??

외부의 위협으로부터 리눅스 서버를 지키기 위한 책

- SSH
    - 포트 변경
    - root 계정 접속 차단
- 방화벽
    - 리눅스에서 지원하는 방화벽
    - firewalld

SSH

SSH(Secure Shell)은 원격지의 셸에 접속하기 위해 사용되는 네트워크 프로토콜로, 서버의 모든 것을 서버실 내에서 모니터와 키보드를 달아서 오프라인으로 통제할 것이 아니라면 좋든 싫든 SSH를 사용할 수밖에 없다. SSH는 서버를 원격으로 통제할 수 있는 강력한 도구인 만큼, 해커들은 해당 서버의 통제권을 얻기 위해 SSH부터 공격을 시도하는 것이 일반적이다. 따라서 SSH 보안은 리눅스 보안의 가장 기본적이며 핵심적인 부분이라고 할 수 있다.

SSH

?? ??

> **ssh**의 기본 포트가 **22**라는 것은 이미 널리 알려진 사실이다. ftp의 21, http/https가 각각 80/443인 것과 같이 특정한 쓰임새를 위해 IANA (Internet Assigned Numbers Authority)에서 할당한 TCP 및 UDP 포트들을 잘 알려진 포트 **(Well-known port)** 라고 부른다. 잘 알려진 포트는 특정 서비스에 접근하기 위해 포트를 알 필요가 없다는 장점이 있지만, 해커들이 취약점을 찾기위해 가장 먼저 공격하는 포트이기도 하다. 따라서, ssh의 기본 포트를 변경하는 것 만으로도 해커들의 공격 속도를 지연시키는데 도움이 될 수 있다.

SSH ???? ?? ??

```
# vi /etc/ssh/sshd_config
```

SSH ?? ??

sshd_config 파일에서 **Port** 항목의 값을 원하는 값으로 변경한다. 최초에는 해당 항목이 주석처리 되어있을 수 있으므로, 주석처리되어 있다면 주석을 해제하고 값을 변경한다. 아래의 내용은 **ssh**의 포트를 **10022**로 설정하기 위한 예제다.

```
Port 10022
```

SSH ???

Debian ??

```
# systemctl restart ssh
```

RHEL ??

```
# systemctl restart sshd
```

SSH

root ?? ?? ??

> **!!! root** 계정 접속을 차단하기에 앞서, 해당 서버에 접속할 수 있는 별도의 수단을 미리 마련해 두는것을 권장한다

SSH ???? ?? ??

```
# vi /etc/ssh/sshd_config
```

PermitRootLogin ? ??

sshd_config 파일에서 **PermitRootLogin** 항목의 값을 **no**로 변경한다.
최초에는 해당 항목이 주석처리 되어있을 수 있으므로, 주석처리되어 있다면 주석을 해제하고 값을 변경한다.

```
PermitRootLogin no
```

SSH ???

Debian ??

```
# systemctl restart ssh
```

RHEL ??

```
# systemctl restart sshd
```

???

(Firewall)

방화벽

????? ???? ???

1. ??? ?? ??? ????? ??

방화벽 소프트웨어	주요 특징
iptables	커널 레벨에서 작동하며 개별 패킷을 직접 필터링하는 강력한 방화벽 도구.
nftables	iptables의 후속 기술로 개발되어 향상된 성능을 제공하며, 더 적은 코드로 유연한 관리 가능.
firewalld	iptables/nftables를 백엔드로 활용하며, Zone 기반의 관리 방식을 제공. 대부분의 RHEL 계열의 운영체제에서 기본적으로 지원. **(RHEL 7 이상)**
ufw	iptables를 백엔드로 활용하며, 초보자 친화적인 명령어 기반 인터페이스를 제공. 대부분의 우분투 계열의 운영체제에서 기본적으로 지원.

2. ??? ??

iptables

장점:

- 강력한 패킷 필터링 기능 제공
- 대부분의 리눅스 배포판에서 지원
- 커스텀 스크립트를 활용한 정밀한 제어 가능

단점:

- 설정이 복잡하고 학습 곡선이 가파름
- 동적 변경이 어렵고, 설정 변경 시 전체 규칙을 다시 로드해야 함

nftables

장점:

- iptables보다 성능이 개선되고 더 적은 코드로 관리 가능
- 하나의 프레임워크에서 IPv4, IPv6, ARP, 브리지 필터링 지원
- 동적 규칙 업데이트가 가능

단점 :

- 사용법에 대한 자료가 iptables에 비해 상대적으로 적음

firewalld

장점 :

- iptables나 nftables에 비해 편리한 설정 가능
- 존 (Zone) 개념을 활용하여 트래픽을 용도에 따라 유연하게 관리 가능
- 동적 규칙 업데이트가 가능

단점 :

- iptables나 nftables 대비 낮은 커스터마이징 자유도
- 성능이 중요한 환경에서는 문제가 발생할 수 있음

ufw

장점 :

- 사용이 매우 간편하여 초보자도 쉽게 접근 가능
- 직관적인 명령어 사용 (`ufw allow 22/tcp` 등)
- Ubuntu 등에서 기본 제공되므로 추가 설치가 필요 없음

단점 :

- 세밀한 설정이 어렵고 , 복잡한 방화벽 규칙 적용이 어려움
- iptables와 연동되므로 직접적인 nftables 지원 부족

방화벽

firewalld

1. firewalld??

firewalld는 방화벽(firewall) + 데몬(deamon)의 합성어로, 주로 RHEL[1] 기반의 운영체제에서 기본적으로 채용하고 있는 방화벽 소프트웨어다. firewalld의 주요 특징은 다음과 같다.

- 동적 방화벽 관리: 서비스 중단 없이 규칙을 변경 가능
- 존 **(Zone)** 개념 지원: 네트워크 인터페이스별로 다른 보안 수준을 적용 가능
- 서비스 기반 규칙 설정: 특정 포트가 아닌 서비스 단위로 방화벽 규칙 적용 가능
- **D-Bus API** 지원: GUI 및 명령줄 도구를 통해 쉽게 관리 가능

동적 방화벽 관리 및 유연한 보안 수준 적용이 가능하며 iptables에 비해 설정이 쉽다보니 다양한 리눅스 배포판에서 지원하고 있다. 특히 서버로 많이 사용되는 RHEL 기반의 운영체제를 사용한다면 firewalld의 사용법을 숙지해 두는 것이 좋다.

---

2. Zone(??)? ?????

firewalld에서 Zone(이하 영역)은 네트워크 인터페이스와 연결된 신뢰 수준을 정의하는 역할을 한다. 각 영역은 특정한 보안 정책을 가지며, 네트워크 인터페이스에 맞춰 적절한 영역을 설정할 수 있다. 예를들어 외부에 공개된 네트워크의 경우에는 최소한의 접근만 허용하거나 아웃바운드 트래픽만 허용하는 영역으로 설정한다거나, 신뢰할 수 있는 내부 네트워크를 사용하는 경우에는 모든 트래픽을 허용하는 영역으로 설정하는 방식으로 여러 개의 네트워크 인터페이스에 서로 다른 방화벽 규칙을 간편하게 적용할 수 있다. firewalld에서 기본적으로 생성되어 있는 영역의 종류는 다음과 같다.

영역 명	네트워크 위치	설명
drop	외부	모든 인바운드 트래픽을 차단하고 아웃바운드 트래픽만 허용.
block	외부	모든 인바운드 트래픽을 차단하고 거부 메시지를 반환함. 단, 시스템 내에서 시작된 네트워크 연결은 모두 허용.
public	외부	최소한의 접근만 허용. (기본 영역)
external	외부	공유기와 같이 라우터의 마스커레이딩이 활성화 된 외부 네트워크에서 사용. 최소한의 접근만 허용.

영역 명	네트워크 위치	설명
dmz	외부	DMZ 영역에서 사용하는 영역. 내부 네트워크에 대한 최소한의 접근만 허용.
work	내부	업무용 네트워크 영역. 최소한의 접근만 허용.
home	내부	가정용 네트워크 영역. 최소한의 접근만 허용.
internal	내부	내부 네트워크 영역. 최소한의 접근만 허용.
trusted	내부	모든 접근 허용.

위의 영역은 자주 쓰이는 영역을 프리셋화 한 것이고, 필요에 따라 새로운 영역을 생성할 수도 있다. 또한, 어플리케이션이 설치되는 과정에서 자동으로 영역이 생성되는 경우도 있다. (EX. 도커)

---

3. firewalld ???

3-1. ?? ???

firewalld ?? ??

현재 firewalld가 동작중인지 확인합니다:

```
# firewall-cmd --state
```

??? ?? ?? ?? ? ??

변경된 규칙이 있을 경우 즉시 저장하고 적용합니다:

```
# firewall-cmd --reload
```

3-2. ?? ?? ???

?? ??? ?? ??

현재 firewalld에서 사용 가능한 모든 영역을 확인합니다:

```
# firewall-cmd --get-zones
```

?? ???? ?? ??

모든 네트워크 인터페이스가 어떤 영역에 존재하는지 확인합니다(사용되지 않는 영역은 표시되지 않음):

```
# firewall-cmd --get-active-zones
```

?? ?? ??

지정된 기본 영역을 확인합니다:

```
# firewall-cmd --get-default-zone
```

?? ??? ?? ??

지정한 영역의 상세 설정을 확인합니다 (지정된 영역이 없는 경우 기본 영역의 상세 설정이 표시됨):

```
# firewall-cmd --zone=public --list-all
```

?? ?????? ?? ?? ??

지정한 인터페이스의 영역을 조회합니다.

```
# firewall-cmd --get-zone-of-interface=eth0
```

?? ?? ??

지정한 영역을 기본 영역으로 변경합니다.

```
# firewall-cmd --set-default-zone=internal
```

?? ?????? ?? ??

지정한 인터페이스의 영역을 지정한 영역으로 변경합니다. `--permanent` 옵션을 사용하면 영구적으로 변경합니다.

```
# firewall-cmd --zone=internal --change-interface=eth1 --permanent
```

3-3. ?? ?? ??/?? ???

?? ??

`--zone=` 옵션에 선언된 영역에 지정된 프로토콜의 포트를 엽니다. `--zone=` 옵션을 지정하지 않을 경우, `default-zone` 영역에 포트를 엽니다. `--permanent` 옵션을 사용하면 영구적으로 변경합니다.

```
# firewall-cmd --add-port=80/tcp --permanent
# firewall-cmd --zone=public --add-port=443/tcp --permanent
# firewall-cmd --zone=public --add-port=443/udp --permanent
```

?? ??

`--zone=` 옵션에 선언된 영역에 지정된 프로토콜의 포트를 차단합니다. `--zone=` 옵션을 지정하지 않을 경우, `default-zone` 영역에 포트를 차단합니다. `--permanent` 옵션을 사용하면 영구적으로 변경합니다.

```
# firewall-cmd --remove-port=80/tcp --permanent
# firewall-cmd --zone=public --remove-port=443/tcp --permanent
# firewall-cmd --zone=public --remove-port=443/udp --permanent
```

3-4. ??? ?? ??/?? ???

??? ??

`--zone=` 옵션에 선언된 영역에 지정된 서비스를 엽니다. `--zone=` 옵션을 지정하지 않을 경우, `default-zone` 영역에 서비스를 엽니다. `--permanent` 옵션을 사용하면 영구적으로 변경합니다.

```
# firewall-cmd --add-service=http --permanent
# firewall-cmd --zone=public --add-service=https --permanent
```

??? ??

`--zone=` 옵션에 선언된 영역에 지정된 서비스를 차단합니다. `--zone=` 옵션을 지정하지 않을 경우, `default-zone` 영역의 서비스를 차단합니다. `--permanent` 옵션을 사용하면 영구적으로 변경합니다.

```
# firewall-cmd --remove-service=http --permanent
# firewall-cmd --zone=public --remove-service=https --permanent
```

---

1. RHEL 7 이상만 해당한다.