HOMEWORK 2

**Group No. 10**

**A Atulya Sundaram ( 210001 )**

**Dhiraj Pareek (231110012)**

**Akash Shivaji Varude (231110006)**

**1 Approach to Mapping Finished Reporting to ATT&CK**

**The snippet from the document is:**

| **Introduction**  Trend Micro’s Managed Extended Detection and Response (MxDR) team discovered that a file called x32dbg.exe  was used (via the DLL Search Order Hijacking or T1574.001 technique) to sideload a malicious DLL(1) we identified as  a variant of PlugX (Trojan.Win32.KORPLUG.AJ.enc). This file is a legitimate open-source debugger tool for Windows  that is generally used to examine kernel-mode and user-mode code, crash dumps, or CPU registers. Meanwhile,  PlugX is a well-known remote access trojan (RAT) that is used to gain remote access to and control over  compromised machines. It allows an attacker to obtain unauthorized access to a system, steal sensitive data, and  use the compromised machine for malicious purposes. The MxDR team employed a number of advanced security  technologies and solutions to gain a comprehensive understanding of the attack, which will be revealed in this report.  **Investigating and analyzing the threat with MxDR**  Being a legitimate application, x32dbg.exe’s valid digital signature can confuse some security tools(2), enabling threat  actors to fly under the radar, maintain persistence, escalate privileges, and bypass file execution restrictions.  The team's attention was first drawn to the command line execution of D:\RECYCLER.BIN\files\x32dbg.exe(3) which was flagged by a VisionOne Workbench alert. Further investigation revealed that this path led to a hidden folder(4) on the USB storage device, which was found to contain a number of threat components. |
| --- |

| **Reference**  **No.** | **ID** | **Tactic** | **Technique** | **Sub Technique** | **Explanation** |
| --- | --- | --- | --- | --- | --- |
| 1 | T1574.001 | Persistance | Hijack execution flow | DLL Search Order Hijacking | As attacker used x32dbg.exe by DLL search order hijacking to sideload a malicious DLL |
| 2 | T1218.011 | Defense Evasion | Subvert Trust Controls | Code Signing | Attacker used a legitimate application, x32dbg.exe’s valid digital signature to evade defenses |
| 3 | T1059.003 | Execution | Command and Scripting Interpreter | Windows command shell | Adversary used command line to execute x32dbg.exe file which was present in D:\RECYCLER.BIN\files\ |
| 4 | T1564.001 | Defense Evasion | Hide Artifacts | Hidden Files and Directories | The USB device had a hidden folder which contained threat components. So when attacker used executed x32dbg.exe file using command line, the path led to hidden folders in usb |

**The snippet from the document is:**

| We uncovered a clear sequence of events that began with a suspicious command line execution(5) launched via  cmd.exe. The command line executed the file  (ec5cf913773459da0fd30bb282fb0144b85717aa6ce660e81a0bad24a2f23e15 ) located at  D:\RECYCLER.BIN\files\x32dbg.exe. The file was signed by ”OpenSource Developer, Duncan Ogilvie” issued by  Certum Code Signing.(6) A visual representation of these events is displayed in Figure 3.  After executing D:\RECYCLER.BIN\files\x32dbg.exe, all of the threat components are copied to the directory(6.1)  C:\ProgramData\UsersDate\Windows\_NT\Windows\User\Desktop.  Subsequently, the file C:\ProgramData\UsersDate\Windows\_NT\Windows\User\Desktop\x32dbg.exe, a duplicate of the original file was invoked. The following command line was used to invoke the dropped file:  *Command Line: "C:\Windows\System32\cmd.exe" /q /c"*  *C:\ProgramData\UsersDate\Windows\_NT\Windows\User\Desktop//x32dbg.exe”*(7)  Figure 6. Vision Ones shows how x32dbg.exe copies itself to various directories and renames itself as  Mediae.exe(8) |
| --- |

| **Reference**  **No.** | **ID** | **Tactic** | **Technique** | **Sub Technique** | **Explanation** |
| --- | --- | --- | --- | --- | --- |
| 5,7 | T1059.003 | Execution | Command and Scripting Interpreter | Windows command shell | X32dbg file was executed using command line |
| 6 | T1588.003 | Resource Development | Obtain Capabilities | Code Signing Certificates | X32dbg.exe file was signed by an open source developer named Ducan Ogilvie. And the certificate was issued by Certum code signing |
| 6.1 | T1091 | Initial Access | Replication Through Removable Media | NA | When x32dbg.exe was executed then all the threat components which were present in USB’s hidden folder were copied to user’s desktop |
| 8 | T1036.005 | Defense Evasion | Masquerading | Match Legitimate Name or Location | When x32dbg.exe copies itself at different locations, it renames itself as Mediae.exe |

**The snippet from the document is:**

| We noticed the creation of a scheduled task via the schtasks command line utility to run a task at a specific time. In  this case, the scheduled task is set to execute the x32dbg.exe file(9), the open source debugger tool that side loads PlugX, every five minutes. The task is disguised under the name "LKUFORYOU\_1"(10) to make it more difficult to detect. |
| --- |

| **Reference**  **No.** | **ID** | **Tactic** | **Technique** | **Sub Technique** | **Explanation** |
| --- | --- | --- | --- | --- | --- |
| 9 | T1053.005 | Persistence | Scheduled Task | Scheduled Task/Job | To establish persistence, adversary sets a scheduled task which executes x32dbg.exe after every 5 minutes |
| 10 | T1036.004 | Defense Evasion | Masquerading | Masquerade Task or Service | A scheduled task which was created to run x32dbg.exe file after every 5 minutes, was renamed as LKUFORYOU\_1 |

**The snippet from the document is:**

| Figure 11 depicts where run registry keys were installed for persistence,(11) and the data associated with them. These  registry keys and values enable the threat to maintain persistence by automatically executing the x32dbg.exe file  every time the user logs in.  **Hiding in plain sight: DLL sideloading with x32dbg.exe**  We observed x32dbge.exe being used to sideload the PlugX file x32bridge.dll(12)  (0490ceace858ff7949b90ab4acf4867878815d2557089c179c9971b2dd0918b9, detected as  Trojan.Win32.KORPLUG.AJ). Sideloading can take advantage of the loader's DLL search order by placing the  malicious payload(s) and victim program side by side. This process is likely used by malicious actors as a cover for  operations carried out within a trusted, legitimate, and maybe elevated system or software process.  We observed that the file akm.dat (0e9071714a4af0be1f96cffc3b0e58520b827d9e58297cb0e02d97551eca3799,  detected as Trojan.Win32.KROPLUG.AJ) was also registered and executed via rundll32(13), a Windows component which attackers can abuse to facilitate the execution of malicious code. By using rundll32.exe to execute the file, the attackers can prevent security tools from monitoring this activity. |
| --- |

| **Reference**  **No.** | **ID** | **Tactic** | **Technique** | **Sub Technique** | **Explanation** |
| --- | --- | --- | --- | --- | --- |
| 11 | T1547.001 | Persistence | Boot or Logon Autostart Execution | Registry Run Keys / Startup Folder | registry keys created by adversary executes x32dbg.exe file every time user logins |
| 12 | T1574.002 | Persistence | Hijack Execution Flow | DLL Side-Loading | PlugX which is a trojan was sideloaded by x32dbg.exe |
| 13 | T1218.011 | Defense Evasion | System Binary Proxy Execution | Rundll32 | Another trojan ‘akm.dat’ was registered and executed by using rundll32 |

**The snippet from the document is:**

| The file x32dbg.exe is a legitimate executable of a debugging software which, when executed, imports x32bridge.dll  and calls on the functions *BridgeStart* and *BridgeInit*. The attackers took advantage of this and replaced the DLL with their own, containing the same export functions but executing entirely different codes(14):  BridgeStart – dummy code that does nothing  BridgeInit – Loads x32bridge.dat, decrypts its contents, then proceeds with the execution of the decrypted code.  The hardcoded key “HELLO\_USA\_PRISIDENT” is used to decode x32bridge.dat(15), after which execution will continue on the decrypted code. It will then check for an event named *LKU\_Test\_0.1* (or creates it if not found)(16). This is followed by the execution of akm.dat found in the same folder.  Next, it creates the scheduled task *LKUFORYOU\_1* to run x32dbg.exe persistently(17) like what was observed in our VisionOne investigation.  It then enumerates all drives and takes note of removable drives(18) for its propagation routine. |
| --- |

| **Reference**  **No.** | **ID** | **Tactic** | **Technique** | **Sub Technique** | **Explanation** |
| --- | --- | --- | --- | --- | --- |
| 14 | T1574.001 | Persistance | Hijack execution flow | DLL Search Order Hijacking | X32dbg.exe when executes it imports x32bridge.dll file. Attacker replaced this .dll file their own .dll file |
| 15 | T1140 | Defense Evasion | Deobfuscate/Decode Files or Information | NA | Adversary decoded x32bridge.dat file using hardcoded key |
| 16 | T1546 | Persistance | Event Triggered Execution | NA | When execution is done on decrypted code of x32bridge.dat file, it creates an event named LKU\_Test\_01 |
| 17 | T1053.005 | Persistance | Scheduled Task | Scheduled Task/Job | A scheduled task named LKUFORYOU\_01 was created by attacker to maintain persistence by executing x32dbg.exe |
| 18 | T1120 | Discovery | Peripheral Device Discovery | NA | After establishing persistence,(by executing x32dbg.exe persistently), it checks for all drives and takes notes of removable drives so that it can be propagated |

**The snippet from the document is:**

| As also seen in the VisionOne analysis, the malware checks if it already has an AutoStart registry key (x32dbg), and creates one if there isn’t.(19) Note that the execution path may vary depending on where x32dbg.exe / Mediae.exe was executed.  **Next stage loader: akm.dat**  The file akm.dat is a DLL with a straightforward function — to execute the next phase of the DLL sideloading routine.(20) Its export function *Start* will execute the file AUG.exe (also included in the previous installation from x32dbg.exe). |
| --- |

| **Reference**  **No.** | **ID** | **Tactic** | **Technique** | **Sub Technique** | **Explanation** |
| --- | --- | --- | --- | --- | --- |
| 19 | T1547.001 | Persistence | Boot or Logon Autostart Execution | Registry Run Keys / Startup Folder | Malware checks and creates Autostart registry key so that it can be persistently executed when user logins |
| 20 | T1574.002 | Defense Evasion | Hijack Execution Flow | DLL Side-Loading | There is an export function ‘start’ in akm.dat file. This function was used to perform next phase of DLL sideloading |

| **The backdoor UDP Shell: AUG.exe (with the components DismCore.dll**  **and Groza\_1.dat)**  AUG.exe is a copy of DISM.EXE, a legitimate Microsoft file which is also vulnerable to DLL sideloading. It imports the function DllGetClassObject from DismCore.dll, which will decrypt the contents of Groza\_1.dat using the hardcoded key *“Hapenexx is very bad”.(21)*  Figure 22. Decrypting Groza\_1.dat using the hardcoded key The execution will continue on the decrypted code, which is a UDP Shell client that does the following:  → Collects host information such as the hostname, IP Address and Mac address*(22)* and sends it to its command  and-control (C&C) server *160[.]20[.]147[.]254 (23)*  → Creates a thread to continuously wait for C&C commands  → Decrypts C&C communication using the hardcoded key*(24)* *“Happiness is a way station between too much and too little.”* |
| --- |

| **Reference**  **No.** | **ID** | **Tactic** | **Technique** | **Sub Technique** | **Explanation** |
| --- | --- | --- | --- | --- | --- |
| 21 | T1140 | Defense Evasion | Deobfuscate/Decode Files or Information | NA | The important function from DismCore decrypts Groza\_1.dat by using hardcoded key |
| 22 | T1082 | Discovery | System Information Discovery | NA | Adversaries discovers system information hostname, IP address and mac address |
| 23 | T1095 | Command and Control | Non-Application Layer Protocol | NA | Collected information is sent to C2C server. Adversary is using UDP shell (which is a non application layer protocol) as a |
| 24 | T1573.002 | Command and Control | Encrypted Channel | Asymmetric Cryptography | Attacker is using hardcoded key to decrypt to decrypt c2c communication. |

**DLL Search Order Hijacking**

| **Mitigation** | **Description** |
| --- | --- |
| Audit | 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.  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. |
| Execution Prevention | 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. |
| Restrict Library Loading | Disallow loading of remote DLLs. This is included by default in Windows Server 2012+ and is available by patch for XP+ and Server 2003+.  Enable Safe DLL Search Mode to force search for system DLLs in directories with greater restrictions (e.g. %SYSTEMROOT%)to be used before local directory DLLs (e.g. a user's home directory)  The Safe DLL Search Mode can be enabled via Group Policy at Computer Configuration > [Policies] > Administrative Templates > MSS (Legacy): MSS: (SafeDllSearchMode) Enable Safe DLL search mode. The associated Windows Registry key for this is located at HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\SafeDLLSearchMode |

**Defensive Recommandation:**

1. Safe DLL Search Mode should be enabled.
2. Auditing can also be done.
3. Monitor and control application execution: Monitor and control the execution of applications and their dependencies, including DLLs, to detect and prevent malicious activities.

**Rational:**

Safe DLL Search Modeprotects the system while not hampering the organization’s performance. This is also easily applied to a group of systems.

Auditing provides visibility into system activities, detecting unusual patterns and facilitating incident response.

# **Defense Evasion: Subvert Control trust (Code Signing)**

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

**Defensive Recommendations:**

1. Implement strong Certificate management practices.
2. Monitor and analyze certificate usage.
3. Enforce Application Whitelisting and Certificate Pinning.
4. Monitor and control application execution: Monitor and control the execution of applications and their dependencies, including DLLs, to detect and prevent malicious activities.

**Rational:**

By implementing robust certificate management practices, organizations can reduce the risk of adversaries acquiring or stealing code signing materials.

By closely monitoring certificate usage, organizations can promptly identify and respond to unauthorized or malicious activities involving code signing materials.

# **Command and Scripting Interpreter**

| **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 and JavaScript scripts from executing potentially malicious downloaded content . |
| Code Signing | Where possible, only permit execution of signed scripts. |
| Disable or Remove Feature or Program | Disable or remove any unnecessary or unused shells or interpreters. |
| Execution Prevention | Use application control where appropriate. For example, PowerShell Constrained Language mode can be used to restrict access to sensitive or otherwise dangerous language elements such as those used to execute arbitrary Windows APIs or files (e.g., Add-Type). |
| Privileged Account Management | When PowerShell is necessary, consider restricting PowerShell execution policy to administrators. Be aware that there are methods of bypassing the PowerShell execution policy, depending on environment configuration.  PowerShell JEA (Just Enough Administration) may also be used to sandbox administration and limit what commands admins/users can execute through remote PowerShell sessions. |
| 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. |

**Defensive Recommendation**

The company should install software that monitors what commands are being executed and control execution. These software are worth the investment seeing as multiple malware utilize the command line for execution. We are already assuming that this software is not in place as otherwise the malware given to us would have successfully attacked the system. This is considered as one of the cyber essentials for any organization.

**Rational:**

In our case malware was executed using the command line. So installing the software that monitors commands being executed and control execution provides essential defense against malware that exploits command line vulnerabilities

# **Hide Artifacts: Hidden Files and Directories**

| **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. |

**Defensive Recommendations:**

It should be made common knowledge in the organization that the user should monitor all newly created directories/files with the hidden attribute, for any malicious files. This can be easily done by selecting the “Show hidden folders” option in windows and set is as a default.

**Rational:**

Here, the company had installed a USB device with the given malware in a hidden folder, thus all the user would have to do was check that there are no hidden folder in the device. There are also many build artifacts that might be shown but with time the organization’s employee can easily get used to this are regain his efficiency.

# 

# 

# **Replication Through Removable Media**

| **Mitigation** | **Description** |
| --- | --- |
| Behavior Prevention on Endpoint | On Windows 10, enable Attack Surface Reduction (ASR) rules to block unsigned/untrusted executable files (such as .exe, .dll, or .scr) from running from USB removable drives. |
| Disable or Remove Feature or Program | Disable Autorun if it is unnecessary.Disallow or restrict removable media at an organizational policy level if it is not required for business operations. |
| Limit Hardware Installation | Limit the use of USB devices and removable media within a network. |

**Defensive Recommendation**

1. Assuming that the systems have Windows 10, the first mitigation can be easily implemented.
2. We cannot assume that Autorun is not necessary in the organization as sometimes removing it can hamper employee productivity.

**Rational:**

Malware was searching for drives and removable media for its propagation. So implementing the Windows 10 feature to disable AutoRun can effectively mitigate the malware's propagation through drives, enhancing the organization's security without compromising necessary functionalities.

# **Obtain Capabilities: Code Signing Certificates**

| **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. |

**Defensive Recommendation and Rational:**

The organization should check for codes that are signed by known sources. But in our case not much can be done as there was masquerading involved as well, which can fool the code checker, as the certificate could have been that of a known organization.

# **Masquerading: Match Legitimate name or location**

| **Mitigation** | **Description** |
| --- | --- |
| Code Signing | Require signed binaries and images. |
| 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. |

**Defensive Recommendations:**

The easiest way is Execution Prevention. Here if there was such prevention already enabled, we would not have reached this stage. Thus, breaking off the chain at the root would be the easiest solution and most recommended.

**Rational:** Attacker has changed malware name to mediae.exe to make it look legitimate. So enforcing execution prevention measures to disrupt the attacker's ability to change the malware name to match legitimate ones can mitigate the risk at its source

# **Scheduled Task/Job: Schedule Task**

| **Mitigation** | **Description** |
| --- | --- |
| Audit | Toolkits like the PowerSploit framework contain PowerUp modules that can be used to explore systems for permission weaknesses in scheduled tasks that could be used to escalate privileges. |
| Operating System Configuration | Configure settings for scheduled tasks to force tasks to run under the context of the authenticated account instead of allowing them to run as SYSTEM. The associated Registry key is located at HKLM\SYSTEM\CurrentControlSet\Control\Lsa\SubmitControl. The setting can be configured through GPO: Computer Configuration > [Policies] > Windows Settings > Security Settings > Local Policies > Security Options: Domain Controller: Allow server operators to schedule tasks, set to disabled. |
| Privileged Account Management | Configure the Increase Scheduling Priority option to only allow the Administrators group the rights to schedule a priority process. This can be configured through GPO: Computer Configuration > [Policies] > Windows Settings > Security Settings > Local Policies > User Rights Assignment: Increase scheduling priority. |
| User Account Management | Limit privileges of user accounts and remediate Privilege Escalation vectors so only authorized administrators can create scheduled tasks on remote systems. |

**Defensive Recommendations:**

1. The second option of OS configuration is the option that is easily recommended as it can be easily incorporated within the organization by changing the system settings. This change will ensure that only the employees or administrator can run the scheduled tasks. In general, there won’t be any necessity for employees to do so unless they are installing software, and this too should be done with the organization's knowledge.

**Rational:**

Attacker has scheduled task multiple times so implementing OS configuration changes to restrict scheduled task execution to employees or administrators could enhance security by preventing unauthorized access. And also this is easily adoptable measure aligns with best practices, limiting potential misuse and ensuring that task scheduling aligns with organizational policies.

# **Boot or Logon Autostart Execution**

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

**Defensive Recommendation**

The user can use tools such as Sysinternals Autorun to detect system changes that could be attempted at persistence, including the startup folders. This utility has the most comprehensive knowledge of auto starting locations of any startup monitor. It will enable you to detect which Run keys have been modified. Installing this software will be worth the extra effort for the organization as many malware enable persistence through Registry Keys and detection will enable containing the malware. We need to contain the malware as clearly this organization has data that may be distributed, seen as how Replication through Removable media is one of the techniques, and thus indicates that the adversary may want to access many systems and that compromising one system is probably not sufficient to handicap the organization.

**Rational:** Adversary has gained persistence by executing malware whenever user logins. So using Sysinternals Autorun or any other tool that is used to monitor changes that attempts persistence could detect autostart execution

# **System Binary Proxy Execution: Rundll32**

| **Mitigation** | **Description** |
| --- | --- |
| Exploit Protection | Microsoft's Enhanced Mitigation Experience Toolkit (EMET) Attack Surface Reduction (ASR) feature can be used to block methods of using rundll32.exe to bypass application control. |

**Defensive Recommendation:** Implement behavioral monitoring on rundll32.exe to detect anomalous activities, employ application whitelisting to restrict its usage, and ensure regular patching for addressing vulnerabilities exploited in system binary proxy execution

**Rational:** In this case adversary used rundll32.exe to continue next phase of sideloading. So doing this can help in the detection of DLL sideloading and attack could have been detected

# 

# **Masquerading: Masquerade Task or Service**

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

**Defensive Recommendation and Rational:**

Again, Execution Control would have prevented this attack. Also we should thoroughly check which tasks are running in order to prevent unknown tasks from continuing to execute. This can be checked by a thorough manual inspection.

**Persistence: DLL Side-Loading**

| **Mitigation** | **Description** |
| --- | --- |
| Application Developer Guidance | When possible, include hash values in manifest files to help prevent side-loading of malicious libraries. |
| Update Software | Update software regularly to include patches that fix DLL side-loading vulnerabilities. |

**Defensive Recommendation**

1) The organization should check if there are updates available for their software that could fix the side-loading vulnerabilities and subsequently update them if it is. Software used by corporate entities generally allow the organization to update their software system-wide through the administrator, thus this task should not be very cost-intensive nor should it require a lot of man-power. However, even if it does require some man-power, it is well worth the effort, as updates could also fix major bugs in the software that might hamper the organization’s functioning in the future

2) Endpoint protection: Use endpoint protection solutions that offer behavioral analysis and predictive machine learning for better security capabilities

3) Implement whitelisting: Allow only known and trusted applications to run on the system while blocking any suspicious or unknown ones.

**Rational:**

Here adversary heavily depended on DLL side loading so, regularly updating software mitigates DLL side-loading vulnerabilities, enhancing security, and endpoint protection with advanced features adds proactive defense, collectively safeguarding the organization against persistent threats.

**Defense Evasion : Deobfuscate/Decode Files or Information**

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

**Persistence : Event Triggered Execution**

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

**Discovery**

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

**Command and Control : Encrypted Channel**

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

**Defensive Recommendation and rational:**

1. Deploy DPI technology to inspect network traffic for signs of encrypted communication patterns associated with C2 channels. DPI can help identify and block suspicious encrypted traffic.
2. Implement NADS to monitor network traffic behavior and detect anomalies indicative of C2 communication. These systems can identify unusual patterns.

**Command and Control : Non-Application Layer Protocol**

| **Mitigation** | **Description** |
| --- | --- |
| Filter Network Traffic | Filter network traffic to prevent use of protocols across the network boundary that are unnecessary. |
| 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. |
| Network Segmentation | Properly configure firewalls and proxies to limit outgoing traffic to only necessary ports and through proper network gateway systems. Also ensure hosts are only provisioned to communicate over authorized interfaces. |

**Defensive Recommendation:**

1. Divide the network into separate segments based on the sensitivity of data and functional requirements. Restrict communication between segments to limit the spread of C2 traffic and unauthorized access to critical assets.
2. Use network monitoring and analysis tools to inspect network traffic for anomalies and suspicious patterns indicative of C2 communication using non-application layer protocols. Look for unusual network connections or traffic flows that deviate from normal behavior.

**Assumptions About the Organization:**

PlugX trojan refers to a type of malware that has been known to target Windows systems. PlugX is a remote access trojan (RAT) that allows attackers to gain unauthorized access to compromised systems.

The targeted users are assumed to be individuals or employees within organizations that deal with sensitive or valuable information, such as intellectual property, financial data, or government secrets.

**Capabilities:**

1. The users may have basic security measures in place, such as antivirus software and firewalls, but their effectiveness may vary.
2. Users have some level of technical knowledge and may be aware of common cybersecurity threats, including malware and phishing attacks.
3. The organization may have implemented email security solutions that can filter out some malicious emails, but they may not catch all sophisticated phishing attempts.

**Constraints:**

1. Users may not have dedicated cybersecurity teams or resources to monitor and respond to security incidents in real-time.
2. Limited awareness of advanced threats like PlugX trojan and their potential impact on organizational security.
3. Users may have limited control over system configurations and security policies, depending on the organization's IT infrastructure and policies.