-dss_centric**	3.2.1
PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_pci-dss**	3.2.1
[DRAFT] RHELH(Red Hat Enterprise Linux Virtualization Host)의 DISA organization	**xccdf_org.ssgproject.content_profile_rhelh-stig**	초안
VPP - 가상화용 프로필 보호 v. RHELH(Red Hat Enterprise Linux Hypervisor)용 1.0	**xccdf_org.ssgproject.content_profile_rhelh-vpp**	1.0
Red Hat Corporate Profile for Certified Cloud Provider (RH CCP)	**xccdf_org.ssgproject.content_profile_rht-ccp**	버전이 지정되지 않음
Standard System Security Profile for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_standard**	버전이 지정되지 않음
DISA STIG for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_stig**	RHEL 7.9.0 및 7.9.1: 1.4 RHEL 7.9.2에서 7.9.4로: V3R1 RHEL 7.9.5 및 7.9.6:V3R2 RHEL 7.9.7에서 RHEL 7.9.9:V3R3 RHEL 7.9.10 및 RHEL 7.9.11:V3R5 RHEL 7.9.12 이상:V3R6
Red Hat Enterprise Linux 7용 GUI가 포함된 DISAhostname	**xccdf_org.ssgproject.content_profile_stig_gui**	RHEL 7.9.7에서 RHEL 7.9.9:V3R3 RHEL 7.9.10 및 RHEL 7.9.11:V3R5 RHEL 7.9.12 이상:V3R6

표 8.3. RHEL 7.8에서 지원되는 SCAP 보안 가이드 프로필

프로파일 이름	프로파일 ID	정책 버전
DRAFT - ANSSI DAT-NT28 (enhanced)	**xccdf_org.ssgproject.content_profile_**anssi_nt28_enhanced	초안
DRAFT - ANSSI DAT-NT28 (high)	**xccdf_org.ssgproject.content_profile_**anssi_nt28_high	초안
DRAFT - ANSSI DAT-NT28 (인터미리)	**xccdf_org.ssgproject.content_profile_**anssi_nt28_intermediary	초안
DRAFT - ANSSI DAT-NT28 (최소)	**xccdf_org.ssgproject.content_profile_**anssi_nt28_minimal	초안
C2S for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_**C2S	버전이 지정되지 않음
CJIS(Cributor Justice Information Services) 보안 정책	**xccdf_org.ssgproject.content_profile_**cjis	5.4
비정형 정보 시스템 및 조직(NIST 800-171)에서 분류되지 않은 정보	**xccdf_org.ssgproject.content_profile_**cui	r1
ACS(Austricial Security Center) Essential Eight	**xccdf_org.ssgproject.content_profile_**e8	버전이 지정되지 않음
HIPAA(Health Insurance Portability and Accountability Act)	**xccdf_org.ssgproject.content_profile_**hipaa	버전이 지정되지 않음
NIST 국가 체크리스트 프로그램 보안 가이드	**xccdf_org.ssgproject.content_profile_**ncp	버전이 지정되지 않음
OSPP - 범용 운영 체제 v4.2.1용 프로필 보호	**xccdf_org.ssgproject.content_profile_**ospp	4.2.1
PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_**pci-dss_centric	3.2.1
PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_**pci-dss	3.2.1
[DRAFT] RHELH(Red Hat Enterprise Linux Virtualization Host)의 DISA organization	**xccdf_org.ssgproject.content_profile_**rhelh-stig	초안

프로파일 이름	프로파일 ID	정책 버전
VPP - 가상화용 프로필 보호 v. RHELH(Red Hat Enterprise Linux Hypervisor)용 1.0	**xccdf_org.ssgproject.content_profile_rhelh-vpp**	1.0
Red Hat Corporate Profile for Certified Cloud Provider (RH CCP)	**xccdf_org.ssgproject.content_profile_rht-ccp**	버전이 지정되지 않음
Standard System Security Profile for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_standard**	버전이 지정되지 않음
DISA STIG for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_stig**	1.4

표 8.4. RHEL 7.7에서 지원되는 SCAP 보안 가이드 프로필

프로파일 이름	프로파일 ID	정책 버전
C2S for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_C2S**	버전이 지정되지 않음
CJIS(Cributor Justice Information Services) 보안 정책	**xccdf_org.ssgproject.content_profile_cjis**	5.4
HIPAA(Health Insurance Portability and Accountability Act)	**xccdf_org.ssgproject.content_profile_hipaa**	버전이 지정되지 않음
비정형 정보 시스템 및 조직(NIST 800-171)에서 분류되지 않은 정보	**xccdf_org.ssgproject.content_profile_nist-800-171-cui**	r1
OSPP - 범용 운영 체제 v용 프로필 보호. 4.2	**xccdf_org.ssgproject.content_profile_ospp42**	4.2
미국 정부 구성 기준	**xccdf_org.ssgproject.content_profile_ospp**	3.9
PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_pci-dss_centric**	3.2.1
PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_pci-dss**	3.2.1
VPP - 가상화용 프로필 보호 v. RHELH(Red Hat Enterprise Linux Hypervisor)용 1.0	**xccdf_org.ssgproject.content_profile_rhelh-vpp**	1.0

프로파일 이름	프로파일 ID	정책 버전
Red Hat Corporate Profile for Certified Cloud Provider (RH CCP)	**xccdf_org.ssgproject.content_profile_rht-ccp**	버전이 지정되지 않음
Standard System Security Profile for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_standard**	버전이 지정되지 않음
DISA STIG for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_stig-rhel7-disa**	1.4

표 8.5. RHEL 7.6에서 지원되는 SCAP 보안 가이드 프로필

프로파일 이름	프로파일 ID	정책 버전
C2S for Red Hat Enterprise Linux 7	xccdf_org.ssgproject.content_profile_**C2S**	버전이 지정되지 않음
CJIS(Cributor Justice Information Services) 보안 정책	**xccdf_org.ssgproject.content_profile_cjis**	5.4
HIPAA(Health Insurance Portability and Accountability Act)	**xccdf_org.ssgproject.content_profile_hipaa**	버전이 지정되지 않음
비정형 정보 시스템 및 조직(NIST 800-171)에서 분류되지 않은 정보	**xccdf_org.ssgproject.content_profile_nist-800-171-cui**	r1
OSPP - 범용 운영 체제 v용 프로필 보호. 4.2	**xccdf_org.ssgproject.content_profile_ospp42**	4.2
미국 정부 구성 기준	**xccdf_org.ssgproject.content_profile_ospp**	3.9
Red Hat Enterprise Linux 7에 대한 PCI-DSS v3 제어 기준	**xccdf_org.ssgproject.content_profile_pci-dss_centric**	3.1
Red Hat Enterprise Linux 7에 대한 PCI-DSS v3 제어 기준	**xccdf_org.ssgproject.content_profile_pci-dss**	3.1
Red Hat Corporate Profile for Certified Cloud Provider (RH CCP)	**xccdf_org.ssgproject.content_profile_rht-ccp**	버전이 지정되지 않음
Standard System Security Profile for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_standard**	버전이 지정되지 않음

프로파일 이름	프로파일 ID	정책 버전
DISA STIG for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_stig-rhel7-disa**	1.4

표 8.6. RHEL 7.5에서 지원되는 SCAP 보안 가이드 프로필

프로파일 이름	프로파일 ID	정책 버전
C2S for Red Hat Enterprise Linux	**xccdf_org.ssgproject.content_profile_C2S**	버전이 지정되지 않음
CJIS(Cributor Justice Information Services) 보안 정책	**xccdf_org.ssgproject.content_profile_cjis-rhel7-server**	5.4
일반적인 목적의 시스템에 대한 일반적인 프로파일	**xccdf_org.ssgproject.content_profile_common**	버전이 지정되지 않음
표준 Docker 호스트 보안 프로필	**xccdf_org.ssgproject.content_profile_docker-host**	버전이 지정되지 않음
비정형 정보 시스템 및 조직(NIST 800-171)에서 분류되지 않은 정보	**xccdf_org.ssgproject.content_profile_nist-800-171-cui**	r1
미국 정부 구성 기준선(USGCB / StatefulSet) - DRAFT	**xccdf_org.ssgproject.content_profile_ospp-rhel7**	3.9
Red Hat Enterprise Linux 7에 대한 PCI-DSS v3 제어 기준	**xccdf_org.ssgproject.content_profile_pci-dss_centric**	3.1
Red Hat Enterprise Linux 7에 대한 PCI-DSS v3 제어 기준	**xccdf_org.ssgproject.content_profile_pci-dss**	3.1
Red Hat Corporate Profile for Certified Cloud Provider (RH CCP)	**xccdf_org.ssgproject.content_profile_rht-ccp**	버전이 지정되지 않음
표준 시스템 보안 프로필	**xccdf_org.ssgproject.content_profile_standard**	버전이 지정되지 않음
DISA STIG for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_stig-rhel7-disa**	1.4
Red Hat Virtualization Hypervisor의 경우 container.	**xccdf_org.ssgproject.content_profile_stig-rhevh-upstream**	1.4

표 8.7. RHEL 7.4에서 지원되는 SCAP 보안 가이드 프로필

프로파일 이름	프로파일 ID	정책 버전
C2S for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_C2S**	버전이 지정되지 않음
CJIS(Cributor Justice Information Services) 보안 정책	**xccdf_org.ssgproject.content_profile_cjis-rhel7-server**	5.4
일반적인 목적의 시스템에 대한 일반적인 프로파일	**xccdf_org.ssgproject.content_profile_common**	버전이 지정되지 않음
표준 Docker 호스트 보안 프로필	**xccdf_org.ssgproject.content_profile_docker-host**	버전이 지정되지 않음
비정형 정보 시스템 및 조직(NIST 800-171)에서 분류되지 않은 정보	**xccdf_org.ssgproject.content_profile_nist-800-171-cui**	r1
미국 정부 구성 기준선(USGCB / StatefulSet) - DRAFT	**xccdf_org.ssgproject.content_profile_ospp-rhel7**	3.9
Red Hat Enterprise Linux 7에 대한 PCI-DSS v3 제어 기준	**xccdf_org.ssgproject.content_profile_pci-dss_centric**	3.1
Red Hat Enterprise Linux 7에 대한 PCI-DSS v3 제어 기준	**xccdf_org.ssgproject.content_profile_pci-dss**	3.1
Red Hat Corporate Profile for Certified Cloud Provider (RH CCP)	**xccdf_org.ssgproject.content_profile_rht-ccp**	버전이 지정되지 않음
표준 시스템 보안 프로필	**xccdf_org.ssgproject.content_profile_standard**	버전이 지정되지 않음
DISA STIG for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_stig-rhel7-disa**	1.4
Red Hat Virtualization Hypervisor의 경우 container.	**xccdf_org.ssgproject.content_profile_stig-rhevh-upstream**	

표 8.8. RHEL 7.3에서 지원되는 SCAP 보안 가이드 프로필

프로파일 이름	프로파일 ID	정책 버전
C2S for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_C2S**	버전이 지정되지 않음
CJIS(Cributor Justice Information Services) 보안 정책	**xccdf_org.ssgproject.content_profile_cjis-rhel7-server**	5.4

프로파일 이름	프로파일 ID	정책 버전
일반적인 목적의 시스템에 대한 일반적인 프로파일	**xccdf_org.ssgproject.content_profile_common**	버전이 지정되지 않음
CNSSI 1253 Low/Low/Low Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_nist-cl-il-al**	버전이 지정되지 않음
미국 정부 구성 기준선(USGCB / StatefulSet)	**xccdf_org.ssgproject.content_profile_ospp-rhel7-server**	버전이 지정되지 않음
Red Hat Enterprise Linux 7에 대한 PCI-DSS v3 제어 기준	**xccdf_org.ssgproject.content_profile_pci-dss**	3.1
Red Hat Corporate Profile for Certified Cloud Provider (RH CCP)	**xccdf_org.ssgproject.content_profile_rht-ccp**	버전이 지정되지 않음
표준 시스템 보안 프로필	**xccdf_org.ssgproject.content_profile_standard**	버전이 지정되지 않음
Red Hat Enterprise Linux 7 Server Running GUIs	**xccdf_org.ssgproject.content_profile_stig-rhel7-server-gui-upstream**	1.4
STIG for Red Hat Enterprise Linux 7 Server	**xccdf_org.ssgproject.content_profile_stig-rhel7-server-upstream**	1.4
STIG for Red Hat Enterprise Linux 7 Workstation	**xccdf_org.ssgproject.content_profile_stig-rhel7-workstation-upstream**	1.4

표 8.9. RHEL 7.2에서 지원되는 SCAP 보안 가이드 프로필

프로파일 이름	프로파일 ID	정책 버전
일반적인 목적의 시스템에 대한 일반적인 프로파일	**xccdf_org.ssgproject.content_profile_common**	버전이 지정되지 않음
draft PCI-DSS v3 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_pci-dss**	초안
Red Hat Corporate Profile for Certified Cloud Provider (RH CCP)	**xccdf_org.ssgproject.content_profile_rht-ccp**	버전이 지정되지 않음

프로파일 이름	프로파일 ID	정책 버전
표준 시스템 보안 프로필	**xccdf_org.ssgproject.content_profile_standard**	버전이 지정되지 않음
Red Hat Enterprise Linux 7 Server의 시험판 Draft vCPUs	**xccdf_org.ssgproject.content_profile_stig-rhel7-server-upstream**	초안

표 8.10. RHEL 7.1에서 지원되는 SCAP 보안 가이드 프로필

프로파일 이름	프로파일 ID	정책 버전
Red Hat Corporate Profile for Certified Cloud Provider (RH CCP)	**xccdf_org.ssgproject.content_profile_rht-ccp**	버전이 지정되지 않음

추가 리소스

- *RHEL 8의 프로필에 대한 자세한 내용은 RHEL 8 에서 지원되는 SCAP 보안 가이드 프로필을 참조하십시오.*

8.13. 관련 정보

- *지원되는 SCAP 보안 가이드 - 이 문서에는 다양한 버전의 RHEL에서 지원되는 SCAP 보안 가이드 버전이 나열되어 있습니다.*
- *OpenSCAP 프로젝트 페이지 - OpenSCAP 프로젝트의 홈 페이지는 SCAP와 관련된 oscap 유틸리티 및 기타 구성 요소 및 프로젝트에 대한 자세한 정보를 제공합니다.*
- *SCAP Workbench 프로젝트 페이지 - SCAP Workbench 프로젝트의 홈 페이지는 scap-workbench 애플리케이션에 대한 자세한 정보를 제공합니다.*
- *SCAP Security Guide(SSG) 프로젝트 페이지 - Red Hat Enterprise Linux에 대한 최신 보안 콘텐츠를 제공하는 SSG 프로젝트의 홈 페이지.*
- *Red Hat 보안 데모: 보안 규정 준수를 자동화할 수 있는 사용자 지정 보안 정책 콘텐츠 생성 - 업계 표준 보안 정책 및 사용자 지정 보안 정책을 모두 준수하기 위해 Red Hat Enterprise Linux에 포함된 툴을 사용하여 보안 규정 준수 자동화에 초기 환경을 제공하는 실습 랩입니다. 팀을 위해 이러한 랩 연습을 교육하거나 액세스하려면 자세한 내용은 Red Hat 계정 팀에 문의하십시오.*

- *Red Hat 보안 데모: RHEL Security Technologies - 실습 랩을 통해 OpenSCAP을 포함한 Red Hat Enterprise Linux에서 사용할 수 있는 주요 보안 기술을 사용하여 RHEL 시스템의 모든 수준에서 보안을 구현하는 방법을 배울 수 있습니다. 팀을 위해 이러한 랩 연습을 교육하거나 액세스하려면 자세한 내용은 Red Hat 계정 팀에 문의하십시오.*

- *National Institute of Standards and Technology (NIST) SCAP 페이지 - 이 페이지는 SCAP 게시, 사양 및 SCAP 검증 프로그램을 포함한 SCAP 관련 자료의 광범위한 컬렉션을 나타냅니다.*

- *NVD(National Vulnerability Database) - 이 페이지는 SCAP 콘텐츠 및 기타 SCAP 표준 기반 취약점 관리 데이터의 가장 큰 리포지토리를 나타냅니다.*

- *Red Hat OVAL 콘텐츠 리포지토리 - Red Hat Enterprise Linux 시스템의 취약성에 대한 OVAL 정의가 포함된 리포지토리입니다. 권장되는 취약점 콘텐츠 소스입니다.*

- *MITRE CVE - MITRE company가 제공하는 알려진 보안 취약점의 데이터베이스입니다. RHEL의 경우 Red Hat에서 제공하는 OVAL CVE 콘텐츠를 사용하는 것이 좋습니다.*

- *MITRE OVAL - 이 페이지는 MITRE 기업이 제공하는 OVAL 관련 프로젝트를 나타냅니다. 다른 OVAL 관련 정보 중에서 이러한 페이지에는 최신 버전의 OVAL 언어 및 수천 개의 OVAL 정의가 포함된 OVAL 콘텐츠 리포지토리가 포함되어 있습니다. RHEL 스캔에는 Red Hat에서 제공하는 OVAL CVE 콘텐츠를 사용하는 것이 좋습니다.*

- *Red Hat Satellite 설명서 - 이 가이드 세트는 OpenSCAP을 사용하여 여러 시스템에서 시스템 보안을 유지하는 방법을 설명합니다.*

9장. 연방 표준 및 규정

보안 수준을 유지하기 위해서는 조직이 연방 및 산업 보안 사양, 표준 및 규정을 준수하기 위해 노력합니다. 이 장에서는 이러한 규칙 및 규정 중 일부를 설명합니다.

9.1. 연방 정보 처리 표준 (FIPS)

연방 정보 처리 표준(FIPS) (Federal Information Processing Standard) (FIPS)Publication집합은 미국이 개발한 컴퓨터 보안 표준입니다. 정부 및 업계 실무 그룹이 암호화 모듈의 품질을 검증합니다. NIST Computer Security Resource Center 에서 공식 FIPS 발행물을 참조하십시오.

FIPS 140-2 표준을 사용하면 암호화 도구가 알고리즘을 적절하게 구현할 수 있습니다. 이러한 수준 및 FIPS 표준의 기타 사양에 대한 자세한 내용은 http://dx.doi.org/10.6028/NIST.FIPS.140-2 에서 전체 FIPS 8.2 표준을 참조하십시오.

규정 준수 요구 사항에 대한 자세한 내용은 Red Hat Government Standards 페이지를 참조하십시오.

9.1.1. FIPS 모드 활성화

Red Hat Enterprise Linux가 연방 정보 처리 표준(FIPS)의 표준(Federal Information Processing Standard) publication를 준수하도록 하려면, certified 암호화 모듈을 사용하기 위해 여러 가지 변경을 수행해야 합니다. 시스템 설치 중에 또는 그 이후에 FIPS 모드를 활성화할 수 있습니다.

시스템 설치 중

엄격한 FIPS 140-2 규정 준수 를 준수하려면 시스템 설치 중에 fips=1 커널 옵션을 커널 명령줄에 추가하십시오. 이 옵션을 사용하면 모든 키 생성은 FIPS 승인 알고리즘과 지속적인 모니터링 테스트를 통해 수행됩니다. 설치 후 시스템은 자동으로 FIPS 모드로 부팅되도록 구성됩니다.

중요

마우스를 이동하거나 많은 키 입력을 눌러 설치 프로세스 중에 시스템에 엔트로피가 충분한지 확인합니다. 권장되는 키 입력은 256개 이상입니다. 256 키 입력보다 작으면 고유하지 않은 키를 생성할 수 있습니다.

시스템 설치 후

설치 후 시스템의 커널 공간 및 사용자 공간을 FIPS 모드로 전환하려면 다음 단계를 따르십시오.

1. ***dracut-fips** 패키지를 설치합니다.*

```
~]# yum install dracut-fips
```

***AES(AES New Instructions)**가 지원하는 **CPU**의 경우 **dracut-fips-aesni** 패키지를 설치하십시오.*

```
~]# yum install dracut-fips-aesni
```

2. ***initramfs** 파일을 다시 생성합니다.*

```
~]# dracut -v -f
```

*모듈 내 무결성 확인을 활성화하고 커널 부팅 중에 필요한 모든 모듈을 제공하려면 **initramfs** 파일을 다시 생성해야 합니다.*

주의

*이 작업은 기존 **initramfs** 파일을 덮어씁니다.*

3. *부트 로더 구성을 수정합니다.*

***FIPS** 모드로 부팅하려면 부트 로더의 커널 명령줄에 **fips=1** 옵션을 추가합니다. **/boot** 또는 **/boot/EFI/** 파티션이 별도의 파티션에 있는 경우 **boot= <partition >**(여기서 **< partition >**은 **/boot**) 매개 변수를 커널 명령 줄에 추가하십시오.*

부팅 파티션을 확인하려면 다음 명령을 입력합니다.

```
~]$ df /boot
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/sda1               495844     53780    416464  12% /boot
```

부팅 시 장치 이름 지정이 변경되어도 **boot=** 구성 옵션이 작동하는지 확인하려면 다음 명령을 실행하여 파티션의 범용 고유 식별자(**UUID**)를 식별합니다.

```
~]$ blkid /dev/sda1
/dev/sda1: UUID="05c000f1-f899-467b-a4d9-d5ca4424c797" TYPE="ext4"
```

커널 명령줄에 **UUID**를 추가합니다.

```
boot=UUID=05c000f1-f899-467b-a4d9-d5ca4424c797
```

부트 로더에 따라 다음과 같이 변경합니다.

- **GRUB 2**

  /etc/default /grub 파일의 **GRUB_CMDLINE_LINUX** 키에 **fips=1** 및 **boot=<partition of /boot>** 옵션을 추가합니다. 변경 사항을 **/etc/default/grub** 에 적용하려면 다음과 같이 **grub.cfg** 파일을 다시 빌드합니다.

  - **BIOS** 기반 시스템에서 **root** 로 다음 명령을 입력합니다.

    ```
    ~]# grub2-mkconfig -o /etc/grub2.cfg
    ```

  - **UEFI** 기반 시스템에서 **root** 로 다음 명령을 입력합니다.

    ```
    ~]# grub2-mkconfig -o /etc/grub2-efi.cfg
    ```

- **zipl (IB z Systems** 아키텍처에서만)

  fips=1 및 **boot=<partition of /boot >** 옵션을 커널 명령행에 **/etc/zipl.conf** 에 추가하고 다음을 입력하여 변경 사항을 적용합니다.

  ```
  ~]# zipl
  ```

4. 사전 연결이 비활성화되어 있는지 확인합니다.

모듈 내 무결성 확인이 제대로 작동하려면 라이브러리 및 바이너리를 사전 링크해야 합니다. 사전 연결은 기본적으로 설치되지 않은 **prelink** 패키지로 수행됩니다. 사전 링크가 설치되지 않은 경우 이 단계는 필요하지 않습니다. 사전 링크를 비활성화하려면 **/etc/sysconfig/prelink** 설정 파일에서 **PRELINKING=no** 옵션을 설정합니다. 모든 시스템 파일에서 기존 사전 연결을 비활성화하려면 **prelink -u -a** 명령을 사용합니다.

5. 시스템을 재부팅합니다.

컨테이너에서 **FIPS** 모드 활성화

호스트가 **FIPS140-2** 모드로 설정되어 있고 다음 요구 사항 중 하나가 충족되면 컨테이너는 **FIPS140-2** 모드로 전환할 수 있습니다.

- **dracut-fips** 패키지는 컨테이너에 설치됩니다.
- **/etc/system-fips** 파일은 호스트의 컨테이너에 마운트됩니다.

9.2. 국제 산업 보안 프로그램 운영 설명서 (NISPOM)

NISPOM (DoD 5220.22-M이라고도 함)은 국가 산업 보안 프로그램 **(NISP)**의 구성 요소로, 분류된 정보와 관련하여 모든 정부 계약자에 대한 일련의 절차 및 요구 사항을 설정합니다. 현재 **NISPOM**은 **2006**년 **2**월 **28**일로, **2013**년 **3**월 **28**일부터 주요 변경 사항이 통합되어 있습니다. **NISPOM** 문서는 다음 **URL**에서 다운로드할 수 있습니다. **http://www.nispom.org/NISPOM-download.html**.

9.3. 결제 카드 산업 데이터 보안 표준(PAYMENT CARD INDUSTRY DATA SECURITY STANDARD)

https://www.pcisecuritystandards.org/about/index.shtml 에서: **PCI** 보안 표준 위원회는 **2006**년에 출시되었으며 **Data Security Standard (DSS)**를 포함한 **PCI** 보안 표준에 대한 개발, 관리, 교육 및 인지도를 담당합니다.

https://www.pcisecuritystandards.org/security_standards/pci_dss.shtml 에서 **PCI DSS** 표준을 다운로드할 수 있습니다.

9.4. 보안 기술 구현 가이드

STIG(Security Technical Implementation Guide)는 컴퓨터 소프트웨어 및 하드웨어의 표준화된 보안 설치 및 유지 관리를 위한 방법론입니다.

STIG https://public.cyber.mil/stigs/ 에 대한 자세한 내용은 다음 **URL**을 참조하십시오.

부록 A. 암호화 표준

A.1. 동기 암호화

A.1.1. 고급 암호화 표준 - AES

암호화에서 AES(Advanced Encryption Standard)는 미국이 채택한 암호화 표준입니다. 정부. 이 표준은 AES-128, AES-192 및 AES-256의 세 가지 블록 암호로, 원래 Rijndael로 게시된 대규모 컬렉션에서 채택되었습니다. 각 AES 암호에는 각각 키 크기가 128, 192 및 256비트인 128비트 블록 크기가 있습니다. AES 암호는 광범위하게 분석되었으며 현재 데이터 암호화 표준(DES)과 마찬가지로 전 세계적으로 사용됩니다.[3]

A.1.1.1. AES 기록

AES는 미국 U.S.NIST(National Standards and Technology)에서 발표했습니다. 2001년 11월 26일 FIPS PUB 197 (FIPS 197) Rijndael을 가장 적합하게 선택하기 전에 15 개의 경쟁 제품이 공개되고 평가되었습니다. 2002년 5월 26일 표준으로 시행되었습니다. 다양한 암호화 패키지에서 사용할 수 있습니다. AES는 NSA가 최상위 비밀 정보에 대해 승인한 첫 번째 공개 및 개방형 암호입니다(AES의 Wikipedia 문서의 보안 섹션 참조).[4]

Rijndael 암호화는 두 개의 Belgian cryptographers, Joan Daemen 및 Vincent Rijmen에 의해 개발되었으며 AES 선택 프로세스에 의해 제출되었습니다. Rijndael은 두 가지 침입자 이름의 portmanteau입니다.[5]

A.1.2. 데이터 암호화 표준 - DES

데이터 암호화 표준(DES)은 1976년 미국 연방 정보 처리 표준(FIPS)으로 표준의 National Bureau가 선택한 블록 암호(공유 비밀 암호화 형태)입니다. 이는 56비트 키를 사용하는 대칭 키 알고리즘을 기반으로 합니다. 이 알고리즘은 처음에 분류된 설계 요소, 비교적 짧은 키 길이, 그리고 NSA(National Security Agency) 백도어에 대한 일시 중단과 관련이 있었습니다. 결과적으로 높은 수준의 학문학 조사가 제공되어 블록 암호 및 암호 해독에 대한 최신 이해를 유도했습니다.[6]

A.1.2.1. DES History

이제 많은 애플리케이션에 대해 DES가 안전하지 않은 것으로 간주됩니다. 이는 56비트 키 크기가 너무 작기 때문에 매우 중요합니다. 1999년 1월, distributed.net 및ctronicdemoier Foundation은 22시간 15분 이내에 DES 키를 공개적으로 중단하기 위해 협력했습니다. 또한 실제로 마운트할 수 없지만 암호에 대한 이론적인 약점을 보여주는 몇 가지 분석 결과가 있습니다. 이 알고리즘은 트립플 DES 형태로 실질적으로 안전한 것으로 알려져 있지만, 이론적인 공격이 있습니다. 최근 몇 년 동안 암호가 AES(Advanced Encryption Standard)로 대체되었습니다.[7]

일부 문서에서는 **DES**를 표준으로 구분하고 **DEA(**데이터 암호화 알고리즘**)**라는 알고리즘을 구분합니다.**[8]**

A.2. 공개 키 암호화

공개 키 암호화는 많은 암호화 알고리즘 및 암호화 시스템에서 사용되는 암호화 방식이며, 특성을 구분하는 것은 대칭 키 알고리즘 대신 대칭 키 알고리즘을 사용하는 것입니다.**Public-key encryption is a cryptographic approach, used by many cryptographic algorithms and cryptosystems, whose distinguishing characteristics is the use of symmetric key algorithms instead of symmetric key algorithms.** 공개 키-개인 키 암호화의 기술을 사용하여 통신을 보호하거나 이전에 알려지지 않은 메시지를 인증하는 많은 방법이 실용적으로되었습니다. 대칭 키 알고리즘을 사용할 때 필요에 따라 하나 이상의 비밀 키를 안전한 초기 교환이 필요하지 않습니다. 또한 디지털 서명을 만드는 데 사용할 수 있습니다.**[9]**

공개 키 암호화는 전 세계의 기본적이고 널리 사용되는 기술이며, 이를 통해 **TLS(Transport Layer Security) (successor to SSL), IRQ** 및 **GPG**와 같은 인터넷 표준을 준수하는 접근 방식입니다.**[10]**

공개 키 암호화에 사용되는 기술을 구별하는 것은 메시지를 암호화하는 데 사용되는 키와 해독하는 데 사용되는 키와 다릅니다.**The distinguishing technique used in public key algorithms, where the key used to encrypt a message is not the same as the key used to decrypt it.** 각 사용자에게는 암호화 키 쌍(공개 키와 개인 키)이 있습니다. 개인 키는 비밀로 유지되지만 공개 키는 널리 분산될 수 있습니다. 메시지는 수신자의 공개 키로 암호화되며 해당 개인 키로만 해독할 수 있습니다. 키가 수학적으로 관련되지만 개인 키는 공개 키에서 파생된 (예: 실제 또는 예상 연습)일 수 없습니다. 이는 **1970**년 중반부터 암호화 관행을 혁신하는 이러한 알고리즘의 발견이었습니다.**[11]**

대조적으로, **Symmetric-key** 알고리즘, 수천 년 동안 사용 된 변형, 전송 및 수신자가 공유하는 단일 비밀 키를 사용합니다 (따라서 암호화 및 암호 해독에 대한 일반적인 용어의 모호도를 고려하여 일반 용어의 모호도를 계산). 대칭 암호화 체계를 사용하려면 발신자와 수신자가 키를 미리 안전하게 공유해야 합니다.**[12]**

대칭 키 알고리즘은 거의 항상 훨씬 덜 컴퓨팅 집약적이므로 키 교환 알고리즘을 사용하여 키를 교환하고 해당 키와 대칭 키 알고리즘을 사용하여 데이터를 전송하는 것이 일반적입니다. 예를 들어, **IRQ** 및 **SSL/TLS** 체계 제품군은 예를 들어 하이브리드 암호화 시스템이라고 하며 그 결과 하이브리드 암호화 시스템이라고 합니다.**[13]**

A.2.1. Diffie-Hellman

Diffie-Hellman 키 교환 **(D-H)**은 서로에 대한 사전 지식이없는 두 당사자가 안전하지 않은 통신 채널을 통해 공유 비밀 키를 공동 설정할 수있는 암호화 프로토콜입니다. 그런 다음 이 키를 사용하여 대칭 키 암호를 사용하여 후속 통신을 암호화할 수 있습니다.**[14]**

A.2.1.1. Diffie-Hellman 기록

이 체계는 1976년 Whitfield Diffie 및 Martin Hellman에 의해 처음 발표되었지만 나중에 Malcolm J.ECDHEson에 의해 GCHQ에서 몇 년 전에 별도로 발명 된 것으로 나타났습니다. 2002년에 Hellman은 공통 키 암호화 (Hellman, 2002)의 Ralph Merkle의 기여에 대한 인식에 알고리즘을 Diffie-Hellman-Merkle 키 교환이라고 제안했습니다.[15]

Diffie-Hellman 키 계약 자체는 익명(인증되지 않은) 키 집계 프로토콜이지만 다양한 인증된 프로토콜을 위한 기반을 제공하며, 암호화 제품군에 따라 EDH 또는 DHE로 참조되는 전송 계층 보안 모드에서 완벽한 전달 보안을 제공하는 데 사용됩니다.[16]

U.S. 이제 특허 4,200,770이 만료되었으며 알고리즘 및 Hellman, Diffie 및 Merkle을 발명자처럼 설명합니다.[17]

A.2.2. RSA

암호화에서 RSA (Rivest, Shamir 및 Adleman)은 공개 키 암호화 알고리즘입니다. 서명과 암호화에 적합한 것으로 알려진 첫 번째 알고리즘이며 공개 키 암호화의 첫 번째 훌륭한 발전 중 하나였습니다. RSA는 전자상거래 프로토콜에서 널리 사용되며, 충분히 긴 키와 최신 구현의 사용을 보장받을 수 있다고 믿고 있습니다.

A.2.3. DSA

DSA (Digital Signature Algorithm)는 디지털 서명을 위한 미국 연방 정부 표준인 디지털 서명을 위한 표준입니다. DSA는 서명에만 사용되며 암호화 알고리즘이 아닙니다. [18]

A.2.4. SSL/TLS

TLS(Transport Layer Security) 및 이전 버전인 SSL(Secure Sockets Layer)은 인터넷과 같은 네트워크를 통한 통신을 위한 보안을 제공하는 암호화 프로토콜입니다. TLS 및 SSL은 전송 계층 엔드 투 엔드에서 네트워크 연결 세그먼트를 암호화합니다.

여러 버전의 프로토콜은 웹 검색, 전자 메일, 인터넷 FAFA, 인스턴트 메시징 및 음성-오IP(VoIP)와 같은 애플리케이션에서 널리 사용됩니다.[19]

A.2.5. Cramer-Shoup Cryptosystem

Cramer-Shoup 시스템은 symmetric 키 암호화 알고리즘이며, 표준 암호화 가정을 사용하여 적응형

암호 텍스트 공격에 대해 안전한 것으로 입증된 첫 번째 효율적인 체계였습니다. 그 보안은 의사 결정 Diffie-Hellman 가정에 대한 계산적 침입(전체적으로 가정하지만 증명되지 않은)을 기반으로 합니다. 1998년 Ronald Cramer 및 Victor Shoup에서 개발한 것은 ElGamal cryptosystem의 확장입니다. 매우 부전할 수 있는 ElGamal과 달리 Cramer-Shoup은 리소스의 공격자에 대해 비중앙성을 보장하기 위해 추가 요소를 추가합니다. 이러한 비마지능은 충돌 방지 해시 함수 및 추가 계산을 사용하여 달성되므로, 암호 텍스트가 ElGamal에서처럼 두 배나 커집니다.[20]

A.2.6. ElGamal Encryption

암호화에서 ElGamal 암호화 시스템은 Diffie-Hellman 키 계약을 기반으로 하는 공개 키 암호화 암호화에 대한 대칭 키 암호화 알고리즘입니다. 1985 Taher ElGamal에 의해 설명되었습니다. ElGamal 암호화는 무료 GNU 개인 정보 보호 가드 소프트웨어, 최신 버전인 pgp 및 기타 cryptosystems에서 사용됩니다.[21]

---

[3] "고급 암호화 표준". Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Advanced_Encryption_Standard

[4] "고급 암호화 표준". Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Advanced_Encryption_Standard

[5] "고급 암호화 표준". Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Advanced_Encryption_Standard

[6] "데이터 암호화 표준". Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Data_Encryption_Standard

[7] "데이터 암호화 표준". Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Data_Encryption_Standard

[8] "데이터 암호화 표준". Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Data_Encryption_Standard

[9] "public-key Encryption". Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Public-key_cryptography

[10]

"public-key Encryption". Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Public-key_cryptography

[11]
"public-key Encryption". Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Public-key_cryptography

[12]
"public-key Encryption". Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Public-key_cryptography

[13]
"public-key Encryption". Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Public-key_cryptography

[14]
"Diffie-Hellman." Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Diffie-Hellman

[15]
"Diffie-Hellman." Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Diffie-Hellman

[16]
"Diffie-Hellman." Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Diffie-Hellman

[17]
"Diffie-Hellman." Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Diffie-Hellman

[18]
"DSA." Wikipedia. 2010년 2월 24일 http://en.wikipedia.org/wiki/Digital_Signature_Algorithm

[19]
"TLS/SSL." Wikipedia. 24 February 2010 http://en.wikipedia.org/wiki/Transport_Layer_Security

[20]
"Cramer-Shoup cryptosystem". Wikipedia. 24 February 2010 http://en.wikipedia.org/wiki/Cramer–Shoup_cryptosystem

[21]
"ElGamal encryption" Wikipedia. 24 February 2010 http://en.wikipedia.org/wiki/ElGamal_encryption

부록 B. 개정 내역

고침 1-43 **Fri Feb 7 2020** **Jan Fiala**
규정 준수 및 취약점 스캔 장의 업데이트가 포함된 비동기 릴리스.

고침 1-42 **Fri Aug 9 2019** **Mirek Jahoda**
7.7 GA 게시에 대한 버전

고침 1-41 **Sat Oct 20 2018** **Mirek Jahoda**
7.6 GA 게시용 버전.

고침 1-32 **Wed Apr 4 2018** **Mirek Jahoda**
7.5 GA용 버전이 공개되었습니다.

고침 1-30 **Thu Jul 27 2017** **Mirek Jahoda**
7.4 GA 게시용 버전.

고침 1-24 **Mon Feb 6 2017** **Mirek Jahoda**
잘못 된 업데이트, 특히 firewalld 섹션에서 async 릴리스.

고침 1-23 **Tue Nov 1 2016** **Mirek Jahoda**
7.3 GA 게시의 버전입니다.

고침 1-19 **Mon Jul 18 2016** **Mirek Jahoda**
스마트 카드 섹션이 추가되었습니다.

고침 1-18 **Mon Jun 27 2016** **Mirek Jahoda**
OpenSCAP-daemon 및 Atomic Scan 섹션이 추가되었습니다.

고침 1-17 **Fri Jun 3 2016** **Mirek Jahoda**
misc. update를 사용한 비동기 릴리스.

고침 1-16 **Tue Jan 5 2016** **Robert Krátký**
7.2 GA 수정

고침 1-15 **Tue Nov 10 2015** **Robert Krátký**
7.2 GA 릴리스 버전

고침 1-14.18 **Mon Nov 09 2015** **Robert Krátký**
misc. update를 사용한 비동기 릴리스.

고침 1-14.17 **Wed Feb 18 2015** **Robert Krátký**
7.1 GA 릴리스 버전

고침 1-14.15 **Fri Dec 06 2014** **Robert Krátký**
Red Hat 고객 포털의 순서를 정렬하도록 업데이트되었습니다.

고침 1-14.13 **Thu Nov 27 2014** **Robert Krátký**
POODLE vuln을 반영하는 업데이트입니다.

고침 1-14.12 **Tue Jun 03 2014** **Tomáš Čapek**
7.0 GA 릴리스 버전.