CS915/435 Advanced Computer Security - Network Security (III)

Firewall

Outline

- Packet Sniffing and Spoofing
- Attacks on the TCP protocol
- **Firewall**
    - **What are firewalls?**
    - **Types of Firewalls: packet filter, stateful firewall, application firewall**
    - **Evading Firewalls**
- DNS attacks

Introduction

- A firewall is a part of computer system or network designed to stop unauthorised traffic flowing from one network to another.
- Main goal is to separate trusted and untrusted components of a network.
- Also used to differentiate networks within a trusted network.
- Main functionalities are filtering data, redirecting traffic and protecting against network attacks.

Requirements of a firewall (1994)

1. All the traffic between trust zones should pass through firewall.
2. Only authorised traffic, as defined by the security policy, should be allowed to pass through.
3. The firewall itself must be immune to penetration, which implies using a hardened system with secured Operating Systems.

Bellovin, Chestwick, “Network Firewalls,” *Communications Magazine*, 1994.

Firewall Policy

- User control: Controls access to the data based on the role of the user who is attempting to access it. Applied to users inside the firewall perimeter.
- Service control: Controls access by the type of service offered by the host. Applied on the basis of network address, protocol of connection and port numbers.
- Direction control: Determines the direction in which requests may be initiated and are allowed to flow through the firewall. It tells whether the traffic is “inbound” (From the network to firewall) or vice-versa “outbound”

Firewall actions

Accepted: Allowed to enter the connected network/host through the firewall.

Denied: Not permitted to enter the other side of firewall.

Rejected: Similar to “Denied”, but tells the source about this decision through ICMP packet.

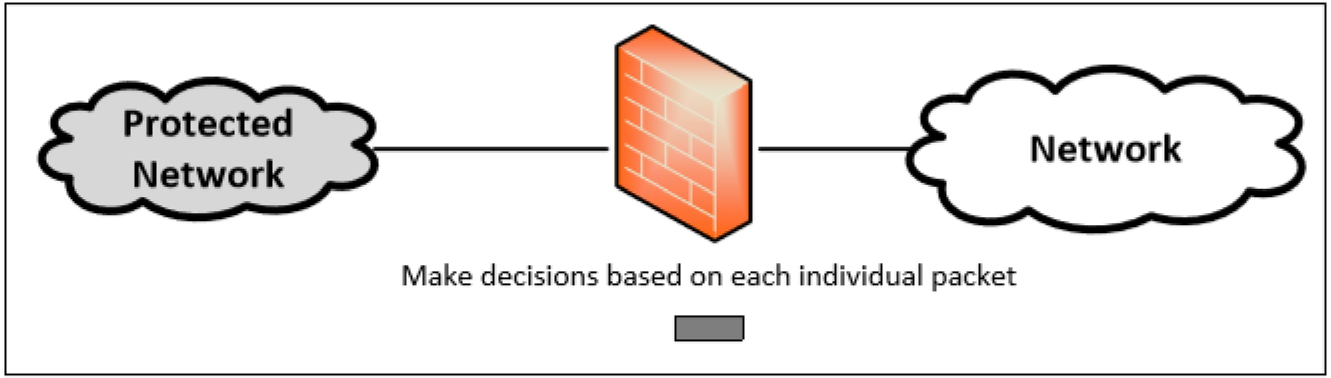
> *Ingress filtering: Inspects the incoming traffic to safeguard an internal network and prevent attacks from outside.*
>
> *Egress filtering: Inspects the outgoing network traffic and prevent the users in the internal network to reach out to the outside network. For example like blocking social networking sites in school, or Great Firewall in China*

Types of filters

Depending on the mode of operation, there are three types of firewalls :

- Packet Filter Firewall
- Stateful Firewall
- Application/Proxy Firewall

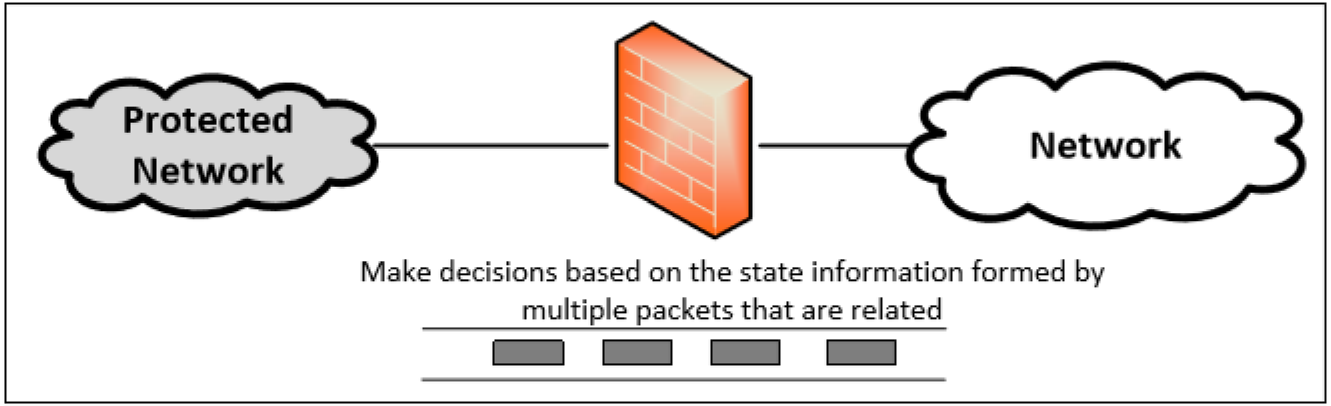
Packet Filter Firewall



- Controls traffic based on the information in **packet headers**, without looking into the payload that contains application data.

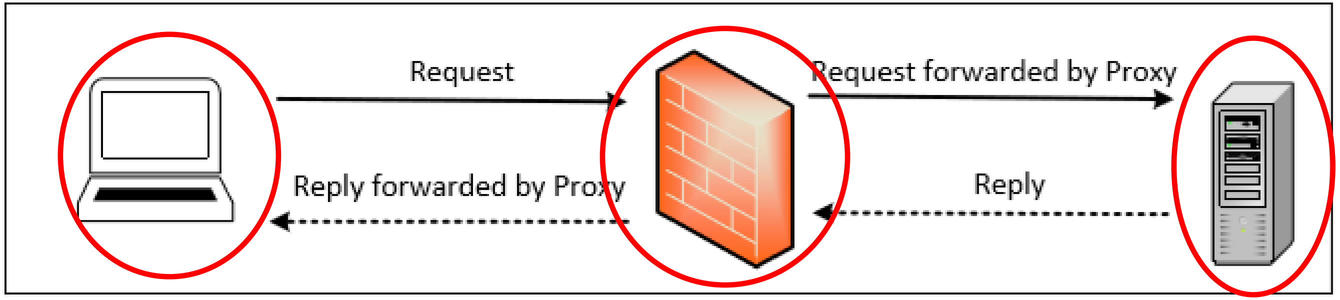
- Doesn’t pay attention to if the packet is part of existing stream or traffic.
- Doesn’t maintain the states about packets. Also called Stateless Firewall.

Stateful Firewall


Protected Network
Network
Make decisions based on the state information formed by multiple packets that are related

- Tracks the state of traffic by monitoring all the connection interactions until is closed.
- **A connection state table** is maintained to understand the **context** of packets.
- Example: only allowing packets that belong to an existing connection (not possible with packet filter), hence reducing the chances of spoofing.

Application/Proxy Firewall



- Controls input, output and access from/to an application or service.
- Acts as an intermediary by impersonating the intended recipient.

- The client's connection terminates at the proxy and a separate connection is initiated from the proxy to the destination host.
- Data on the connection is analysed up to the application layer to determine if the packet should be allowed or rejected.

Building a Firewall using Netfilter

Packet filter firewall implementation in Linux

- Packet filtering can be done inside the kernel.
- Need changes in the kernel

You can build your own packet filter module by using

- Netfilter: Provides hooks at critical points on the packet traversal path inside Linux Kernel
- Loadable Kernel Modules: Allow privileged users to dynamically add/remove modules to the kernel, so there is no need to recompile the entire kernel.

Alternatively, you can use an existing built-in packet filter (e.g., Iptables in Linux)

Stateful Firewall using Connection Tracking

- A stateful firewall monitors incoming and outgoing packets over a period of time.
- Records attributes like IP address, port numbers, sequence numbers. Collectively known as connection states.
- A connection state, in context of a firewall signifies whether a given packet is a part of an existing flow or not.
- Hence, it is applied to both connection-oriented (TCP) and connectionless protocols (UDP and ICMP).

Connection Tracking Framework in Linux

- nf_conntrack is a connection tracking framework in Linux kernel built on the top of netfilter with the additional function of storing *context* in memory.
- Each incoming packet is marked with a connection state as described:
    - NEW: The connection is starting and packet is a part of a valid sequence. It only exists for a connection if the firewall has only seen traffic in one direction.
    - ESTABLISHED: The connection has been established and is a two-way communication.
    - RELATED: Special state that helps to establish relationships among different connections. E.g., FTP Control traffic and FTP Data traffic are related.
    - INVALID : This state is used for packets that do not follow the expected behavior of a connection.
- iptables can use nf_conntrack to build stateful firewall rules.

Example: Set up a Stateful Firewall

```
// -A OUTPUT: Append to existing OUTPUT chain rules.
// -p tcp: Apply on TCP protocol packets.
// -m conntrack: Apply the rules from conntrack module.
// --ctsate ESTABLISHED,RELATED: Look for traffic in ESTABLISHED or
   RELATED states.
// -j ACCEPT: Let the selected packets through.

$ sudo iptables -A OUTPUT -p tcp -m conntrack --ctstate
   ESTABLISHED,RELATED -j ACCEPT
```

- This rule allows only outgoing TCP packets if they belong to an established TCP connection. What is it for?
  - A compromised internal machine wants to send information out.
  - It can’t directly initiate a TCP connection to the outside since the reply must go to the TCP servers (ssh, http) at the Firewall; otherwise will be blocked.
  - However, it may simply send TCP packets containing the exfiltrated information.
  - This rule prevents the compromised machine from doing this.

Application/Proxy Firewall and Web Proxy

- An application firewall controls input/output and access from/to an application of service, by inspecting network traffic at all layers (up to application layer)
- Typical implementation of an application firewall is a proxy (application proxy)
- Web proxy: a widely used proxy firewall to control web browsing
- The critical challenge in setting up a web proxy is to ensure that all the web traffic goes through the proxy server.
  - Configuring each host computer to redirect all the web traffic to the proxy. (but some users may change configuration to bypass the restriction.)
  - Place web proxies on a network bridge that connects internal and external networks - if users change configuration, they lose web access

Other applications of web proxy

- Anonymizing Proxy: One can also use proxies to hide the origin of a network request from servers. As the servers can only see the traffic after it passes through proxies, source IP will be the proxy’s and actual origin is hidden.
- Proxy can also be used to evade egress filtering.
    - If a firewall conducts packet filtering based on destination address, we can evade this firewall by browsing the Internet using web proxy.
    - The destination address will be modified to the proxy server which defeats the packet filtering rules of the firewall.
    - Why do you want to do that?

Evading Firewalls

There are situations where you may want to evade firewalls in both directions

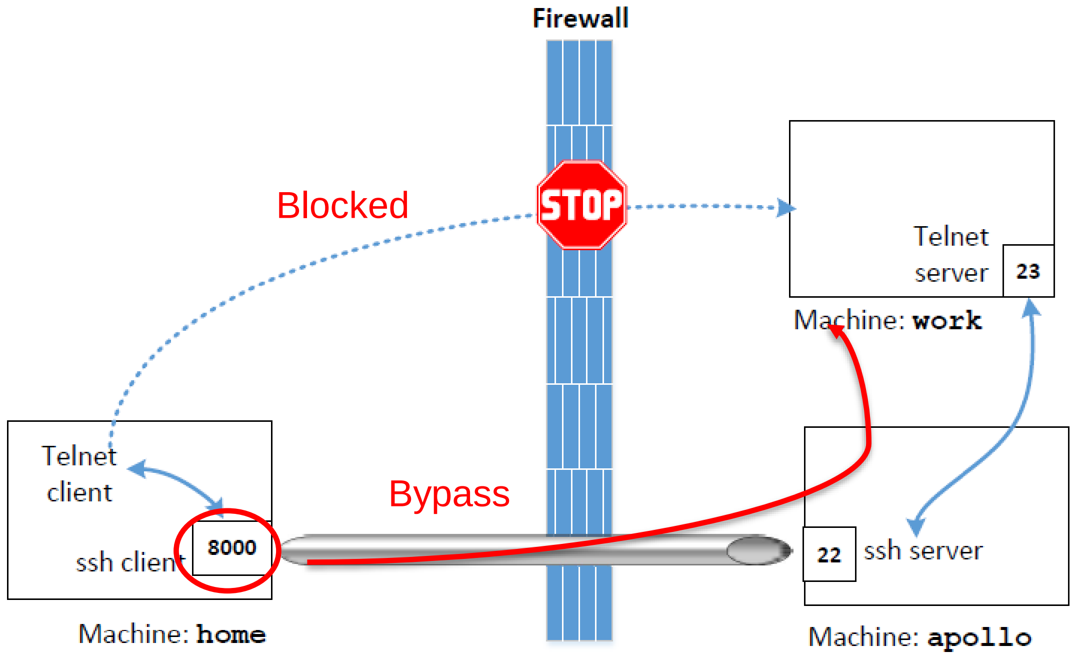
- SSH Tunneling
- Dynamic Port Forwarding
- Reverse SSH Tunneling
- Virtual Private Network

1) SSH Tunneling to Evade Firewalls

Scenario :

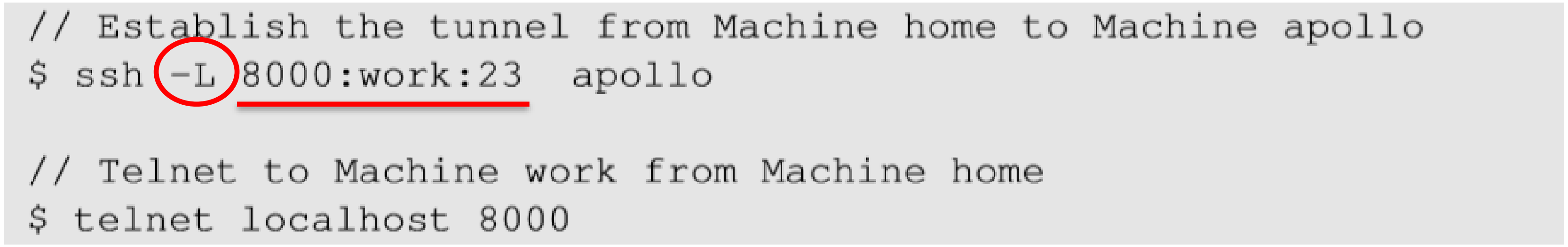
We are working in a company and need to telnet to a machine called “work”. Sometimes as we work from home, we need to telnet from machine “home” to “work”. However, the company’s firewall blocks all incoming traffic which makes telnet from “home” impossible.The company’s firewall does allow ssh traffic to reach its internal machine “apollo”, where we have an account. How can we use this machine to evade the firewall?

SSH Tunneling to Evade Firewalls



- Establish a ssh tunnel between “home” and “apollo”.
- On the “home” end, the tunnel receives TCP packets from the telnet client.
- It forwards the TCP data to “apollo” end, from where the data is out in another TCP packet which is sent to machine “work”.
- The firewall can only see the traffic between “home” and “apollo” and not from “apollo” to “work”. Also ssh traffic is encrypted.

SSH Tunneling to Evade Firewalls



```
// Establish the tunnel from Machine home to Machine apollo
$ ssh -L 8000:work:23  apollo

// Telnet to Machine work from Machine home
$ telnet localhost 8000
```

- Establish an ssh tunnel from “home” to “apollo”. This tunnel will forward TCP data received on 8000 on “home” to port 23 on work (-L for binding address).
- After establishing the tunnel, telnet to the port 8000, and the telnet traffic will be forwarded host work via the ssh tunnel.

SSH Tunneling to Evade Firewalls

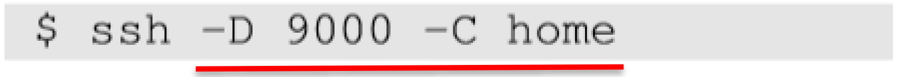
Scenario : We are working in a company and working on a machine called “work”. We would like to visit facebook, but the company has blocked it to prevent employees from getting distracted. We use an outside machine “home” to bypass such a firewall. How can we bypass it?

```
$ ssh -L 8000:www.facebook.com:80 home
```

- We establish an ssh tunnel from “work” to “home”.
- After establishing the tunnel, we can type “localhost:8000” in our browser.
- The tunnel will forward our HTTP requests to the website via home.
- The firewall can only see the ssh traffic between “work” and “home” and not the actual web traffic between “work” and “facebook.com”.

2) Dynamic Port Forwarding

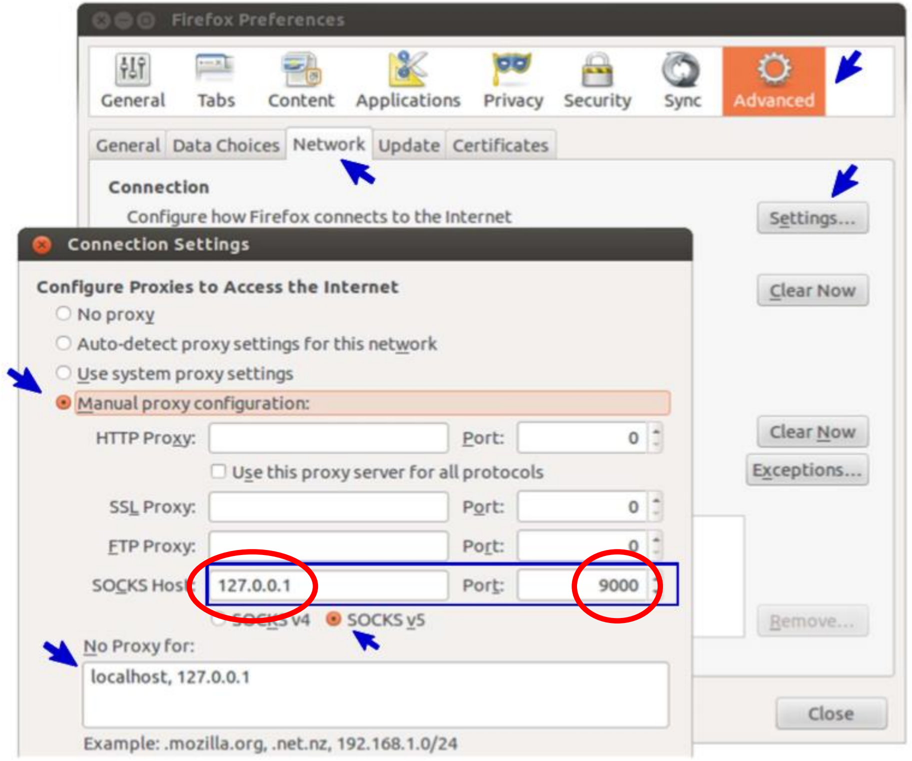
- The previous solution is also called **Static Port Forwarding**.
- However, doing this for many websites will be tedious.
- SSH also supports **Dynamic Port Forwarding**



```
$ ssh -D 9000 -C home
```

- It establishes an ssh tunnel between localhost:9000 and the machine “home”.
- We should redirect all the web requests to localhost:9000, which can be done by configuring browser setting (next slide)
- What we have set up is a ***SOCKS proxy*** on our local host.
- SSH will send the TCP data over the tunnel to the machine “home” which will communicate with the blocked site.

Dynamic Port Forwarding - configure SOCKS Proxy



Connection

Configure how Firefox connects to the Internet

Configure Proxies to Access the Internet

No proxy

Auto-detect proxy settings for this network

Use system proxy settings

[✓ Manual proxy configuration:]

HTTP Proxy: Port: 0

Use this proxy server for all protocols

SSL Proxy: Port: 0

FTP Proxy: Port: 0

SOCKS Host: 127.0.0.1 Port: 9000

SOCKS v4 [✓ SOCKS v5]

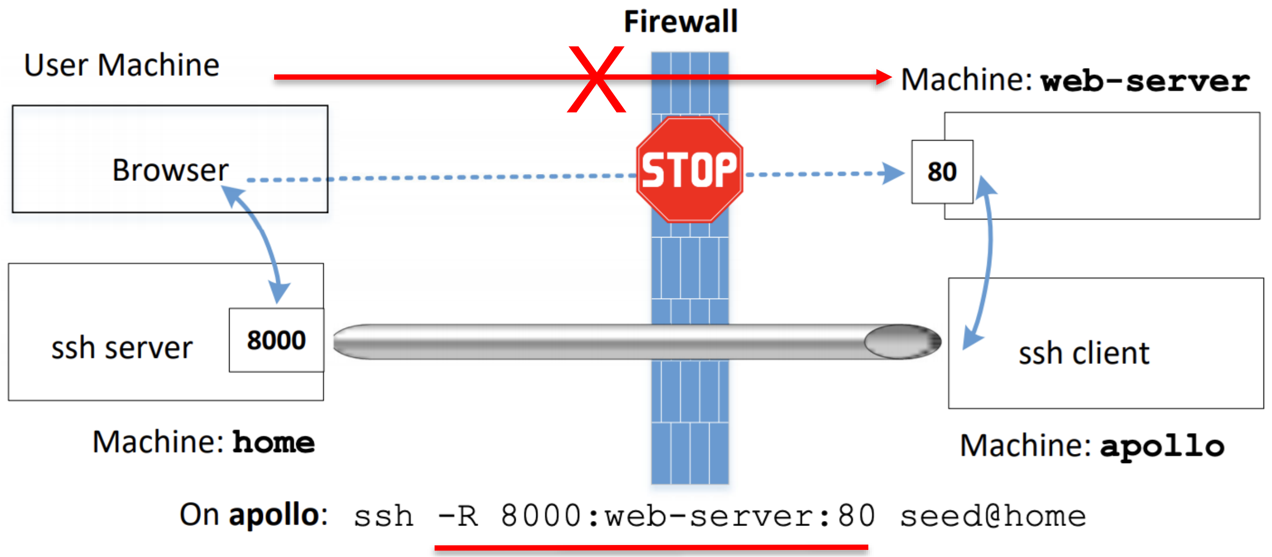
No Proxy for:

localhost, 127.0.0.1

Example: .mozilla.org, .net.nz, 192.168.1.0/24

The client software must have a native SOCKS support to use SOCKS proxies (e.g., Firefox)

3) Reverse SSH tunneling



On **apollo**: `ssh -R 8000:web-server:80 seed@home`

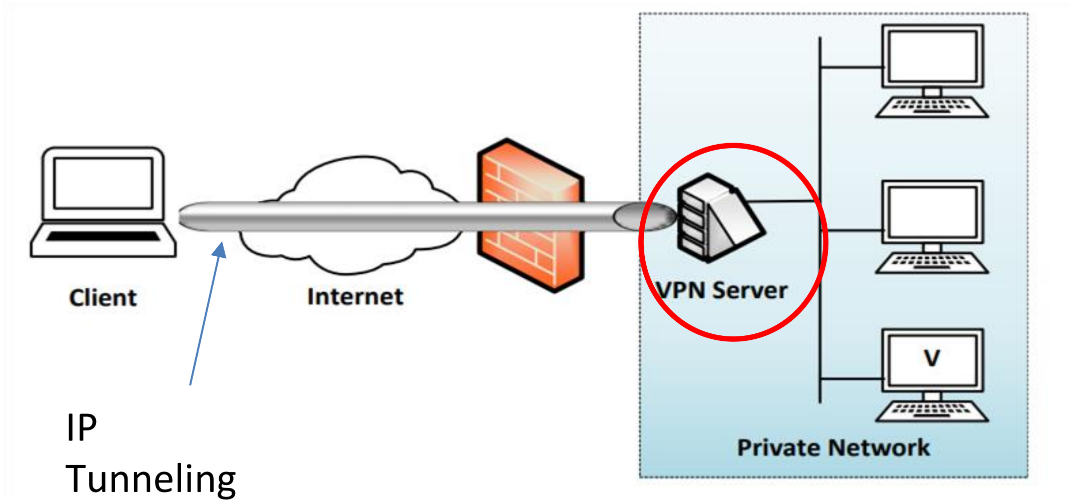
- Firewall may block any incoming ssh sessions.
- Assume it doesn't block outgoing ssh
- We can use reverse SSH tunneling to access an internal website

4) Using VPN to Evade Firewall

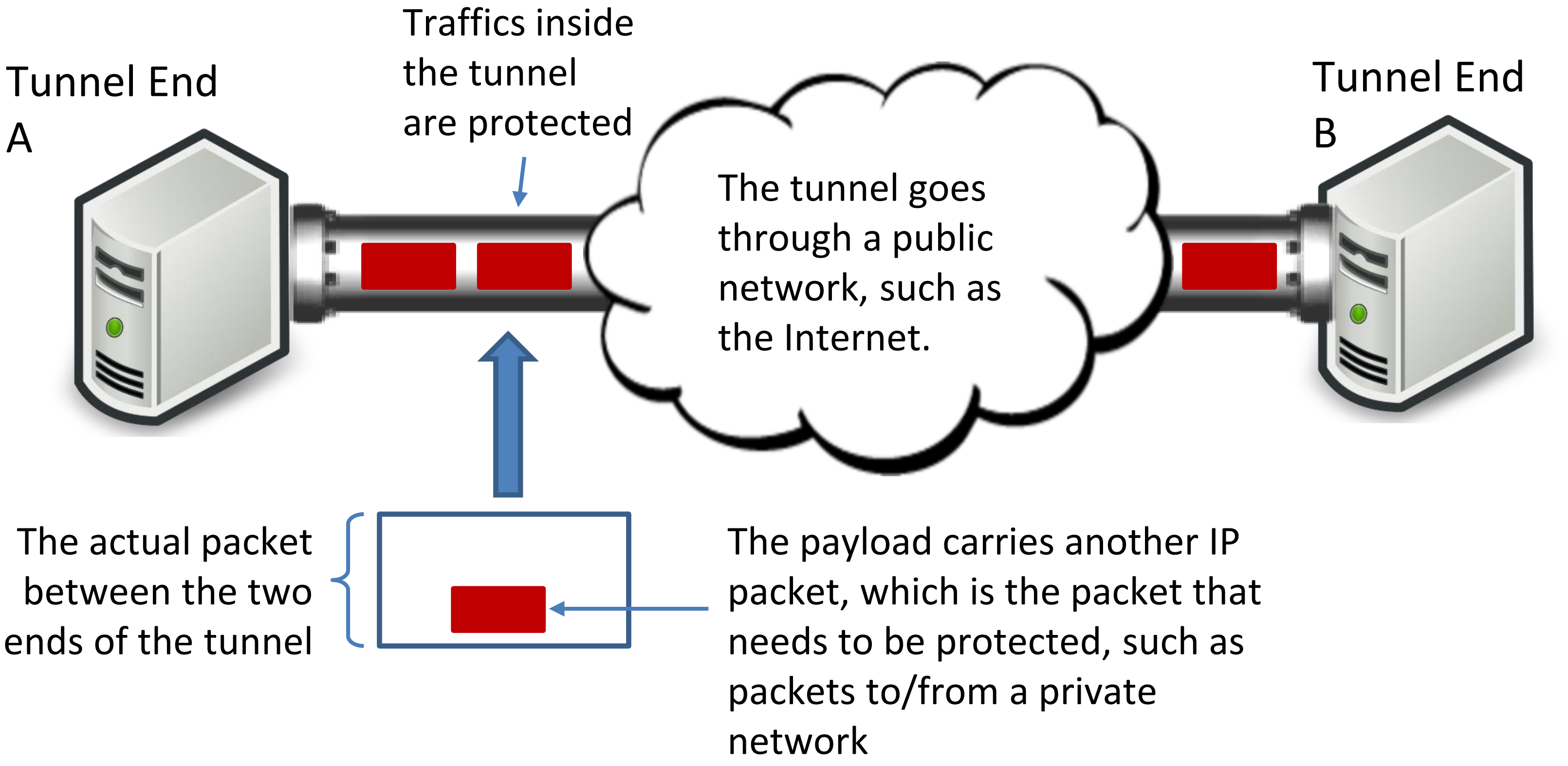
Using VPN, one can create a tunnel between a computer inside the network and another one outside. IP packets can be sent using this tunnel. Since the tunnel traffic is encrypted, firewalls are not able to see what is inside this tunnel and cannot conduct filtering.

A Typical Setup

This is a typical VPN setup where the "Client" machine wants to connect with machine "V" on a private network. "Client" uses the "VPN Server" to get authenticated to the private network



IP Tunneling

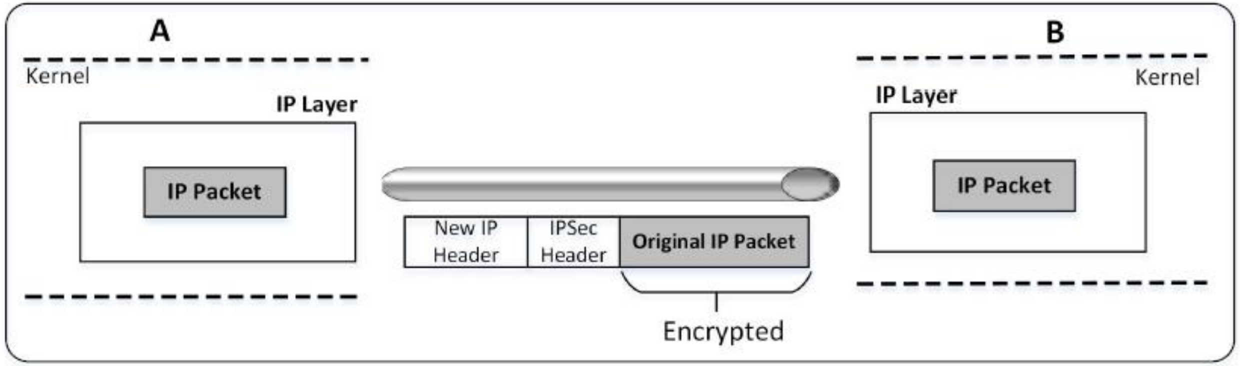

Tunnel End A
Traffics inside the tunnel are protected
The tunnel goes through a public network, such as the Internet.
Tunnel End B
The actual packet between the two ends of the tunnel
The payload carries another IP packet, which is the packet that needs to be protected, such as packets to/from a private network

Two Types of IP Tunneling

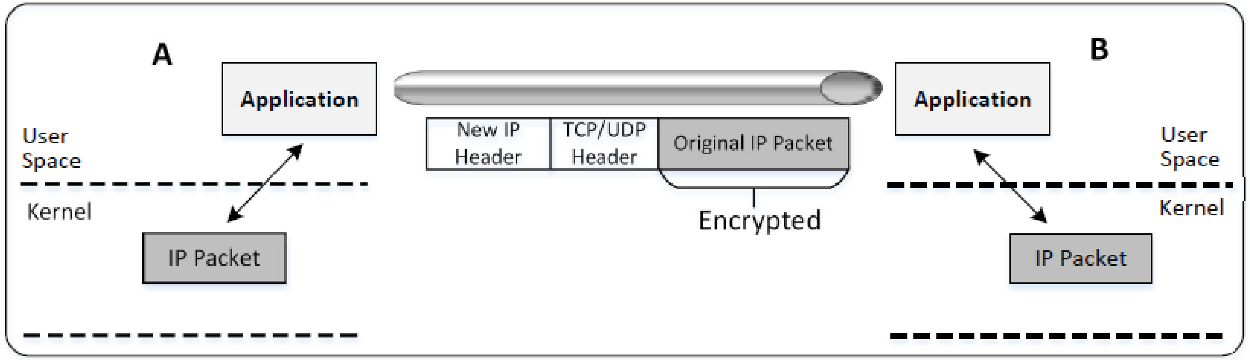
- IPSec Tunneling:
  - Utilises the Internet Protocol Security protocol
  - IPSec has a mode called Tunneling mode, where the original IP packet is encapsulated and placed into a new IP packet
- TLS/SSL Tunneling:
  - Tunneling done outside the kernel, at the application level
  - Idea is to put each VPN-bound IP packet inside a TCP or UDP packet
  - The other end of the tunnel will extract the IP packet from the TCP/UDP payload
  - To secure the packets, both ends will use TLS/SSL protocol on top of TCP/UDP

Two Types of IP Tunneling

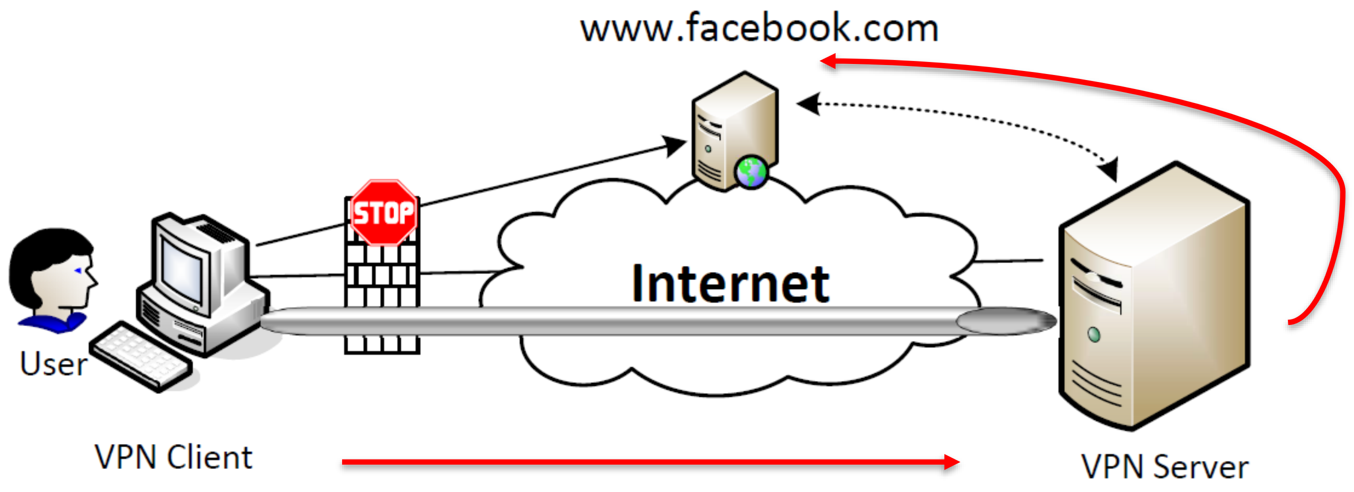
IPSec Tunneling



TLS/SSL Tunneling



Bypassing Firewall using VPN: the Main Idea



- Send our Facebook-bound packets to the VPN server via the tunnel
- VPN server will release our Facebook-bound packets to the Internet
- Facebook's reply packets will be routed to the VPN server (question: why)
- VPN server sends the reply packets back to us via the tunnel

Summary

- The concept of firewall
- Perform packet filtering via netfilter, iptables
- Stateful firewalls and web proxy
- Bypassing firewalls