Date of submission: 23/04/2025
Batch: C1 Roll No.: 16010122323
Div: C
Student Name: Vedansh Savla
Experiment No: 7
Staff In-charge: SHIVANI DEOSTHALE

TITLE: Illustrate and Compare network security mechanisms
AIM: Implementation and configuration of Firewall using Iptable. Demo of Palo Alto Next Gen Firewall
OUTCOME: Student will be able to
CO4: Illustrate and Compare network security mechanisms

Theory:

1. Firewall and Its Role

A firewall is a security system that monitors and controls incoming and outgoing network traffic based on security rules. Its primary role is to prevent unauthorized access while allowing legitimate communication, protecting networks from external threats.

2. Types of Firewalls

- **Packet-Filtering Firewalls:** Block traffic based on IP addresses, ports, and protocols.
- **Stateful Inspection Firewalls:** Track the state of active connections for more context-aware filtering.
- **Proxy Firewalls:** Act as intermediaries, filtering traffic at the application level.
- **Next-Generation Firewalls (NGFW):** Include advanced features like deep packet inspection, intrusion prevention, and application awareness.
- **Unified Threat Management (UTM):** Combine multiple security functions (firewall, antivirus, etc.) in one device.

- **Cloud Firewalls:** Firewalls hosted in the cloud for scalable protection.

3. Next-Generation Firewall (NGFW) – Palo Alto

Palo Alto's Next-Generation Firewall (NGFW) offers:

- **Application Awareness:** Identifies and controls apps running on the network.
- **Threat Prevention:** Blocks known threats like malware and viruses.
- **Deep Packet Inspection (DPI):** Examines traffic in detail for hidden threats.
- **User Identification:** Controls access based on users, not just IPs.
- **SSL Decryption:** Inspects encrypted traffic.
- **Centralized Management:** Simplifies configuration and monitoring across multiple devices.

Palo Alto's NGFW provides advanced security features for modern networks.

Algorithm:

- **Define the Ruleset:**
  - Define rules for allowed and blocked IPs, ports, and protocols.
  - Establish logging and alert mechanisms.
- **Monitor Traffic:**
  - Monitor all incoming and outgoing traffic based on defined rules.
- **Evaluate Packet State (Stateful Firewall):**
  - Evaluate the state of the packet (new, established, or related).
  - Stateful firewalls track the state of active connections and make decisions based on the connection context.
- **Block or Allow Traffic:**
  - If the packet matches an allowed rule, forward it.
  - If it matches a block rule or doesn't conform to any rules, drop the packet.
- **Logging and Reporting:**
  - Keep a log of traffic and alerts for security analysis.

Program Implementation and Output(s):

```
┌──(root㉿kali)-[~]
└─# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

┌──(root㉿kali)-[~]
└─#
```

```
┌──(root㉿kali)-[~]
└─# apt install iptables-persistent
iptables-persistent is already the newest version (1.0.23).
Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 1555

┌──(root㉿kali)-[~]
└─# netfilter-persistent save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/15-ip4tables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/25-ip6tables save
```

loopback interface

```
┌──(root㉿kali)-[~]
└─# iptables -A INPUT -i lo -j ACCEPT

┌──(root㉿kali)-[~]
└─# iptables -A OUTPUT -o lo -j ACCEPT
```

Allowing incoming and outgoing connections.

```
┌──(root㉿kali)-[~]
└─# iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
```

Allowing Internal Network to access External network.

```
┌──(root㉿kali)-[~]
└─# iptables -A FORWARD -i eth1 -o eth0 -j ACCEPT
```

Dropping invalid packets.

```
┌──(root㉿kali)-[~]
└─# iptables -A INPUT -m conntrack --ctstate INVALID -j DROP
```

```
┌──(root㉿kali)-[~]
└─# ping -c 4 192.168.1.1 -s 65536
PING 192.168.1.1 (192.168.1.1) 65536(65564) bytes of data.

── 192.168.1.1 ping statistics ──
4 packets transmitted, 0 received, 100% packet loss, time 3058ms
```

Blocking IP address.

```
┌──(root㉿kali)-[~]
└─# iptables -A INPUT -s 203.0.113.51 -j DROP

┌──(root㉿kali)-[~]
└─# iptables -A INPUT -s 203.0.113.51 -j REJECT
```

```
┌──(root㉿kali)-[~]
└─# hping3 -S -p 80 203.0.113.51  --spoof 192.168.1.100
HPING 203.0.113.51 (eth0 203.0.113.51): S set, 40 headers + 0 data bytes
^C
── 203.0.113.51 hping statistic ──
118 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

Blocking emails.

```
┌──(root㉿kali)-[~]
└─# # Block outgoing SMTP (Port 25)
iptables -A OUTPUT -p tcp --dport 25 -j REJECT

# Block incoming SMTP (Port 25)
iptables -A INPUT -p tcp --dport 25 -j REJECT

# Block outgoing IMAP (Port 143)
iptables -A OUTPUT -p tcp --dport 143 -j REJECT

# Block incoming IMAP (Port 143)
iptables -A INPUT -p tcp --dport 143 -j REJECT
```

Allowing incoming HTTP and HTTPS requests.

```
┌──(root㉿kali)-[~]
└─# # Allow incoming HTTP (Port 80)
iptables -A INPUT -p tcp --dport 80 -j ACCEPT

# Allow incoming HTTPS (Port 443)
iptables -A INPUT -p tcp --dport 443 -j ACCEPT
```

Blocking Outgoing SMTP Mail.

```
┌──(root㉿kali)-[~]
└─# iptables -A OUTPUT -p tcp --dport 25 -j REJECT
```

Allowing all incoming IMAP, IMAPS, POP3, POP3S.

```
┌──(root㉿kali)-[~]
└─# # Allow incoming IMAP traffic on port 143 (for standard email retrieval)
iptables -A INPUT -p tcp --dport 143 -j ACCEPT

# Allow incoming IMAPS traffic on port 993 (for secure email retrieval using SSL)
iptables -A INPUT -p tcp --dport 993 -j ACCEPT

# Allow incoming POP3 traffic on port 110 (for standard email retrieval)
iptables -A INPUT -p tcp --dport 110 -j ACCEPT

# Allow incoming POP3S traffic on port 995 (for secure email retrieval using SSL)
iptables -A INPUT -p tcp --dport 995 -j ACCEPT
```

Listing, adding, deleting, saving entries.

```
┌──(root㉿kali)-[~]
└─# # List all current iptables rules
iptables -L

# Add a rule to allow incoming SSH traffic on port 22 (standard SSH port)
iptables -A INPUT -p tcp --dport 22 -j ACCEPT

# Delete the rule that allows incoming SSH traffic on port 22
iptables -D INPUT -p tcp --dport 22 -j ACCEPT

# Save the current iptables rules to a file for persistence (usually for reboot)
iptables-save > /etc/iptables/rules.v4

Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
DROP       all  --  anywhere             anywhere             ctstate INVALID
DROP       all  --  203.0.113.51         anywhere
REJECT     all  --  203.0.113.51         anywhere             reject-with icmp-port-unreachable
REJECT     tcp  --  anywhere             anywhere             tcp dpt:smtp reject-with icmp-port-unreachable
REJECT     tcp  --  anywhere             anywhere             tcp dpt:smtp reject-with icmp-port-unreachable
REJECT     tcp  --  anywhere             anywhere             tcp dpt:imap2 reject-with icmp-port-unreachable
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:https
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:imap2
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:imaps
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:pop3
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:pop3s
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:imap2
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:imaps
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:pop3
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:pop3s

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere
REJECT     tcp  --  anywhere             anywhere             tcp dpt:smtp reject-with icmp-port-unreachable
REJECT     tcp  --  anywhere             anywhere             tcp dpt:smtp reject-with icmp-port-unreachable
REJECT     tcp  --  anywhere             anywhere             tcp dpt:imap2 reject-with icmp-port-unreachable
REJECT     tcp  --  anywhere             anywhere             tcp dpt:smtp reject-with icmp-port-unreachable
```

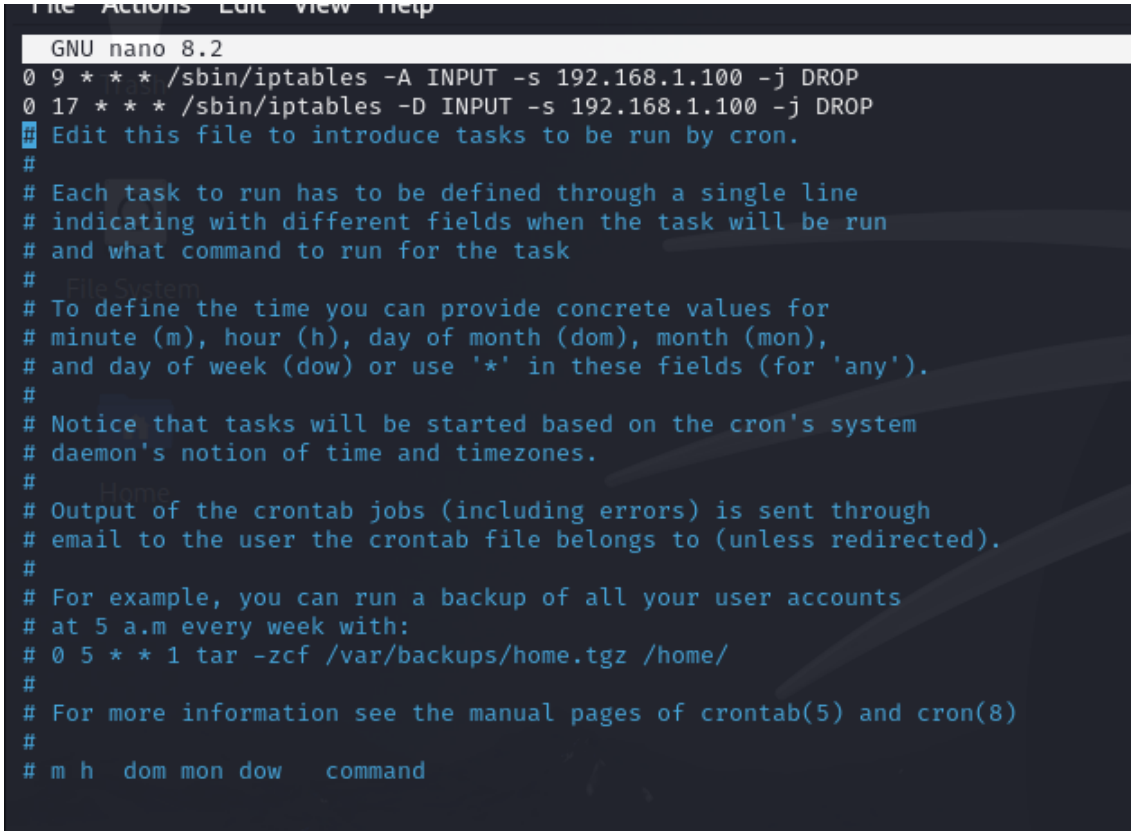
Restricting access to a website for a given time interval.

```
┌──(root㉿kali)-[~]
└─# # Block incoming traffic to a specific IP (e.g., 93.184.216.34) - this is blocking access to a website
iptables -A INPUT -d 93.184.216.34 -j

iptables v1.8.11 (nf_tables): option "-j" requires an argument
Try `iptables -h' or 'iptables --help' for more information.
```

```
┌──(root㉿kali)-[~]
└─# crontab -e
no crontab for root - using an empty one
Select an editor.  To change later, run select-editor again.
  1. /bin/nano        <---- easiest
  2. /usr/bin/vim.basic
  3. /usr/bin/vim.tiny

Choose 1-3 [1]: 1
crontab: installing new crontab
```


GNU nano 8.2
0 9 * * * /sbin/iptables -A INPUT -s 192.168.1.100 -j DROP
0 17 * * * /sbin/iptables -D INPUT -s 192.168.1.100 -j DROP
Edit this file to introduce tasks to be run by cron.
#
Each task to run has to be defined through a single line
indicating with different fields when the task will be run
and what command to run for the task
#
To define the time you can provide concrete values for
minute (m), hour (h), day of month (dom), month (mon),
and day of week (dow) or use '*' in these fields (for 'any').
#
Notice that tasks will be started based on the cron's system
daemon's notion of time and timezones.
#
Output of the crontab jobs (including errors) is sent through
email to the user the crontab file belongs to (unless redirected).
#
For example, you can run a backup of all your user accounts
at 5 a.m every week with:
0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
For more information see the manual pages of crontab(5) and cron(8)
#
m h dom mon dow command

```
┌──(root㉿kali)-[~]
└─# crontab -l
0 9 * * * /sbin/iptables -A INPUT -s 192.168.1.100 -j DROP
0 17 * * * /sbin/iptables -D INPUT -s 192.168.1.100 -j DROP
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
```

Post Lab Questions:

Implementation and configuration of Firewall using Iptable / Fortinet / Palo Alto.

7.1 What is the difference between stateful and stateless firewalls?

Ans:

Aspect	**Stateful Firewall**	**Stateless Firewall**
Packet Inspection	Tracks the state of active connections.	Inspects each packet independently.

Traffic Context	Maintains a table of active connections to allow only valid packets.	No memory of past packets, makes decisions based on fixed rules.
Security Level	More secure, as it can filter traffic based on session information.	Less secure, can be bypassed easily by attackers.
Efficiency	More resource-intensive due to connection tracking.	More efficient, simpler to implement and manage.
Performance	May have slightly reduced performance due to tracking states.	Faster, but less flexible and secure.
Use Case	Ideal for more dynamic, complex networks.	Suitable for simpler, less dynamic environments.

7.2 How does a firewall protect data?

Ans: A firewall protects data by:

1. **Blocking Unauthorized Access**: It controls incoming and outgoing traffic based on predefined security rules, preventing unauthorized users or devices from accessing the network.
2. **Traffic Filtering**: It inspects traffic and blocks malicious or unwanted data packets (e.g., viruses, malware).
3. **Network Segmentation**: It can segment different parts of a network, isolating sensitive data from the rest of the network and reducing exposure to threats.

4. **Encryption Protection**: It can enforce encryption protocols (e.g., HTTPS) to ensure that data transmitted over the network is encrypted and secure from eavesdropping.
5. **Logging and Monitoring**: It logs traffic and events for later analysis, helping to detect and respond to potential security incidents in real-time.

7.3 What can't a firewall protect against?

Ans: Firewalls cannot protect against:

1. **Internal Threats**: They can only control traffic between networks but cannot stop attacks originating from within the network (e.g., insider threats).
2. **Encrypted Traffic**: Firewalls may not inspect encrypted traffic (unless they are configured to decrypt it), meaning malicious activities within encrypted communications can bypass protection.
3. **Social Engineering Attacks**: Firewalls cannot prevent phishing or other forms of social engineering that manipulate users into giving up sensitive information.
4. **Malware on Endpoints**: Firewalls do not protect against malware installed on devices or endpoints within the trusted network.
5. **Application-Level Attacks**: Basic firewalls may not be effective against application-layer attacks like SQL injection or cross-site scripting (XSS).

7.4 How is a firewall different from an IDS and an IPS? Explain.

Ans:

Aspect	**Firewall**	**Intrusion Detection System (IDS)**	**Intrusion Prevention System (IPS)**

Primary Function	Controls network traffic based on rules (allow/block).	Monitors network traffic for suspicious activity and alerts administrators.	Monitors network traffic for suspicious activity and actively blocks malicious traffic.
Detection	Filters traffic based on predefined security rules (IP, port, protocol).	Detects anomalies, policy violations, and known attack patterns in real-time.	Similar to IDS, but with the added ability to block harmful traffic.
Response	Allows or blocks traffic but does not actively detect attacks.	Alerts administrators when suspicious activity is detected.	Actively prevents attacks by blocking malicious traffic in real time.
Placement	Typically placed at the perimeter of the network to filter traffic.	Can be placed within the network to detect threats after they enter.	Can be placed in-line with the network to actively block threats.
Action on Detection	Takes action by blocking or allowing traffic based on set rules.	Does not take action but generates alerts for further investigation.	Takes immediate action to block detected threats, often in real-time.

Conclusion: Implemented and configured Firewall using Iptable. Carried the experiment in Virtual Machine.