1. Firewall Rule Implementation

A **firewall** is used to monitor and control incoming and outgoing network traffic based on predetermined security rules. We will configure a basic **firewall rule** to restrict access to a web server.

Objective: Block inbound traffic on port 22 (SSH) from an external IP address while allowing it from trusted IPs.

Steps to Implement:

1. **Firewall Type**: For this example, let's assume we are using **UFW (Uncomplicated Firewall)** on a Linux server.
2. **Firewall Configuration**:
    - Allow SSH access from trusted IP `192.168.1.100`.
    - Deny SSH access from all other external IP addresses.

Commands:
bash
Copy

```
# Default deny incoming traffic
sudo ufw default deny incoming

# Allow outgoing traffic
sudo ufw default allow outgoing

# Allow SSH from trusted IP
sudo ufw allow from 192.168.1.100 to any port 22

# Deny SSH from all other IPs (implicit, since all other access is
denied by default)
sudo ufw deny 22/tcp

# Enable UFW
sudo ufw enable
```

3.

Firewall Rule Breakdown:

- **Default deny incoming**: Blocks all incoming traffic unless explicitly allowed.

- **Allow from 192.168.1.100 to port 22**: Specifically allows the trusted IP `192.168.1.100` to access SSH.
- **Deny port 22/tcp**: Denies SSH access to all external sources by default.

Example of Detected Event:

- **Event**: A connection attempt from an unauthorized IP `203.0.113.10` to port 22 is blocked.

Detected Log Entry (found in `ufw.log`):
less
Copy
```
Jan 29 14:32:01 server ufw[1234]: [DENIED] IN=eth0 OUT=
MAC=00:1a:2b:3c:4d:5e:6f:7g:8h:9i:10j:11k:12l: SRC=203.0.113.10
DST=192.168.1.200 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=12345 DF
PROTO=TCP SPT=56789 DPT=22 WINDOW=29200 RES=0x00 SYN URGP=0
```

-

---

2. IDS Configuration (Intrusion Detection System)

An **IDS** monitors network traffic and alerts when it detects suspicious activity. We will configure **Snort** as our IDS.

Objective: Configure Snort to detect and log an SSH brute-force attempt.

Steps to Implement:
Install Snort: On a Linux server, install Snort:
bash
Copy
```
sudo apt update
sudo apt install snort
```

1.

Configure Snort: Edit the `snort.conf` configuration file to specify the home network and rules path.
Example configuration (add to `snort.conf`):
bash
Copy
```
ipvar HOME_NET [192.168.1.0/24]
var EXTERNAL_NET any
```

```
include $RULE_PATH/local.rules
```

2.

Create a Local Rule to Detect SSH Brute-Force Attacks: Add a rule in `local.rules` to detect multiple failed SSH login attempts (e.g., more than 5 attempts within 10 minutes from a single IP).
Example Rule:
bash
Copy

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 22 (msg:"SSH Brute Force Attack"; flow:to_server,established; content:"Failed password"; threshold:type threshold, track by_src, count 5, seconds 600; sid:1000001; )
```

3. This rule triggers an alert when there are 5 failed SSH login attempts from the same source IP within 10 minutes.

Start Snort:
bash
Copy

```
sudo snort -A console -c /etc/snort/snort.conf -i eth0
```

4.

Example of Detected Event:

- **Event**: Snort detects multiple failed login attempts within a short time frame, indicating a brute-force attack.

Detected Log Entry:
css
Copy

```
[**] [1:1000001:0] SSH Brute Force Attack [**]
[Priority: 2] {TCP} 192.168.1.200:22 -> 203.0.113.10:23456
```

-

---

3. IPS Configuration (Intrusion Prevention System)

An **IPS** not only detects but also prevents attacks by actively blocking malicious traffic. We will configure **Suricata** as our IPS to block SSH brute-force attempts.

Objective: Configure Suricata to block an SSH brute-force attempt in real-time.

Steps to Implement:

Install Suricata: On a Linux server, install Suricata:

bash

Copy

```
sudo apt update
sudo apt install suricata
```

1.

Configure Suricata: In the `suricata.yaml` configuration file, ensure that IPS mode is enabled and set up network interfaces.

yaml

Copy

```
af-packet:
  - interface: eth0
    threads: 4
    cluster-type: cluster_flow
```

2.

Create a Rule to Block SSH Brute-Force Attempts: Suricata uses **rule sets** similar to Snort. Add the following rule to block SSH brute-force attacks:

bash

Copy

```
drop tcp $EXTERNAL_NET any -> $HOME_NET 22 (msg:"SSH Brute Force
Attack"; flow:to_server,established; content:"Failed password";
threshold:type threshold, track by_src, count 5, seconds 600;
sid:1000002; )
```

3. This rule will drop the connection if there are 5 failed SSH login attempts from the same IP within 10 minutes.

Start Suricata:

bash

Copy

```
sudo suricata -c /etc/suricata/suricata.yaml -i eth0 --runmode=ips
```

4.

Example of Detected Event:

- **Event**: Suricata blocks an SSH brute-force attack attempt in real-time.

Detected Log Entry (in `eve.json` output from Suricata):

json

Copy

```
{
  "timestamp": "2025-01-29T14:50:00.123456",
  "flow_id": 123456789,
  "in_iface": "eth0",
  "src_ip": "203.0.113.10",
  "src_port": 56789,
  "dest_ip": "192.168.1.200",
  "dest_port": 22,
  "proto": "TCP",
  "event_type": "blocked",
  "message": "SSH Brute Force Attack"
}
```