Final Report on "FoggyWeb_ Targeted NOBELIUM malware leads to persistent backdoor _ Microsoft Security Blog" by Team Pineapples

Approach to mapping Finished Reporting to ATT&CK:

1. The finished report of 23 pages was initially divided between the group members so that each individual can analyse his portion with greater focus and higher accuracy.

2. The report by Microsoft gives an in-depth analysis of the malware NOBELIUM, the actor behind the SUNBURST backdoor, which they refer to as FoggyWeb. Every statement was carefully read to extract information about how the attack was sewn and what was the target of the attack. For example: the statement **"Microsoft.IdentityServer.ServiceHost.exe loads the said DLL file via the DLL search order hijacking technique"** clearly maps to the **Technique: DLL search order hijacking** which falls under the **Tactic: Persistence**. In a similar fashion, each statement was broken down to understand the thought behind it and mapped to a technique whenever found necessary.

3. The TTPs encountered by each individual was then cross-checked for removing any redundant/unnecessary tactics & techniques and the TTPs for the entire finished report were finalized.

4. Upon finalising the TTPs, the ATT&CK navigator tool was used to create a layer to store the mapping and the layer was saved into an Excel file named foggyweb.xlsx

Assumption about the Organization:

Our report includes the NOBELIUM threat actor, whose objective is to gain admin-level access to AD FS servers through backdoor, maintain persistence and deepen its infiltration. It belongs to the APT29 group. For this, We have assumed a large-sized company with huge amounts of confidential data to protect and an in-house **Vulnerability Assessment and Penetration Testing(VAPT)** team for the purpose of identifying vulnerabilities in various important assets. The NOBELIUM being a dangerous malware and being the actor behind SUNBURST backdrop, the organization has prioritized the detection of and mitigation against this malware. It has enough budgets for research on FoggyWeb and development of cybersecurity practices to detect this.

With these assumptions, we begin stating the defensive recommendations for the TTPs mapped from the finished report.

Defensive Recommendations with Rationale:

Upon analysing the entire report carefully, a number of TTPs were found to be used by the attacker to achieve his target. Below mentioned are some important mitigation techniques to reduce the severity of any such attack in the future. Also mentioned are a few worthwhile detection techniques that an organization may undertake to detect an attack in the early stages:

- ## Dll Search Order Hijacking-T1574.001

  Mitigations

  1. Use auditing tools capable of detecting DLL search order hijacking opportunities on systems within an enterprise and correct them. Toolkits like the PowerSploit framework contain PowerUp modules that can be used to explore systems for DLL hijacking weaknesses.

  2. Use the program sxstrace.exe that is included with Windows along with manual inspection to check manifest files for side-by-side problems in software.
  3. Adversaries may use new DLLs to execute this technique. Identify and block potentially malicious software executed through search order hijacking by using application control solutions capable of blocking DLLs loaded by legitimate software.

  Detection

  1. Monitor newly constructed .manifest and .local redirection files that do not correlate with software updates.

  2. Monitor for changed made to .manifest/.local redirection files, or file systems for moving, renaming, replacing, or modifying DLLs. Changes in the set of DLLs that are loaded by a process (compared with past behaviour) that do not correlate with known software, patches, etc., are suspicious.

- ## Dll Side Loading-T1574.002

  Mitigations

  1. Whenever possible, include hash values in manifest files to help prevent side-loading of malicious libraries.

  2. Update software regularly to include patches that fix DLL side-loading vulnerabilities.

  Detection

  1. Monitor for newly constructed files in common folders on the computer system.

  2. Monitor for changes made to files for unexpected modifications to access permissions and attributes

  3. Monitor DLL/PE file events, specifically creation of these binary files as well as the loading of DLLs into processes. Look for DLLs that are not recognized or not normally loaded into a process.

- ## Double File Extension-T1036.007

  Mitigations

  1. Disable the default to "hide file extensions for known file types" in Windows OS.

  2. Train users to look for double extensions in filenames, and in general use training as a way to bring awareness to common phishing and spear-phishing techniques and how to raise suspicion for potentially malicious events.

  Detection

  1. Monitor for files written to disk that contain two file extensions, particularly when the second is an executable.

2. Monitor for contextual data about a file, which may include information such as name, the content (ex: signature, headers, or data/media), user/owner, permissions, etc.

- ## Masquerade Task or Service-T1036.004

Mitigation

This type of attack technique cannot be easily mitigated with preventive controls since it is based on the abuse of system features, so some of the detection techniques are-

Detection

1. Monitor executed commands and arguments that may attempt to manipulate the name of a task or service to make it appear legitimate or benign.

2. Monitor for contextual data about a scheduled job, which may include information such as name, timing, command(s), etc.

3. Monitor for changes made to scheduled jobs for unexpected modifications to execution launch

- ## Match Legitimate Name or Location-T1036.005

Mitigations

1. Require signed binaries and images.

2. Use tools that restrict program execution via application control by attributes other than file name for common operating system utilities that are needed.

3. Use file system access controls to protect folders such as C:\Windows\System32.

Detection

1. Collect file hashes; file names that do not match their expected hash are suspect. Perform file monitoring; files with known names but in unusual locations are suspect. Likewise, files that are modified outside of an update or patch are suspect.

2. In containerized environments, use image IDs and layer hashes to compare images instead of relying only on their names. Monitor for the unexpected creation of new resources within your cluster in Kubernetes, especially those created by atypical users.

- ## Rename System Utilities-T1036.003

Mitigation

1. Use file system access controls to protect folders such as C:\Windows\System32.

Detection

1. Monitor executed commands and arguments for actions that could be taken to gather.

2. Collecting and comparing disk and resource filenames for binaries by looking to see if the InternalName, OriginalFilename, and/or ProductName match what is expected could provide useful leads, but may not always be indicative of malicious activity.

3. Monitor for changes made to files for unexpected modifications to file names that are mismatched between the file name on disk and that of the binary's PE metadata. This is a likely indicator that a binary was renamed after it was compiled.

- ## Foggy Web Credentials-T1606

Mitigations

1. Administrators should perform an audit of all access lists and the permissions they have been granted to access web applications and services. This should be done extensively on all resources in order to establish a baseline, followed up on with periodic audits of new or updated resources. Suspicious accounts/credentials should be investigated and removed.
2. Enable advanced auditing on ADFS. Check the success and failure audit options in the ADFS Management snap-in. Enable Audit Application Generated events on the AD FS farm via Group Policy Object.

Detection

1. Monitor for anomalous authentication activity, such as logons or other user session activity associated with unknown accounts. Monitor for unexpected and abnormal access to resources, including access of websites and cloud-based applications by the same user in different locations or by different systems that do not match expected configurations. These logins may occur on any on-premises resources as well as from any cloud environment that trusts the credentials.

2. Monitor for creation of access tokens using SAML tokens which do not have corresponding 4769 and 1200 events in the domain.

3. Monitor for the use of Access Tokens to access services such as Email that were created using SAML tokens which do not have corresponding 1202 events in the domain.

- ## Network Sniffing-T1040

Mitigations

1. Ensure that all wired and/or wireless traffic is encrypted appropriately. Use best practices for authentication protocols, such as Kerberos, and ensure web traffic that may contain credentials is protected by SSL/TLS.

2. Ensure that all wired and/or wireless traffic is encrypted appropriately. Use best practices for authentication protocols, such as Kerberos, and ensure web traffic that may contain credentials is protected by SSL/TLS.

3. Use multi-factor authentication wherever possible.

4. In cloud environments, ensure that users are not granted permissions to create or modify traffic mirrors unless this is explicitly required.

Detection

1. Monitor executed commands and arguments for actions that aid in sniffing network traffic to capture information about an environment, including authentication material passed over the network

2. Monitor for newly executed processes that can aid in sniffing network traffic to capture information about an environment, including authentication material passed over the network

- ## Bash History-T1552.003

  Mitigation

  1. There are multiple methods of preventing a user's command history from being flushed to their .bash_history file, including use of the following commands:`set +o history` and `set -o history` to start logging again;`unset HISTFILE` being added to a user's .bash_rc file; and`ln -s /dev/null ~/.bash_history` to write commands to `/dev/null`instead.

  Detection

  1. While users do typically rely on their history of commands, they often access this history through other utilities like "history" instead of commands like `cat ~/.bash_history`.

  2. Monitoring when the user's `.bash_history` is read can help alert to suspicious activity.

- ## Credentials in Files-T1552.001

  Mitigations

  1. Preemptively search for files containing passwords and take actions to reduce the exposure risk when found.

  2. Establish an organizational policy that prohibits password storage in files.

  3. Restrict file shares to specific directories with access only to necessary users.

  Detection

  1. While detecting adversaries accessing these files may be difficult without knowing they exist in the first place, it may be possible to detect adversary use of credentials they have obtained. Monitor executed commands and arguments of executing processes for suspicious words or regular expressions that may indicate searching for a password (for example: password, pwd, login, secure, or credentials).

  2. Monitor for files being accessed that may search local file systems and remote file shares for files containing insecurely stored credentials. While detecting adversaries accessing these files may be difficult without knowing they exist in the first place, it may be possible to detect adversary use of credentials they have obtained.

- ## Credentials in Registry-T1552.002

  Mitigations

  1. Proactively search for credentials within the Registry and attempt to remediate the risk.

  2. Do not store credentials within the Registry.

  3. If it is necessary that software must store credentials in the Registry, then ensure the associated accounts have limited permissions so they cannot be abused if obtained by an adversary.

Detection

1. Monitor executed commands and arguments that may search the Registry on compromised systems for insecurely stored credentials.

2. Monitor newly executed processes for applications that can be used to query the Registry, such as Reg, and collect command parameters that may indicate credentials are being searched. Correlate activity with related suspicious behaviour that may indicate an active intrusion to reduce false positives.

3. Monitor for unexpected windows registry key being accessed that may search the Registry on compromised systems for insecurely stored credentials.

- ## Private Keys-T1552.004

Mitigation

1. Ensure only authorized keys are allowed access to critical resources and audit access lists regularly.

Detection

1. Monitor executed commands and arguments that may search for private key certificate files on compromised systems for insecurely stored credentials.

2. Monitor access to files and directories related to cryptographic keys and certificates as a means for potentially detecting access patterns that may indicate collection and exfiltration activity.

- ## Web Protocols-T1071.001

Mitigation

1. Network intrusion detection and prevention systems that use network signatures to identify traffic for specific adversary malware can be used to mitigate activity at the network level.

Detection

1. Monitor and analyse traffic patterns and packet inspection associated to protocols, leveraging SSL/TLS inspection for encrypted traffic that do not follow the expected protocol standards and traffic flows (e.g extraneous packets that do not belong to established flows, gratuitous or anomalous traffic patterns, anomalous syntax, or structure). Consider correlation with process monitoring and command line to detect anomalous processes execution and command line arguments associated to traffic patterns (e.g. monitor anomalies in use of files that do not normally initiate connections for respective protocol(s)).

2. Monitor for web traffic to/from known-bad or suspicious domains and analyse traffic flows that do not follow the expected protocol standards and traffic flows (e.g extraneous packets that do not belong to established flows, or gratuitous or anomalous traffic patterns). Consider correlation with process monitoring and command line to detect anomalous processes execution and command line arguments associated to traffic patterns (e.g. monitor anomalies in use of files that do not normally initiate connections for respective protocol(s)).

- ## Asymmetric Cryptography-T1573.002

Mitigations

1. Network intrusion detection and prevention systems that use network signatures to identify traffic for specific adversary malware can be used to mitigate activity at the network level.

2. SSL/TLS inspection can be used to see the contents of encrypted sessions to look for network-based indicators of malware communication protocols.

Detection

1. Monitor and analyse traffic patterns and packet inspection associated to protocols that do not follow the expected protocol standards and traffic flows (e.g extraneous packets that do not belong to established flows, gratuitous or anomalous traffic patterns, anomalous syntax, or structure).Consider correlation with process monitoring and command line to detect anomalous processes execution and command line arguments associated to traffic patterns. (e.g. monitor anomalies in use of files that do not normally initiate connections for respective protocol(s)).

- ## Symmetric Cryptography-T1573.001

Mitigation

1. Network intrusion detection and prevention systems that use network signatures to identify traffic for specific adversary malware can be used to mitigate activity at the network level.

Detection

1. Monitor and analyse traffic patterns and packet inspection associated to protocol(s) that do not follow the expected protocol standards and traffic flows (e.g extraneous packets that do not belong to established flows, gratuitous or anomalous traffic patterns, anomalous syntax, or structure). Consider correlation with process monitoring and command line to detect anomalous processes execution and command line arguments associated to traffic patterns (e.g. monitor anomalies in use of files that do not normally initiate connections for respective protocol(s)).

- ## Ingress Tool Traansfer-T1105

Mitigation

1. Network intrusion detection and prevention systems that use network signatures to identify traffic for specific adversary malware or unusual data transfer over known protocols like FTP can be used to mitigate activity at the network level. Signatures are often for unique indicators within protocols and may be based on the specific obfuscation technique used by a particular adversary or tool, and will likely be different across various malware families and versions. Adversaries will likely change tool C2 signatures over time or construct protocols in such a way as to avoid detection by common defensive tools.

Detection

1. Monitor for file creation and files transferred into the network.

2. Monitor for newly constructed network connections that are sent or received by untrusted hosts or creating files on-system may be suspicious. Use of utilities, such as FTP, that does not normally occur may also be suspicious.

3. Monitor network traffic content for files and other potentially malicious content, especially data coming in from abnormal/unknown domain and IPs.

- ## Deobfuscate/Decode File or Information - T1140

Mitigation

This type of attack technique cannot be easily mitigated with preventive controls since it is based on the abuse of system features.

Detection

1. Monitor for changes made to files for unexpected modifications that attempt to hide artifacts.

2. Monitor for newly executed processes that attempt to hide artifacts of an intrusion, such as common archive file applications and extensions (ex: Zip and RAR archive tools), and correlate with other suspicious behavior to reduce false positives from normal user and administrator behavior.

- ## Native API - T1106

Mitigations

1. On Windows 10, enable Attack Surface Reduction (ASR) rules to prevent Office VBA macros from calling Win32 APIs.

2. Identify and block potentially malicious software that may be executed through this technique by using application control.Tools, like Windows Defender Application Control, AppLocker, or Software Restriction Policies where appropriate.

Detection

1. Monitor DLL/PE file events, specifically creation of these binary files as well as the loading of DLLs into processes. Utilization of the Windows APIs may involve processes loading/accessing system DLLs associated with providing called functions (ex: ntdll.dll, kernel32.dll, advapi32.dll, user32.dll, and gdi32.dll). Monitoring for DLL loads, especially to abnormal/unusual or potentially malicious.

2. Monitor DLL/PE file events, specifically creation of these binary files as well as the loading of DLLs into processes. Utilization of the Windows APIs may involve processes loading/accessing system DLLs associated with providing called functions (ex: ntdll.dll, kernel32.dll, advapi32.dll, user32.dll, and gdi32.dll). Monitoring for DLL loads, especially to abnormal/unusual or potentially malicious processes, may indicate abuse of the Windows API. Though noisy, this data can be combined with other indicators to identify adversary activity.

- ## Shared Modules-T1129

Mitigation

1. Identify and block potentially malicious software executed through this technique by using application control tools capable of preventing unknown DLLs from being loaded.

Detection

1. Monitoring DLL module loads may generate a significant amount of data and may not be directly useful for defense unless collected under specific circumstances, since benign use of Windows modules load functions are common and may be difficult to distinguish from malicious behavior. Legitimate software will likely only need to load routine, bundled DLL modules or Windows system DLLs such that deviation from known module loads may be suspicious. Limiting DLL module loads to %SystemRoot% and %ProgramFiles% directories will protect against module loads from unsafe paths

2. Monitor for API calls that may execute malicious payloads via loading shared modules.

- ## Malicious File-T1204.002

Mitigations

1. On Windows 10, various Attack Surface Reduction (ASR) rules can be enabled to prevent the execution of potentially malicious executable files (such as those that have been downloaded and executed by Office applications/scripting interpreters/email clients or that do not meet specific prevalence, age, or trusted list criteria). Note: cloud-delivered protection must be enabled for certain rules.

2. Application control may be able to prevent the running of executables masquerading as other files.

3. Use user training as a way to bring awareness to common phishing and spear phishing techniques and how to raise suspicion for potentially malicious events

Detection

1. Monitor for newly constructed files that are downloaded and executed on the user's computer. Endpoint sensing or network sensing can potentially detect malicious events once the file is opened (such as a Microsoft Word document or PDF reaching out to the internet or spawning powershell.exe).

2. Monitor for newly constructed processes and/or command-lines for applications that may be used by an adversary to gain initial access that require user interaction. This includes compression applications, such as those zip files that can be used to Deobfuscate/Decode Files or Information in payloads.

- ## Obfuscated file or information - T1027:

Mitigations

1. Consider utilizing the Antimalware Scan Interface (AMSI) on Windows 10 to analyse commands after being processed/interpreted.

2. On Windows 10, enable Attack Surface Reduction (ASR) rules to prevent execution of potentially obfuscated scripts

Detection

1. Monitor executed commands and arguments containing indicators of obfuscation and known suspicious syntax such as uninterpreted escape characters like "^" and """". Deobfuscation tools can be used to detect these indicators in files/payloads.

2. Detection of file obfuscation is difficult unless artifacts are left behind by the obfuscation process that are uniquely detectable with a signature. If detection of the obfuscation itself is not possible, it may be possible to detect the malicious activity that caused the obfuscated file (for example, the method that was used to write, read, or modify the file on the file system)

- ## Reflective code loading - T1620

Mitigation

This type of attack technique cannot be easily mitigated with preventive controls since it is based on the abuse of system features.

Detection

1. Monitor for artifacts of abnormal process execution. For example, a common signature related to reflective code loading on Windows is mechanisms related to the .NET Common Language Runtime (CLR) -- such as mscor.dll, mscoree.dll, and clr.dll -- loading into abnormal processes (such as notepad.exe)

2. Monitor for code artifacts associated with reflectively loading code, such as the abuse of .NET functions such as `Assembly.Load()` and Native API functions such as `CreateThread()`, `memfd_create()`, `execve()`, and/or `execveat()`.

- ## File and Directory Discovery - T1083

Mitigation

This type of attack technique cannot be easily mitigated with preventive controls since it is based on the abuse of system features.

Detection

1. Monitor executed commands and arguments that may enumerate files and directories or may search in specific locations of a host or network share for certain information within a file system.

2. Monitor for API calls that may enumerate files and directories or may search in specific locations of a host or network share for certain information within a file system.

- ## Process Discovery - T1057

Mitigation

This type of attack technique cannot be easily mitigated with preventive controls since it is based on the abuse of system features.

Detection

1. Monitor executed commands and arguments for actions that may attempt to get information about running processes on a system.

2. Monitor for API calls may attempt to get information about running processes on a system.

- **Data from local system - T1005**

Mitigation

Data loss prevention can restrict access to sensitive data and detect sensitive data that is unencrypted.

Detection

1. Monitor executed commands and arguments that may search and collect local system sources, such as file systems or local databases, to find files of interest and sensitive data prior to Exfiltration. Remote access tools with built-in features may interact directly with the Windows API to gather data. Data may also be acquired through Windows system management tools such as Windows Management Instrumentation and PowerShell.

2. Monitor for unexpected/abnormal access to files that may be malicious collection of local data, such as user files (pdf, .docx, .jpg, etc.) or local databases.

- **Archive via Utility-T1560.001**

Mitigation

1. System scans can be performed to identify unauthorized archival utilities.

Detection

1. Monitor executed commands and arguments for actions that will aid in compression or encrypting data that is collected prior to exfiltration, such as tar.

2. Monitor newly constructed files being written with extensions and/or headers associated with compressed or encrypted file types. Detection efforts may focus on follow-on exfiltration activity, where compressed or encrypted files can be detected in transit with a network intrusion detection or data loss prevention system analyzing file headers.

- **Archive via Library-T1560.002**

Mitigation

This type of attack technique cannot be easily mitigated with preventive controls since it is based on the abuse of system features.

Detection

1. Monitor newly constructed files being written with extensions and/or headers associated with compressed or encrypted file types. Detection efforts may focus on follow-on exfiltration activity, where compressed or encrypted files can be detected in transit with a network intrusion detection or data loss prevention system analysing file headers.

2. Monitor for any attempts to enable scripts running on a system would be considered suspicious. If scripts are not commonly used on a system, but enabled, scripts running out of cycle from patching or other administrator functions are suspicious. Scripts should be captured from the file system when possible to determine their actions and intent.

- **Archive via Custom Method- T1560.003**

Mitigation

This type of attack technique cannot be easily mitigated with preventive controls since it is based on the abuse of system features.

Detection

1. Monitor newly constructed files being written with extensions and/or headers associated with compressed or encrypted file types. Detection efforts may focus on follow-on exfiltration activity, where compressed or encrypted files can be detected in transit with a network intrusion detection or data loss prevention system analysing file headers.

2. Monitor for any attempts to enable scripts running on a system would be considered suspicious. If scripts are not commonly used on a system, but enabled, scripts running out of cycle from patching or other administrator functions are suspicious. Scripts should be captured from the file system when possible to determine their actions and intent.

- ## Exfiltration over C2 Channel -T1041

Mitigations

1. Data loss prevention can detect and block sensitive data being sent over unencrypted protocols.

2. Network intrusion detection and prevention systems that use network signatures to identify traffic for specific adversary malware can be used to mitigate activity at the network level. Signatures are often for unique indicators within protocols and may be based on the specific obfuscation technique used by a particular adversary or tool, and will likely be different across various malware families and versions. Adversaries will likely change tool command and control signatures over time or construct protocols in such a way to avoid detection by common defensive tools.

Detection

1. Monitor executed commands and arguments that may steal data by exfiltrating it over an existing command and control channel.

2. Monitor for suspicious files (i.e. .pdf, .docx, .jpg, etc.) viewed in isolation that may steal data by exfiltrating it over an existing command and control channel.

- ## Data Destruction - T1485

Mitigation

1. Consider implementing IT disaster recovery plans that contain procedures for taking regular data backups that can be used to restore organizational data.[42] Ensure backups are stored off system and is protected from common methods adversaries may use to gain access and destroy the backups to prevent recovery.

Detection

1. Monitor for unexpected deletion of a cloud storage infrastructure, such as the DeleteDBCluster and DeleteGlobalCluster events in AWS, or a high quantity of data deletion events, such as DeleteBucket. Many of these events within a short period of time may indicate malicious activity.

2. Monitor executed commands and arguments for binaries that could be involved in data destruction activity, such as SDelete.

References used:

1. Microsoft Threat Intelligence Center. (2021, October 25). NOBELIUM targeting delegated administrative privileges to facilitate broader attacks. Retrieved March 25, 2022.

2. Smith, L., Leathery, J., Read, B. (2021, March 4). New SUNSHUTTLE Second-Stage Backdoor Uncovered Targeting U.S.-Based Entity; Possible Connection to UNC2452. Retrieved March 12, 2021.

3. ANSSI. (2021, December 6). PHISHING CAMPAIGNS BY THE NOBELIUM INTRUSION SET. Retrieved April 13, 2022.

4. Ramin Nafisi. (2021, September 27). FoggyWeb: Targeted NOBELIUM malware leads to persistent backdoor. Retrieved October 4, 2021.

5. Microsoft Defender Research Team. (2018, December 3). Analysis of cyberattack on U.S. think tanks, non-profits, public sector by unidentified attackers. Retrieved April 15, 2019.

6. Microsoft Threat Intelligence Center (MSTIC). (2021, May 27). New sophisticated email-based attack from NOBELIUM. Retrieved May 28, 2021.

7. https://attack.mitre.org/