Analysis and Defense Recommendation on "North Korea's Lazarus APT leverages Windows Update client, GitHub in latest campaign : Malwarebytes"

1 Approach to Mapping Finished Reporting to ATT&CK

The snippet from the document is:

The two macro-embedded documents seem to be luring the targets[1] about new job opportunities at Lockheed Martin:

- `Lockheed_Martin_JobOpportunities.docx`[2]
- `Salary_Lockheed_Martin_job_opportunities_confidential.doc`[3]

...

The attack starts by executing the malicious macros that are embedded in the Word document[4]. The malware performs a series of injections and achieves startup persistence in the target system. In the next section we will provide technical details about various stages of this attack and its payload capabilities.

The above code uses a very unusual and lesser known technique to hijack the control flow and execute malicious code. The malware retrieves the address of the "WMIsAvailableOffline" function from "wmvcore.dll"[5], then it changes the memory protection permissions for code in "WMIsAvailableOffline" and proceeds to overwrite the code in memory[6] with the malicious base64 decoded shell-code.

Reference No.	ID	Tactic	Technique	Sub Technique	Explanation
1	`T1204.002`	Execution	User Execution	Malicious File	The targets or the users can only initialize this execution by downloading and opening the malicious document here. Here the adversary relies upon some specific actions by the user for execution.
2, 3	`T1566.001`	Initial Access	Phishing	Spearphishing Attachment	Here the attachments may have been sent using e-mail or some other medium to the targeted victims who are looking for job opportunities at Lockheed Martin.
4	`T1059.005`	Execution	Command and scripting interpreter	Visual Basic	Adversary has used macros embedded inside the word document which is executed by opening Microsoft Office. VBA enables documents to contain macros used to automate the execution of tasks and other functionality on the host. Adversary here has exploited this functionality of MS Office.

Reference No.	ID	Tactic	Technique	Sub Technique	Explanation
5	T1057	Discovery	Process Discovery	-	Here the adversary is trying to get information about WMIsAvailableOffline function exported by the "wmvcore.dll" file. This can be done by using the GetProcAddress function.
6	T1055	Defense Evasion	Process Injection	-	Here the adversary is overwriting the WMIsAvailableOffline function in the memory with malicious shell code. So the adversary is injecting the shell code inside the address space of the function. The adversary can use "Process Doppelgänging" or "Extra Window Memory Injection" sub-techniques to inject shell code into the memory. The details are not specified here.

The snippet from the document is:

Another interesting thing happening in the above code is the control flow hijacking through the KernelCallbackTable[7] member of the PEB. A call to NtQueryInformationProcess is made[8] with ProcessBasicInformation class as the parameter which helps the malware to retrieve the address of PEB and thus retrieving the KernelCallbackTable pointer[9]. KernelCallbackTable is initialized to an array of callback functions when user32.dll is loaded into memory, which are used whenever a graphical call (GDI) is made by the process. To hijack the control flow, malware replaces the USER32!_fnDWORD callback in the table with the malicious WMIsAvailableOffline function[10]. Once the flow is hijacked and malicious code is executed the rest of the code takes care of restoring the KernelCallbackTable to its original state[11].

Reference No.	ID	Tactic	Technique	Sub Technique	Explanation
7, 10	T1574.013	Persistence, Privilege Escalation, Defense Evasion	Hijack Execution Flow	KernelCallback-Table	The adversary specifically uses the Hijack Execution Flow technique with KernelCallbackTable as a sub-technique. After getting the KernelCallbackTable pointer, the adversary replaces USER32!_fnDWORD function with malicious WMIsAvailableOffline function which was modified earlier.
8	T1106	Execution	Native API	-	The function NtQueryInformationProcess is a Native API function in windows. The adversary uses this function to retrieve the pointer to the KernelCallbackTable.
9	T1057	Discovery	Process Discovery	-	The adversary uses windows native function to retrieve the pointer to the KernelCallbackTable. Here the adversary is trying to get some information about a particular process.
11	T1070	Defense Evasion	Indicator Removal on Host	-	As a measure of defense evasion, the adversary here has used the malicious code in such a way that it would put back the KernelCallbackTable back to its original state.

The snippet from the document is:

The shellcode loaded by the macro contains an encrypted DLL[12] which is decrypted at runtime and then manually mapped into memory by the shellcode[13]. After mapping the DLL, the shellcode jumps to the entry point of that DLL. The shellcode uses some kind of custom hashing method to resolve the APIs. We used hollows_hunter to dump the DLL and reconstruct the IAT once it is fully mapped into memory.

...

The injection function is responsible for resolving all the required API calls. It then opens a handle to the target process by using the OpenProcess API[14]. It uses the SizeOfImage field in the NT header of the DLL to be injected into allocated space into the target process along with a separate space for the init_dll function. The purpose of the init_dll function is to initialize the injected DLL and then pass the control flow to the entry point of the DLL. One thing to note here is a simple CreateRemoteThread method is used to start a thread inside the target process[15] unlike the KernelCallbackTable technique used in our macro.

...

stage1_winword.dll – This is the DLL which is mapped inside the Word process. This DLL is responsible for restoring the original state of KernelCallbackTable and then injecting stage2_explorer.dll into the explorer.exe process[16].

Reference No.	ID	Tactic	Technique	Sub Technique	Explanation
12	`T1027`	Defense Evasion	Obfuscated Files or Information	-	The adversary has made sure that the files in the disk cannot be detected by any AVs. The malicious DLLs are embedded in the shell code in encrypted form.
13	`T1140`	Defense Evasion	Deobfuscate/Decode Files or Information	-	To avoid detection the adversary has decrypted the encrypted DLLs in the runtime. The decrypted codes are never saved in the disk, so the AVs are not going to detect the malicious codes.
14	`T1106`	Execution	Native API	-	The adversary here uses the OpenProcess API of windows system.
15	`T1055.001`	Defense Evasion	Process Injection	Dynamic-link Library Injection	The adversary here uses CreateRemoteThread to start thread inside the target process and then injects the DLL file into the allocated space of the target process along with a function which initializes the DLL and pass the control flow to the entry point of the DLL.
16	`T1055.001`	Defense Evasion	Process Injection	Dynamic-link Library Injection	Here stage1_winword.dll (DLL) injected inside word process is responsible for injecting stage2_explorer.dll (DLL) into explorer.exe process.

The snippet from the document is:

The above code snippet shows the main routine of stage2_explorer.dll. As you can see it checks for the existence of `C:\Windows\system32\wuaueng.dll`[17] and then if it doesn't exist it takes its path to drop additional files. It executes the drops_lnk.dll in the current process and then tries to create the RuntimeBroker process[18] and if successful in creating RuntimeBroker, it injects stage3_runtimebroker.dll into the newly created process. If for some reason process creation fails, it just executes stage3_runtimebroker.dll in the current explorer.exe process[19].

. . .

drops_lnk.dll – This DLL is loaded and executed inside the explorer.exe process, it mainly drops the lnk file (WindowsUpdateConf.lnk) into the startup folder[20] and then it checks for the existence of wuaueng.dll[21] in the malicious directory and manually loads and executes it from the disk if it exists. The lnk file (WindowsUpdateConf.lnk) executes
`C:\Windows\system32\wuauclt.exe /UpdateDeploymentProvider`
`C:\Windows\system32\wuaueng.dll /RunHandlerComServer`[22]
This is an interesting technique used by Lazarus to run its malicious DLL using the Windows Update Client to bypass security detection mechanisms[23]. With this method, the threat actor can execute its malicious code through the Microsoft Windows Update client by passing the following arguments: /UpdateDeploymentProvider, Path to malicious dll and /RunHandlerComServer argument after the dll.

Reference No.	ID	Tactic	Technique	Sub Technique	Explanation
17	`T1083`	Discovery	File and Directory Discovery	-	Here the adversary is trying to check if the file `C:\Windows\system32\wuaueng.dll` exist in the disk or not.
18	`T1543`	Persistence	Create or Modify System Process	-	The stage2_explorer.dll file tries to create a process named RuntimeBroker from explorer.exe process.
19	`T1055.001`	Defense Evasion	Process Injection	Dynamic-link Library Injection	Here the stage2_explorer.dll injects the stage3_runtimebroker.dll into the process RuntiemBroker, if it is successfully created. Else the DLL is injected in the current process.
20	`T1547.001`	Persistence	Boot or Logon Autostart Execution	Registry Run Keys / Startup Folder	The adversary achieve persistence by droping the WindowsUpdateConf.lnk file in the startup folder.
21	`T1083`	Discovery	File or Directory Discovery	-	The malicious code looks for the file wuaueng.dll file in the malicious directory.
22	`T1106`	Execution	Native API	-	The WindowsUpdateConf.lnk file executes the wuauclt.exe and wuaueng.dll files.
23	`T1218`	Defense Evasion	System Binary Proxy Execution	-	To bypass the security detection the adversary here has used the windows update client to execute the malicious code.

The snippet from the document is:

stage3_runtimebroker.dll – This DLL is responsible for creating the malicious directory (“`C:\Windows\system32\`”) and then drops the wuaueng.dll in that directory, furthermore it sets the attributes of the directory to make it hidden[24].

. . .

wuaueng.dll –
. . .
This DLL has embedded inside another DLL which contains the core module (core_module.dll) of this malware responsible for communicating with the Command and Control (C2) server. This DLL can be loaded into memory in two ways:

- If drops_lnk.dll loads this DLL into explorer.exe then it loads the core_module.dll and then executes it[25]
- If it is being executed from wuauclt.exe, then it retrieves the PID of explorer.exe and injects the core_module.dll into that process.[26]

. . .

get_module_from_repo uses the hardcoded username, repo_name, directory, token to make a http request to GitHub and retrieves the files present in the “images” directory of the repository[27].

The HTTP request retrieves contents of the files present in the repository with an interesting validation which checks that the retrieved file is a PNG. The file that was earlier retrieved was named “readme.png”; this PNG file has one of the malicious modules embedded in it[28].

Reference No.	ID	Tactic	Technique	Sub Technique	Explanation
24	`T1564.001`	Defense Evasion	Hide Artifacts	Hidden Files and Directories	attacker creates a malicious directory ”`C:\Windows\system32\`” and drops the file wuaueng.dll in there. The attribute of the directory is made hidden as a measure of defense evasion.
25, 26	`T1055.001`	Defense Evasion	Process Injection	Dynamic-link Library Injection	The adversary uses two ways : • drops_lnk.dll injects wuaueng.dll into explorer.exe, which loads core_module.dll in explorer.exe to execute. • drops_lnk.dll loads wuaueng.dll into wuauclt.exe, which injects core_module.dll in explorer.exe to execute.
27	`T1071.001`	Command and Control	Application Layer Protocol	Web Protocols	The Adversary here uses http request to Github and retrieves the files in ”images” directory.
28	`T1036`	Defense Evasion	Masquerading	-	The file retrieved from the Github repo is named as ”readme.png” which contains malicious modules embedded in it.

The snippet from the document is:

It then executes GetNumberOfMethods and saves the result obtained by the module[29]. This result is committed to the remote repo under the metafiles directory[30] with a filename denoting the time at which the module was executed. This file committed to the repo contains the result of the commands executed by the module on the target system. To commit the file the malware makes a PUT HTTP request to Github[31].

. . .

Coming back to this module, it has very limited capabilities. It retrieves the Username, ComputerName and a list of all the running processes on the system[32] and then returns the result so it can be committed to the C2.

. . .

The major difference in this document is that it tries to retrieve a remote HTML page and then executes it using mshta.exe[33]. The remote HTML page is located at `https[:]//markettrendingcenter[.]com/member.htm` and throws a 404 Not Found which makes it difficult for us to analyze this document any further.

. . .

Using job opportunities as template is the known method used by Lazarus to target its victims. The documents created by this actor are well designed and contain a large icon for a known company such as LockHeed Martin, BAE Systems, Boeing and Northrop Grumman in the template[34].

Reference No.	ID	Tactic	Technique	Sub Technique	Explanation
29	`T1119`	Collection	Automated collection	-	Adversary here uses the function GetNumberOfMethods obtained from the malicious module to collect data from the user's machine and save it in the disk, which will be exfiltrated later.
30	`T1041`	Exfiltration	Exfiltration Over C2 Channel	-	Here the adversary uses GitHub repo as command and control. After collecting data from the victim the adversary commits the data to the GitHub repo.
31	`T1567.001`	Exfiltration	Exfiltration Over Web Service	Exfiltration to Code Repository	Here the adversary is using PUT HTTPS request to GitHub to exfiltrate the data to the repository.
32	`T1087.001`	Discovery	Account Discovery	Local Account	The malicious DLL, GetbaseInfo.dll retrieves Username, ComputerName on the system.
	`T1057`	Discovery	Process Discovery	-	The malicious DLL, GetbaseInfo.dll retrieves a list of all running processes on the system.
33	`T1071.001`	Command and Control	Application Layer Protocol	Web Protocols	Here the adversary tries to retrieve a HTML page.
	`T1218.005`	Defense Evasion	System Binary Proxy Execution	Mshta	The adversary here tries to retrieve HTML page and then executes it with mshta.exe
34	`T1591.004`	Reconnaissance	Gather Victim Org Information	Identify Roles	During targeting the adversary gathers information about the victims' job preference, also information about Lock-Heed Martin company.

2 Defensive recommendations

The attack analysis that has been given in the document, is about North Korea's Lazarus APT group. In this section, we have compared the techniques that we've found in the analysis with the techniques used by the actual Lazarus APT group in the past. We have used the attack navigator tool at `https://mitre-attack.github.io/attack-navigator/`, for this comparison.

Techniques Identified : The common techniques that have been identified are as follows

1. Dynamic-Link Library Injection (`T1055.001`)
2. Native API (`T1106`)
3. Process Discovery (`T1057`)
4. Kernel Callback Table (T1574.013)
5. Web Protocols (`T1071.001`)
6. File and Directory Discovery (`T1083`)
7. Identify Roles (`T1591.004`)
8. Spearphishing Attachment (`T1566.001`)
9. Visual Basic (`T1059.005`)
10. Malicious File (`T1204.002`)
11. Registry Run Keys/Startup Folder (`T1547.001`)
12. Deobfuscate/Decode Files or Information (`T1140`)
13. Hidden Files and Directories (`T1564.001`)
14. Indicator Removal on Host (`T1070`)
15. Masquerading (`T1036`)
16. Obfuscate Files or Information (`T1027`)
17. System Binary Proxy Execution (`T1218`)
18. Mshta (`T1218.005`)
19. Exfiltration Over C2 (`T1041`)

The above list contains all the techniques that have been identified in the document and also were used by the Lazarus APT group in the past. We have prioritized the list according to the number of times the techniques have been invoked in the attack method explained in the report.

2.1 Dynamic-Link Library Injection (`T1055.001`)

First, we will look back at the report and try to figure out how the technique `T1055.001` is used for the attack.

2.1.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	15*	The adversary here uses CreateRemoteThread to start thread inside the target process and then injects the DLL file into the allocated space of the target process along with a function which initializes the DLL and pass the control flow to the entry point of the DLL.
2	16	Here stage1_winword.dll (DLL) injected inside word process is responsible for injecting stage2_explorer.dll (DLL) into explorer.exe process.

*The reference number refers back to the text in the document that we've marked in the previous section

Sr. No.	Reference No.	Explanation
3	19	Here the stage2_explorer.dll injects the stage3_runtimebroker.dll into the process RuntimeBroker, if it is successfully created. Else the DLL is injected in the current process.
4	25, 26	The adversary uses two ways : • drops_lnk.dll injects wuaueng.dll into explorer.exe, which loads core_module.dll in explorer.exe to execute. • drops_lnk.dll loads wuaueng.dll into wuauclt.exe, which injects core_module.dll in explorer.exe to execute.

2.1.2 Defensive options for "Dynamic-Link Library Injection" technique

Detections The following table contains detection methods to detect the technique. We've used the resources available at [2]

Data Source.	Data Component	Detects
Module	Module Load	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.
Process	OS API Execution	Monitoring Windows API calls indicative of the various types of code injection may generate a significant amount of data and may not be directly useful for defense unless collected under specific circumstances for known bad sequences of calls, since benign use of API functions may be common and difficult to distinguish from malicious behavior. Windows API calls such as CreateRemoteThread and those that can be used to modify memory within another process, such as VirtualAllocEx/WriteProcessMemory, may be used for this technique.[11]
	Process Access	Monitor for process being viewed that may inject dynamic-link libraries (DLLs) into processes in order to evade process-based defenses as well as possibly elevate privileges.
	Process Modification	Monitor for changes made to processes that may inject dynamic-link libraries (DLLs) into processes in order to evade process-based defenses as well as possibly elevate privileges.

Mitigations The following table contains the possible mitigations against the attack technique.

Mitigation	Description
Behavior Prevention on Endpoint	Some endpoint security solutions can be configured to block some types of process injection based on common sequences of behavior that occur during the injection process.

2.1.3 Recommendations

The following are the recommendations for the user to protect themselves against "Dynamic-Link Library Injection" technique.

- The user must make sure that their access privileges must not be as same as administrators.
- The user must password protect the administrator account.

2.2 Native API (`T1106`)

2.2.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	8	The function NtQueryInformationProcess is a Native API function in windows. The adversary uses this function to retrieve the pointer to the KernelCallbackTable.

Sr. No.	Reference No.	Explanation
2	14	The adversary here uses the OpenProcess API of windows system.
3	22	The WindowsUpdateConf.lnk file executes the wuauclt.exe and wuaueng.dll files.

2.2.2 Defensive options for "Native API" technique

Detections The following table contains detection methods to detect the technique.

Data Source.	Data Component	Detects
Module	Module Load	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.
Process	OS API Execution	Monitoring API calls may generate a significant amount of data and may not be useful for defense unless collected under specific circumstances, since benign use of API functions are common and may be difficult to distinguish from malicious behavior. Correlation of other events with behavior surrounding API function calls using API monitoring will provide additional context to an event that may assist in determining if it is due to malicious behavior. Correlation of activity by process lineage by process ID may be sufficient.

Mitigations The following table contains the possible mitigations against the attack technique.

Mitigation	Description
Behavior Prevention on Endpoint	On Windows 10, enable Attack Surface Reduction (ASR) rules to prevent Office VBA macros from calling Win32 APIs. [12]
Execution Prevention	Identify and block potentially malicious software executed that may be executed through this technique by using application control [3] tools, like Windows Defender Application Control [9], AppLocker, [17] or Software Restriction Policies [5] where appropriate. [6]

2.2.3 Recommendations

We're recommending the user the following to defend its system against the "Native API" technique.

- If the user has a Windows 10 system or higher, then it can use the mitigation method described above.
- The user can use different tools mentioned above to identify and block potentially malicious software.

2.3 Process Discovery (T1057)

2.3.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	5	Here the adversary is trying to get information about WMIsAvailableOffline function exported by the "wmvcore.dll" file. This can be done by using the GetProcAddress function.
2	9	The adversary uses windows native function to retrieve the pointer to the KernelCallbackTable. Here the adversary is trying to get some information about a particular process.
3	32	The malicious DLL, GetbaseInfo.dll retrieves a list of all running processes on the system.

2.3.2 Defensive options for "Process Discovery" technique

Detections

Data Source.	Data Component	Detects
Command	Command and Execution	Monitor executed commands and arguments for actions that may attempt to get information about running processes on a system.
Process	OS API Execution	Monitor for API calls may attempt to get information about running processes on a system.
	Process Creation	Monitor for newly executed processes that may attempt to get information about running processes on a system.

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

2.3.3 Recommendations

The user doesn't have many options here. As this is a system feature that adversaries may abuse. The user may ignore the severity of this technique.

2.4 Kernel Callback Table (`T1574.013`)

2.4.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	7,10	The adversary specifically uses the Hijack Execution Flow technique with KernelCallbackTable as a sub-technique. After getting the KernelCallbackTable pointer, the adversary replaces USER32!_fnDWORD function with malicious WMIsAvailableOffline function which was modified earlier.

2.4.2 Defensive options for "Kernel Callback Table" technique

Detections

Data Source.	Data Component	Detects
Process	OS API Execution	Monitoring Windows API calls indicative of the various types of code injection may generate a significant amount of data and may not be directly useful for defense unless collected under specific circumstances. for known bad sequence of calls, since benign use of API functions may be common and difficult to distinguish from malicious behavior. Windows API calls such as WriteProcessMemory() and NtQueryInformationProcess() with the parameter set to ProcessBasicInformation may be used for this technique [4].

Mitigations

Mitigation	Description
Behavior Prevention on Endpoint	Some endpoint security solutions can be configured to block some types of behaviors related to process injection/memory tampering based on common sequences of indicators (ex: execution of specific API functions).

2.4.3 Recommendations

It is recommended that all Windows users keep their devices up-to-date with the latest security updates. This will reduce the risk of a possible system compromise by helping prevent malicious actors from leveraging the discovered vulnerabilities [14].

2.5 Web Protocols (`T1071.001`)

2.5.1 The process of the technique being used in the report

Detections

Sr. No.	Reference No.	Explanation
1	27	The Adversary here uses http request to Github and retrieves the files in "images" directory.
2	33	Here the adversary tries to retrieve a HTML page.

2.5.2 Defensive options for "Web Protocols" technique

Detection Abuse of standard application protocols can be difficult to detect as many legitimate mobile applications leverage such protocols for language-specific APIs. Enterprises may be better served focusing on detection at other stages of adversarial behavior.

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

2.5.3 Recommendations

Since the adversary is using common web protocols, it is not easy to detect the actions done by the adversary. The user can ignore this technique as he/she doesn't have much to do.

2.6 File and Directory Discovery (`T1083`)

2.6.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	17	Here the adversary is trying to check if the file `C:\Windows\system32\wuaueng.dll` exist in the disk or not.
2	21	The malicious code looks for the file wuaueng.dll file in the malicious directory.

2.6.2 Defensive options for "File and Directory Discovery" technique

Detections The following table contains detection methods to detect the technique.

Data Source.	Data Component	Detects
Command	Command Execution	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.
Process	OS API Execution	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 Creation	Monitor newly executed processes 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.

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

2.6.3 Recommendations

The user doesn't have many options here. As this is a system feature that adversaries may abuse. The user may ignore the severity of this technique.

2.7 Identify Roles (`T1591.004`)

2.7.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	34	During targeting the adversary gathers information about the victims' job preference, also information about Lock-Heed Martin company.

2.7.2 Defensive options for "Identify Roles" technique

Detections Much of this activity may have a very high occurrence and associated false positive rate, as well as potentially taking place outside the visibility of the target organization, making detection difficult for defenders.

Detection efforts may be focused on related stages of the adversary lifecycle, such as during Initial Access.

Mitigations

Mitigation	Description
Pre compromise	This technique cannot be easily mitigated with preventive controls since it is based on behaviors performed outside of the scope of enterprise defenses and controls. Efforts should focus on minimizing the amount and sensitivity of data available to external parties.

2.7.3 Recommendations

This is not an attack technique on the user side. So the User doesn't need to worry about this technique.

2.8 Spearphishing Attachment (`T1566.001`)

2.8.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	2,3	Here the attachments may have been sent using e-mail or some other medium to the targeted victims who are looking for job opportunities at Lockheed Martin.

2.8.2 Defensive options for "Spearphishing Attachment" technique

Detections The following table contains detection methods to detect the technique.

Data Source.	Data Component	Detects
Application Log	Application Log Content	Events collected by third-party services such as mail servers, web applications, or other appliances (not by the native OS or platform).
Network Traffic	Network Traffic Content	Data transmitted across a network (ex: Web, DNS, Mail, File, etc.), that is either summarized (ex: Netflow) and/or captured as raw data in an analyzable format (ex: PCAP).

Mitigations

Mitigation	Description
Antivirus/Antimalware	Deploy anti-virus on all systems that support external email.
Network Intrusion Prevention	Network intrusion prevention systems and systems designed to scan and remove malicious email attachments can be used to block activity.
Restrict Web-Based Content	Consider restricting access to email within critical process environments. Additionally, downloads and attachments may be disabled if email is still necessary.
User Training	Users can be trained to identify social engineering techniques and spearphishing emails.

2.8.3 Recommendations

Email gateways provide an easy and comprehensive way to filter received emails—effectively in real-time based on filenames, email size, sender information, subject lines, text within the email, and several other parameters. Email gateways can be tuned with rules that quarantine, delete, or forward potentially suspicious email messages—based on any or all of the attributes above [7].

Also, the user can use the anti-viruses to detect any malicious macros in the downloaded attachments.

2.9 Visual Basic (`T1059.005`)

First, we will look back at the report and try to figure out how the technique `T1059.005` is used for the attack.

2.9.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	4	Adversary has used macros embedded inside the word document which is executed by opening Microsoft Office. VBA enables documents to contain macros used to automate the execution of tasks and other functionality on the host. Adversary here has exploited this functionality of MS Office.

2.9.2 Defensive options for "Visual Basic" technique

Detections The following table contains detection methods to detect the technique. We've used the resources available at [1]

Data Source.	Data Component	Detects
Command	Command Execution	Monitor executed commands and arguments that may abuse Visual Basic (VB) for execution.
Module	Module Load	Monitor for the loading of modules associated with VB languages (ex: vbscript.dll).
Process	Process Creation	Monitor for events associated with VB execution, such as Office applications spawning processes, usage of the Windows Script Host (typically cscript.exe or wscript.exe). VB execution is likely to perform actions with various effects on a system that may generate events, depending on the types of monitoring used.
Script	Script Execution	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.

Mitigations The following table contains the possible mitigations against the attack technique.

Mitigation	Description
Antivirus/Antimalware	Anti-virus can be used to automatically quarantine suspicious files.
Behavior Prevention on Endpoint	On Windows 10, enable Attack Surface Reduction (ASR) rules to prevent Visual Basic scripts from executing potentially malicious downloaded content.
Disable or Remove Feature or Program	Turn off or restrict access to unneeded VB components.
Execution Prevention	Use application control where appropriate. VBA macros obtained from the Internet, based on the file's Mark of the Web (MOTW) attribute, may be blocked from executing in Office applications (ex: Access, Excel, PowerPoint, Visio, and Word) by default starting in Windows Version 2203.
Restrict Web-Based Content	Script blocking extensions can help prevent the execution of scripts and HTA files that may commonly be used during the exploitation process. For malicious code served up through ads, adblockers can help prevent that code from executing in the first place.

2.9.3 Recommendations

- On Windows 10, enable Attack Surface Reduction (ASR) rules to prevent Visual Basic scripts from executing potentially malicious downloaded content.
- Turn off or restrict access to unneeded VB components.
- The user can use Script blocking extensions can help prevent the execution of scripts and HTA files.

2.10 Malicious File (`T1204.002`)

First, we will look back at the report and try to figure out how the technique `T1204.002` is used for the attack.

2.10.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	1	The targets or the users can only initialize this execution by downloading and opening the malicious document here. Here the adversary relies upon some specific actions by the user for execution.

2.10.2 Defensive options for "Malicious File" technique

Detection

Data Source.	Data Component	Detects
File	File Creation	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).
Process	Process Creation	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 for zip files, that can be used to Deobfuscate/Decode Files or Information in payloads.

Mitigation

Mitigation	Description
Behavior Prevention on Endpoint	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.
Execution Prevention	Application control may be able to prevent the running of executables masquerading as other files.
User Training	Use user training as a way to bring awareness to common phishing and spearphishing techniques and how to raise suspicion for potentially malicious events [12].

2.10.3 Recommendations

- On Windows 10, enable Attack Surface Reduction (ASR) rules to prevent Visual Basic scripts from executing potentially malicious downloaded content.
- If the user is not completely aware of the files that they are dealing with, then the user should use antivirus to automatically detect and quarantine the files.

2.11 Registry Run Keys/Startup Folder (`T1547.001`)

2.11.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	20	The adversary achieve persistence by droping the WindowsUpdateConf.lnk file in the startup folder

2.11.2 Defensive options for "Registry Run Keys/Startup Folder" technique

Detection

Data Source.	Data Component	Detects
Command	Command Excution	Monitor executed commands and arguments that may achieve persistence by adding a program to a startup folder or referencing it with a Registry run key.

Data Source.	Data Component	Detects
File	File Modification	Monitor the start folder for additions or changes. Tools such as Sysinternals Autoruns may also be used to detect system changes that could be attempted at persistence, including the startup folders [15].

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

2.11.3 Recommendations

The user can use tools such as Sysinternals Autorun to detect system changes that could be attempted at persistence, including the startup folders.

2.12 Deobfuscate/Decode Files or Information (`T1140`)

2.12.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	13	To avoid detection the adversary has decrypted the encrypted DLLs in the runtime. The decrypted codes are never saved in the disk, so the AVs are not going to detect the malicious codes.

2.12.2 Defensive options for "Deobfuscate/Decode Files or Information " technique

Detection

Data Source.	Data Component	Detects
File	File Modification	Monitor for changes made to files for unexpected modifications that attempt to hide artifacts.
Process	Process Creation	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.
Script	Script Execution	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.

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

2.12.3 Recommendations

As the adversary uses decryption to decrypt files in the runtime, the user can only view or detect the files at the process level. This may not be an easy task for the user. This attack technique may be ignored here.

2.13 Hidden Files and Directories (`T1564.001`)

2.13.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	24	attacker creates a malicious directory "`C:\Windows\system32\`" and drops the file wuaueng.dll in there. The attribute of the directory is made hidden as a measure of defense evasion.

2.13.2 Defensive options for “Hidden Files and Directories ” technique

Detection

Data Source.	Data Component	Detects
Command	Command Execution	Monitor the file system and shell commands for files being created with a leading ”.” and the Windows command-line use of attrib.exe to add the hidden attribute.
File	File Creation	Monitor the file system and shell commands for files being created with a leading
	File Metadata	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 may set files and directories to be hidden to evade detection mechanism
Process	Process Creation	Monitor newly executed processes that may set files and directories to be hidden to evade detection mechanisms.

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

2.13.3 Recommendations

The user should monitor all newly created directories/files with the hidden attribute, for any malicious files.

2.14 Indicator Removal on Host (T1070)

2.14.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	11	As a measure of defense evasion, the adversary here has used the malicious code in such a way that it would put back the KernelCallbackTable back to its original state.

2.14.2 Defensive options for “Indicator Removal on Host ” technique

Detection

Data Source.	Data Component	Detects
Command	Command Execution	Monitor executed commands and arguments that may delete or alter generated artifacts on a host system, including logs or captured files such as quarantined malware.
Process	Process OS API Executionnt	Monitor for API calls that may delete or alter generated artifacts on a host system, including logs or captured files such as quarantined malware.
	Process Creation	Monitor for newly executed processes that may delete or alter generated artifacts on a host system, including logs or captured files such as quarantined malware.

Mitigation The following table contains the possible mitigations against the attack technique.

Mitigation	Description
Behavior Prevention on Endpoint	Encrypt Sensitive Information Obfuscate/encrypt event files locally and in transit to avoid giving feedback to an adversary.
Remote Data Storage	Automatically forward events to a log server or data repository to prevent conditions in which the adversary can locate and manipulate data on the local system. When possible, minimize time delay on event reporting to avoid prolonged storage on the local system.

Mitigation	Description
Restrict File and Directory Permissions	Protect generated event files that are stored locally with proper permissions and authentication and limit opportunities for adversaries to increase privileges by preventing Privilege Escalation opportunities.

2.14.3 Recommendations

Protect generated event files that are stored locally with proper permissions and authentication and limit opportunities for adversaries to increase privileges by preventing Privilege Escalation opportunities.

2.15 Masquerading (`T1036`)

2.15.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	28	The file retrieved from the Github repo is named ”readme.png” which contains malicious modules embedded in it

2.15.2 Defensive options for “Masquerading” technique

Detections

Data Source.	Data Component	Detects
File	File Metadata	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. Look for indications of common characters that may indicate an attempt to trick users into misidentifying the file type, such as a space as the last character of a file name or the right-to-left override characters”2̌02E”, ”[U+202E]”, and ”%E2%80%AE”.
	File Modification	Monitor for changes made to files outside of an update or patch that may attempt to manipulate features of their artifacts to make them appear legitimate or benign to users and/or security tools.

Mitigations

Mitigation	Description
Code Signing	Require signed binaries.
Execution Prevention	Use tools that restrict program execution via application control by attributes other than file name for common operating system utilities that are needed.
Restrict File and Directory Permissions	Use file system access controls to protect folders such as C:\Windows\System32.

2.15.3 Recommendations

- The user must restrict themselves to execute signed binaries.
- The user can use tools that restrict program execution via application control by attributes other than file name for common operating system utilities that are needed.

2.16 Obfuscate Files or Information (`T1027`)

2.16.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	12	The adversary has made sure that the files in the disk cannot be detected by any AVs. The malicious DLLs are embedded in the shell code in encrypted form.

2.16.2 Defensive options for "Obfuscate Files or Information" technique

Detections

Data Source.	Data Component	Detects
File	File Creation	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).
	File Metadata	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.

Mitigations

Mitigation	Description
Antivirus/Antimalware	Consider utilizing the Antimalware Scan Interface (AMSI) on Windows 10 to analyze commands after being processed/interpreted [16].
Behavior Prevention on Endpoint	On Windows 10, enable Attack Surface Reduction (ASR) rules to prevent execution of potentially obfuscated scripts [13].

2.16.3 Recommendations

- The user can use anti-viruses to detect and remove potentially harmful obfuscated files.
- On Windows 10, user can enable Attack Surface Reduction (ASR) rules to prevent execution of potentially obfuscated scripts.

2.17 System Binary Proxy Execution (T1218)

2.17.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	23	To bypass the security detection the adversary here has used the windows update client to execute the malicious code.

2.17.2 Defensive options for "System Binary Proxy Execution" technique

Detections

Data Source.	Data Component	Detects
Command	Command Execution	Monitor executed commands and arguments that may forge credential materials that can be used to gain access to web applications or Internet services.
Process	OS API Execution	Monitor for API calls that may forge credential materials that can be used to gain access to web applications or Internet services.
	Process Creation	Monitor processes and command-line parameters for signed binaries that may be used to proxy execution of malicious files. Compare recent invocations of signed binaries that may be used to proxy execution with prior history of known good arguments and loaded files to determine anomalous and potentially adversarial activity. Legitimate programs used in suspicious ways, like msiexec.exe downloading an MSI file from the Internet, may be indicative of an intrusion. Correlate activity with other suspicious behavior to reduce false positives that may be due to normal benign use by users and administrators.

Mitigations

Mitigation	Description
Exploit Protection	Microsoft's Enhanced Mitigation Experience Toolkit (EMET) Attack Surface Reduction (ASR) feature can be used to block methods of using using trusted binaries to bypass application control.
Privileged Account Management	Restrict execution of particularly vulnerable binaries to privileged accounts or groups that need to use it to lessen the opportunities for malicious usage.
Execution Prevention	Consider using application control to prevent execution of binaries that are susceptible to abuse and not required for a given system or network.

2.17.3 Recommendations

Microsoft's Enhanced Mitigation Experience Toolkit (EMET) Attack Surface Reduction (ASR) feature can be used to block methods of using using trusted binaries to bypass application control.

Restrict execution of particularly vulnerable binaries to privileged accounts or groups that need to use it to lessen the opportunities for malicious usage.

2.18 Mshta (`T1218.005`)

2.18.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	33	The adversary here tries to retrieve HTML page and then executes it with mshta.exe

2.18.2 Defensive options for "Mshta" technique

Detection

Data Source.	Data Component	Detects
Command	Command Execution	Look for mshta.exe executing raw or obfuscated script within the command-line. Compare recent invocations of mshta.exe with prior history of known good arguments and executed .hta files to determine anomalous and potentially adversarial activity. Command arguments used before and after the mshta.exe invocation may also be useful in determining the origin and purpose of the .hta file being executed.
File	File Creation	Monitor use of HTA files. If they are not typically used within an environment then execution of them may be suspicious.
Network Traffic	Network Connection Creation	Monitor for newly constructed network connections that are sent or received by untrusted hosts.
Process	Process Creation	Use process monitoring to monitor the execution and arguments of mshta.exe.

Mitigation The following table contains the possible mitigations against the attack technique.

Mitigation	Description
Disable or Remove Feature or Program	Mshta.exe may not be necessary within a given environment since its functionality is tied to older versions of Internet Explorer that have reached end of life.
Execution Prevention	Use application control configured to block execution of mshta.exe if it is not required for a given system or network to prevent potential misuse by adversaries. For example, in Windows 10 and Windows Server 2016 and above, Windows Defender Application Control (WDAC) policy rules may be applied to block the mshta.exe application and to prevent abuse [8].

2.18.3 Recommendations

Since the functionality of the mshta.exe is tied to older versions of Internet Explorer that have reached end of life, the user should remove or disable this feature.

2.19 Exfiltration Over C2 (`T1041`)

2.19.1 The process of the technique being used in the report

Sr. No.	Reference No.	Explanation
1	30	Here the adversary uses GitHub repo as command and control. After collecting data from the victim the adversary commits the data to the GitHub repo.

2.19.2 Defensive options for "Exfiltration Over C2" technique

Detection

Data Source.	Data Component	Detects
Command	Command Execution	Monitor executed commands and arguments that may steal data by exfiltrating it over an existing command and control channel.
File	File Access	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.
Network Traffic	Network Connection Creation	Monitor for newly constructed network connections that are sent or received by untrusted hosts.
	Network Connection Creation	Monitor and analyze traffic patterns and packet inspection associated with the 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 with traffic patterns (e.g. monitor anomalies in the use of files that do not normally initiate connections for the respective protocol(s)).
	Network Traffic Flow	Monitor network data for uncommon data flows. Processes utilizing the network that do not normally have network communication or have never been seen before are suspicious.

Mitigation The following table contains the possible mitigations against the attack technique.

Mitigation	Description
Data Loss Prevention	Data loss prevention can detect and block sensitive data being sent over unencrypted protocols.
Network Intrusion Prevention	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.[10]

2.19.3 Recommendations

The user should monitor network data for uncommon data flows. Processes utilizing the network that do not normally have network communication or have never been seen before are suspicious.

Since the adversary is using GitHub as C2 server, the user should look out for an unauthorized/ suspicious HTTP PUT request as this might be a possible exfiltration.

3 Assumption about the organization/user

In this campaign, the actor has targeted people that are looking for job opportunities at Lockheed Martin. Therefore the targets are assumed to be users with very limited resources. Here are the list of capabilities and constraints that we've assumed about the user.

Capabilities

- The user has anti-viruses installed and running.
- The user has the ability to change his/her own privilege.
- The user uses an e-mail gateway that can filter out spam e-mails in the mailbox.
- The user account can elevate its privilege to the administrator account.
- User can install new applications into their system.

Constraints

- It is highly likely that the user is searching job at a cybersecurity firm, so it can be assumed that the user has enough technical knowledge. But we will assume the worst.
- The user does not have enough resources to monitor each and every execution or function call at the process level.
- The user here can be a developer, so running arbitrary binaries might be a part of the work.

The recommendations made in the previous section are based on the assumptions made about the user here.

4 References

[1] Command and scripting interpreter: Visual basic.

[2] Process injection: Dynamic-link library injection, https://attack.mitre.org/techniques/T1055/001/.

[3] Application whitelisting: Panacea or propaganda, Sep 2022.

[4] Hossein Jazi Ankur Saini. North korea's lazarus apt leverages windows update client, github in latest campaign.

[5] Archiveddocs. Application lockdown with software restriction policies, Available here.

[6] Archiveddocs. Using software restriction policies and applocker policies, Available here.

[7] Red Canary. Spearphishing attachment. 2022.

[8] D. et al. Coulter. Microsoft recommended block rules. 2019.

[9] Denisebmsft. Application control for windows - windows security.

[10] Joseph Gardiner, Marco Cova, and Shishir Nagaraja. Command and control: Understanding, denying and detecting. 08 2014.

[11] A. Hosseini. Ten process injection techniques: A technical survey of common and trending process injection techniques. 2017, July 18.

[12] jweston 1. Use attack surface reduction rules to prevent malware infection, Available here.

[13] Microsoft. Attack surface reduction rules overview. 2022.

[14] Avira Security research. Anatomy of an exploit in windows win32k – cve-2022-21882. 06 2022.

[15] Mark Russinovich. Autoruns for windows v14.09, Available here.

[16] Microsoft Defender Security Research Team. Windows 10 to offer application developers new malware defenses. 2015.

[17] Shusei Tomonaga. Windows commands abused by attackers, Available here.