Firewalls and Tunnels

Michalis Polychronakis

Stony Brook University
Typical Network Topology
**Firewalls:** separate local networks from the Internet

- LAN: Internal networks
- WiFi: Public/private wireless networks
- DMZ ("Demilitarized Zone"): Publicly accessible servers
Firewalls: Reality

- LAN
- WiFi
- DMZ
- Internet
Firewalls

Filter traffic according to a predefined *policy*

Mostly statically defined, but dynamic updates are possible (block an ongoing DoS attack, protocols with dynamically negotiated port numbers, etc.)

Barrier between administrative domains

Internal networks vs. the outside world
Mission-specific subnets/VLANs (publicly accessible servers, machine clusters, user groups, printers, VoIP, IoT, …)
Less trusted segments (guest WiFi network, contractors, …)

Main strategies

Default-deny: drop everything unless explicitly allowed
Default-allow: block specific protocols/hosts/ports/…
Firewalls: why are they needed?

Hosts may run vulnerable services: prevent outside access

  Limit the “attack surface” ➔ expose fewer services

Internal hosts may get compromised: damage control

  Prevent propagation, outgoing attacks, exfiltration, …

Hide the structure of private networks: hinder network reconnaissance

  Block port scanning, service fingerprinting, …

Network intelligence: log interesting events

  Troubleshooting, monitoring/tuning, auditing, forensics, …

Simply block unwanted traffic: policy enforcement

  Noise, backscatter, spoofed packets, DoS attacks, brute-force password guessing, Bittorrent, Facebook, Netflix, games, VPNs, …
A Theory of Firewalls (Bellovin)

Three properties must hold for a firewall to be effective

The firewall should be placed at a topological chokepoint

- Not always true in modern enterprises: links to suppliers/contractors, cellular connectivity, VPN/proxy software, …

“Inside” nodes share the same security policy

- Do they? BYOD, IoT, work from home, …

“Inside” nodes are trusted, “outside” hosts are untrusted

- BYOD: an already infected device may appear inside the network
- Internal hosts can be infected due to client-side attacks (e.g., drive-by download attacks, malware, phishing, supply chain attacks, …)
- Insider threats, disgruntled employees, …
Stateless Filtering

Decide by considering each packet in isolation
- Rules mostly based on network and transport layer fields
- Simple implementation: no need to keep state

Limitations
- Dynamically negotiated/non-standard port numbers (FTP, SIP, BitTorrent, …)
- Connectionless protocols (e.g., UDP): cannot distinguish between queries and replies
- IP fragmentation: port numbers are present only in the first fragment
- Rulesets can get complex and hard to understand

Still useful for simple scenarios
- Ingress/egress filtering, strict configurations, …
Stateless Firewalls and TCP

Common configuration: block incoming but allow outgoing connections

- Incoming (externally initiated) connections should be blocked
- Incoming packets of established connections should be allowed

Can be achieved without keeping state

- Block incoming SYN-only packets
- Allow incoming packets with the ACK bit set

Not an ideal solution: *ACK scanning*
**ACK-scanning** (nmap -sA)

Can determine whether a stateless firewall is used
   Not whether a specific port is open or not

When an ACK is sent to a closed port, or sent out-of-sync to an open port, the expected behavior is to respond with a RST
   Stateful firewalls discard out-of-sync ACK packets, leading to no response

Step 1: SYN $\rightarrow$ SYN/ACK or RST
Step 2: ACK $\rightarrow$ RST  \textit{The port is unfiltered by any firewall type}

Step 1: SYN $\rightarrow$ SYN/ACK
Step 2: ACK $\rightarrow$ no response  \textit{Stateful firewall}

Step 1: SYN $\rightarrow$ no response
Step 2: ACK $\rightarrow$ RST  \textit{Stateless firewall}
Stateful Filtering

Firewall keeps per-connection state

- Track TCP three-way handshake, UDP query/responses, …
- Decisions are made by considering each packet in the context of the connection/session it belongs to

Most common firewall type

More flexible policies

- Internally vs. externally initiated connections/sessions

Still cannot handle dynamically negotiated port numbers and higher-level protocol semantics

- Missing application-level context
Network Address Translation

Share a public IP address with many internal hosts
   In general: remap an IP address space into another
   Global shortage of IPv4 addresses
   Widely used (home networks, wireless networks, …)

Rewrite packet address and port information (per-connection state)
NAT vs. Stateful Firewall

Similar functionality and state

NAT in addition **modifies** packets: performs address/port translation

Are NATs firewalls?

Not in the strict sense, as they do not fully track the TCP 3-way handshake or any other higher-layer state

But they *do* provide some protection: allow only outgoing connections

Internal hosts can become accessible through *port forwarding*

Explicitly map a local IP:port to a public IP:port
**UPnP**

Universal Plug and Play

Widely supported protocol by home routers to enable device discovery and NAT traversal

"Please allow external hosts to reach me on port 12345"

Skype, Bittorrent, games, …

No authentication!

Malware can easily punch holes

Worse: Flash, XSS, …

Even worse: external requests (!)
Security Flaws in Universal Plug and Play: Unplug, Don’t Play

This morning we released a whitepaper entitled Security Flaws in Universal Plug and Play. This paper is the result of a research project spanning the second half of 2012 that measured the global exposure of UPnP-enabled network devices. The results were shocking to say the least. Over 80 million unique IPs were identified that responded to UPnP discovery requests from the internet. Somewhere between 40 and 50 million IPs are vulnerable to at least one of three attacks outlined in this paper. The two most commonly used UPnP software libraries both contained remotely exploitable vulnerabilities. In the case of the Portable UPnP SDK, over 23 million IPs are vulnerable to remote code execution through a single UDP packet. All told, we were able to identify over 6,900 product versions that were vulnerable through UPnP. This list encompasses over 1,500 vendors and only took into account devices that
Generic Port Forwarding

Bypass firewall policies!

Example: connect to a host that is blocked by a local firewall policy

Remote host: `nc -l -p 12345 -c ‘nc blocked.com 80’`
Local host: `wget remote.edu:12345`

Or using SSH local port forwarding

`ssh -L 12345:blocked.com:80 remote.edu`

Also the other way around: remote port forwarding

Example: allow public access to a server running in a private network

`ssh -R 8080:localhost:80 remote.edu`
Proxies

Intermediate “stepping stones”
  Operate at the application layer
  Act as both a client and a server

Application-level filtering
  Example: HTTP-level filtering (domains, URLs, ads, …)

Many non-security uses as well
  HTTP content caching (one of the first uses of web proxies)
  Reverse proxies (in front of application servers): quickly serve the same dynamically-generated content
  Transcoding (reduce the resolution of media content for mobile devices)

Explicit vs. transparent proxies
  The former require application configuration
SOCKS Proxies

Also known as circuit-level gateways

Socket Secure (SOCKS): protocol for generic forwarding of packets through a proxy

Supported by many applications and protocols

HTTP, FTP, SMTP, POP3, NNTP, …

Example: dynamic application-level port forwarding

```
ssh -D 12345 sshserver.com
chrome --proxy-server='socks://localhost:12345'
```
A secure socks5 proxy, designed to protect your Internet traffic.

If you want to keep a secret, you must also hide it from yourself.

Super Fast
Bleeding edge techniques using Asynchronous I/O and Event-driven programming.

Flexible Encryption
Secured with industry level encryption algorithm. Flexible to support custom algorithms.

Cross Platform
Available on most platforms, including Windows, Linux, Mac, Android, iOS, and OpenWRT.

Open Source
Totally free and open source. A worldwide community devoted to deliver bug-free code and long-term support.

Mobile Ready
Optimized for mobile device and wireless network, without any keep-alive connections.

Easy Deployment
Easy deployment with pip, aur, freshports and many other package manager systems.
Application-level “Firewalls”

Similar to proxies, but less generic
- Application-specific filtering
- Often built into applications

Example: SMTP
- Spam filtering, phishing detection, attachment scanning, …

Overlap with more generic intrusion detection systems (future lecture)

Recent buzzword: web application firewalls (WAF)
- Server-side HTTP filtering for common attack patterns (XSS, SQL injection, …)
- A specific instance of application-level filtering/scanning
Host-based Firewalls

Firewalls running on end hosts

- Windows firewall
- IPtables

“Personal” firewalls: apply common-sense policies (deny incoming, allow outgoing)

 Particularly important for home users, laptops, etc.

On-by-default client firewall deployment contributed significantly in ending the era of internet worms
### Simple IPtables Example

```bash
# flush all chains
iptables -F
iptables -X

# defaults for predefined chains
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

# allow anything on localhost interface
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

# allow all traffic from specific subnets
iptables -A INPUT -s 128.59.0.0/255.255.0.0 -j ACCEPT
iptables -A INPUT -s 160.39.0.0/255.255.0.0 -j ACCEPT
```
Simple IPtables Example

# allow all inbound traffic for specific services
iptables -A INPUT -p tcp -m tcp --syn --dport 22 -j ACCEPT
iptables -A INPUT -p tcp -m tcp --syn --dport 80 -j ACCEPT

# allow inbound established and related outside communication
iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT

# allow ICMP
iptables -A INPUT -p icmp -j ACCEPT

# allow all outgoing traffic
iptables -A OUTPUT -j ACCEPT

Is this a good idea?
Before Host-based Firewalls:

After Host-based Firewalls:
Per-process Firewall

Most “personal” firewalls still allow all outgoing traffic by default

Severe usability problems otherwise

Do all programs really need to communicate with the outside world?

Deny by default and whitelist only what is needed

No easy solution for this in most OSes – need to rely on hacks or third party solutions

GlassWire, TinyWall, Windows Firewall Control, …
Virtual Private Networks

Users may not always be behind the firewall, but still need full access to internal network resources

- Offices at different locations, employees on the move, access to home “cloud,” …

VPNs bridge private networks across a public (untrusted) network

- Virtual point-to-point secure connections (encryption)
- Create a *trusted* shared network among them

Remote host/network virtually becomes part of the local network
VPN Examples

VPN Implementations

Tunneling/encapsulation: packets are transferred as data over another protocol

- L2 over L4 (PPTP), L2 over L2 (PPPoE), L2 over L3 (L2TP), L3/L4 over L3 (IPsec), L2/L3 over L4 (OpenVPN)

Three major families in wide use today:

**PPTP**: L2, introduced in 1995, commonly used in Windows ➔ *Broken*

**IPsec**: L3, widely supported by most operating systems
  - Completely transparent to applications
  - Tunnel is handled directly by the OS

**SSL**: Application layer – OpenVPN
  - User-space implementation, multiplatform
  - Typically requires the installation of a software client
VPN Implementations

Tunneling/encapsulation: packets are transferred as data over another protocol

- L2 over L2 (PPTP), L2 over L2 (PPPoE), L2 over L3 (L2TP), L3/L4 over L3 (IPsec), L2/L3 over L4 (OpenVPN)

Three major ones in wide use today:

- **PPTP**: L2, introduced in 1995, commonly used in Windows ➔ Broken

- **IPsec**: L3, widely supported across most operating systems
  - Completely transparent to applications
  - Tunnel is handled directly by the OS

- **SSL**: Application layer – OpenVPN
  - User-space implementation, multiplatform
  - Typically requires the installation of a software client

---

**Just use Wireguard**
WireGuard® is an extremely simple yet fast and modern VPN that utilizes state-of-the-art cryptography. It aims to be faster, simpler, leaner, and more useful than IPsec, while avoiding the massive headache. It intends to be considerably more performant than OpenVPN. WireGuard is designed as a general purpose VPN for running on embedded interfaces and super computers alike, fit for many different circumstances. Initially released for the Linux kernel, it is now cross-platform (Windows, macOS, BSD, iOS, Android) and widely deployable. It is currently under heavy development, but already it might be regarded as the most secure, easiest to use, and simplest VPN solution in the industry.

# Simple & Easy-to-use

WireGuard aims to be as easy to configure and deploy as SSH. A VPN connection is made simply by exchanging very simple public keys – exactly like exchanging SSH keys – and all the rest is transparently handled by WireGuard. It is even capable of roaming between IP addresses, just like Mosh. There is no need to manage connections, be concerned about state, manage daemons, or worry about what’s under the hood. WireGuard presents an extremely basic yet powerful interface.

Cryptographically Sound

WireGuard uses state-of-the-art cryptography, like the Noise protocol framework, Curve25519, ChaCha20, Poly1305,
VPN Risks

Personal use of VPNs has become popular for bypassing restrictions
  Country-based content, censorship, corporate/school/parental controls, …

A third-party VPN server can observe all our traffic (!)
  Be wary of VPN services that claim “privacy” and “anonymity”
  VPN services are still subject to local laws
  Shady services may monetize our traffic

Client-side VPN software is too powerful
  Can monitor (spy on) system-wide activities besides the traffic itself
  Most operating systems have built-in VPN support!
  There shouldn’t be any need to install closed-source VPN software
Trivially easy to set up a **personal** VPN in the cloud!

No excuse for not using a VPN when you are in a public WiFi!
“Secure Gateways”

Nowadays it is common for most of the discussed technologies to be consolidated into a single box

Routing, Firewall, NAT, VPN, Proxy, WiFi access point, …

Common in home and enterprise settings

Routers and firewalls used to be “simple” devices – not anymore

Features ➔ complexity ➔ security issues

Critical hosts in the network that need to be protected

Administrative interface, OS patches/updates, service vulnerabilities, …
Welcome to the internet's largest and most updated default router passwords database.

Select Router Manufacturer:

- CISCO

Find Password

<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>Model</th>
<th>Protocol</th>
<th>Username</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>CISCO</td>
<td>CACHE ENGINE</td>
<td>CONSOLE</td>
<td>admin</td>
<td>diamond</td>
</tr>
<tr>
<td>CISCO</td>
<td>CONFIGMAKER</td>
<td></td>
<td>cmaker</td>
<td>cmaker</td>
</tr>
<tr>
<td>CISCO</td>
<td>CNR Rev. ALL</td>
<td>CNR GUI</td>
<td>admin</td>
<td>change me</td>
</tr>
<tr>
<td>CISCO</td>
<td>NETRANGER/SECURE IDS</td>
<td>MULTI</td>
<td>netranger</td>
<td>attack</td>
</tr>
<tr>
<td>CISCO</td>
<td>BBSM Rev. 5.0 AND 5.1</td>
<td>TELNET OR NAMED PIPES</td>
<td>bbsd-client</td>
<td>change me2</td>
</tr>
<tr>
<td>CISCO</td>
<td>BBSD MSDE CLIENT Rev. 5.0 AND 5.1</td>
<td>TELNET OR NAMED PIPES</td>
<td>bbsd-client</td>
<td>NULL</td>
</tr>
</tbody>
</table>
Owning Modems And Routers Silently

Modems

Do you have cable Internet? Own a surfboard modem? Since most of my buddies in AZ do, I sent them to this page and to my amusement, they got knocked off the net for a few minutes. How? Javascript. Specifically a CSRF in the Motorola Surfboard.

The Surfboard cable modem offers little in functionality besides rebooting unless of course I wanted to be malicious and remove all settings on the cable modem and essentially turn it into a door stop until the thing can be activated again by the ISP.
VPNFilter malware infecting 500,000 devices is worse than we thought

Malware tied to Russia can attack connected computers and downgrade HTTPS.

DAN GOODIN - 6/6/2018, 9:00 AM

Two weeks ago, officials in the private and public sectors warned that hackers working for the Russian government infected more than 500,000 consumer-grade routers in 54 countries with malware that could be used for a range of nefarious purposes. Now, researchers from Cisco's Talos security team say additional analysis shows that the malware is more powerful than originally thought and runs on a much broader base of models, many from previously unaffected manufacturers.

The most notable new capabilities found in VPNFilter, as the malware is known, come in a newly discovered module that performs an active man-in-the-middle attack on incoming Web traffic. Attackers can use this ssler module to inject malicious payloads into traffic as it passes through an infected router. The payloads can be tailored to exploit specific devices connected to the infected network. Pronounced “essler,” the module can also be used to surreptitiously modify content delivered by websites.
Discussion Topic: Port Knocking

Open otherwise firewalled ports on demand by “knocking” the right combination of ports

- Sometimes recommended for “securing” SSH and other servers
- Firewall opens the port only after observing connection attempts to the right combination of (closed) ports
- Variation: single packet authorization (packet containing a magic value)

*Is it useful or pointless?*
Discussion Topic: Running SSH on a non-default port

Is it useful or pointless?