Downloaded:

nov 6 2016 22:12:12 Slovak time

1a. What observations and conclusions can you draw about this incident?

Short version: The Windows 7 system is exploited using Internet Explorer Flash plugin, malicious payload is downloaded and executed.

- the first visited web-page "goodwinstep.tk" probably has iframe\js that collect basic intelligence about the visiting user like IP, URL, User-Agent, and installed plugins
- the user is then immediately redirected to the exploit based on the previous intel, which, in this case, exploited Flash Player vulnerability in the served SWF file (via redirection or iframe)
- exploit use flash Vector object vulnerability to achieve arbitrary read\write memory access, then it takes over program – Flash plugin – execution
- exploit code loads winhttp.dll and advapi32.dll libraries
- exploit code calls VirtualProtect with the parameter PAGE_EXECUTE_READWRITE, then change the memory region to PAGE_EXECUTE_READ for the execution
- memory is allocated using VirtualAlloc with the flag PAGE_EXECUTE_READWRITE, multiple calls are made
- malicious payload is stored to the Temp directory as "80D4.tmp" file, the Temp path string is found by calling GetTempPathA, the file is stored by calling CreateFileA function
- I believe "80D4.tmp" is an DLL library, which is then injected into Internet Explorer process using CreateProcessInternalW API call, but due to really small differences between DLL and PE EXE file structure, both of them can be loaded into an existing process
- The payload then contact control server – 10.0.0.2:8080 - IP belongs to the local network, I believe that some Metasploit module, or other exploitation framework is used for penetration testing in this case

1b. What automated detection and analysis could have been used to detect this malware? Provide as much detail as you can about possible detection techniques, including those that weren't used to produce this alert.

- Check for the suspicious redirections, in the case that “suspicious” URL redirecting to the Flash component, Flash component should not be served to the endpoint.

- Obfuscated JavaScript can be executed and analyzed by independent JavaScript engine (ie. JSDetox, Chrome v8, SpiderMonkey) using honeypot / virtual machine. If the known URL is present, we should break the execution. This can be implemented like recommendation engine within the internal network, but the latency will be high, if we want to check every link.
- URL patterns, file names, and file hashes should be investigated so an appropriate rules can be created for the monitoring
- the first website serving the redirection is probably hacked, but due to expensive nature of the exploits, we can predict that the exploit is hosted on the machine operated directly by the attackers as they do not want to have exploits stolen, analyzed and IPs blocked. Directly operated and owned \ rented machines are not cheaply changed, we should use this in our blacklist

1c. What recommendations would you make to prevent future compromise from similar threats?

- in this case update Flash, because vulnerability which exploits Vector structure is not the newest one
- I would strongly recommend to uninstall \ forbid Flash, ActiveX and Java plugin as well
- Better practice is to DISABLE all 3rd party plugins, and restrict JavaScript for the same URL origin policy,
- Block all JavaScript beacons in general using browser extensions (AdBlock, Ghostery - probably not helpful in this case)
- Use browser with sandboxing implemented - ie. Chromium, or use browser in an sandbox or Virtual machine
- Implement policy for all browsers on the internal network to have recommended settings and regular updates
- IPS system should blacklist all URL addresses that are known to serve malicious code, we should add this URL to the blacklist as well
- If the setting does not brick the environment, in some corner cases we can set variables like DebuggerPresent, SystemKernelDebuggerInformation or use known names of the VM processes in order to stop execution of the payload. This trick can be used again obfuscated, honeypot aware samples, but this is really not the good choice
- setup firewall with the manual rules creation, in this case, the malware will be installed anyway, but you can mitigate his ability to communicate after infection

2a. What observations and conclusions can you draw about this incident?

- The infected php Wordpress webpage is used as an C&C server. Unknown data blob is sent there using HTTP POST request.
- 4 different addresses are contacted, blob is sent to 3 of them, the first domain accepted it with the HTTP 200 OK
- According to the “/bstr.php” URLs used for POST requests, TeslaCrypt:) ransomware is used. We do not have an executable, so we must find out how the request data are structured.
- TeslaCrypt in the earlier versions used symmetric key for key delivery mechanism, we should try to exploit this vulnerability.

2b. What actions and recommendations would you make to recover from this ransomware infection?

- I can write decryption routine based on the CheckPoint TeslaCrypt analysis: https://blog.checkpoint.com/wp-content/uploads/2016/05/Tesla-crypt-whitepaper_V3.pdf
- from the captured traffic, we can decrypt captured HTTP POST data using static IV, key:
  IV = '\xFF\xFF\xAA\xAA\x00\x00\xBE\xEF\xDE\xAD\x00\x00\xBE\xFF\xFF\xFF'
  key = sha256("0324532423723948572379453249857")
- We have obtained all the necessary information for the decryption:
  Sub=Ping&dh=04CD4AD422DD2AC3E2CF8FC37C6D223649E0FFA4372B92FA680649B0ACE2D73D776B11767711CFB1BDB2CC40EB3FA5F043B6FE9182D0514C772FE1F99C5A2E1B5815617B0B397A628FB47200B2EF744B0766A7A58213EA886636A92796751E6C2E&addr=17idD8nU8YaYxcscRSo7Do1JdZtSHMY1tx&size=0&version=3.0.1&OS=2600&ID=85&inst_id=A0E07DC459DED0DA

- Where DH parameter is global recovery ECDH key:
  04CD4AD422DD2AC3E2CF8FC37C6D223649E0FFA4372B92FA680649B0ACE2D73D776B11767711CFB1BDB2CC40EB3FA5F043B6FE9182D0514C772FE1F99C5A2E1B5815617B0B397A628FB47200B2EF744B0766A7A58213EA886636A92796751E6C2E
- we are now able to decrypt infected file

3. You have received a disk image suspected of containing malware. Please examine the disk and documentation all relevant observations.

- I have reconstructed the partition but I was not able to automatically infect the computer. Even when the Windows autorun flags were set. I believe that autorun was not able to execute file from the partition.
- I have used testdisk - linux disk recovery software as well as hex editor in order to obtain hidden VBscript code. Then I have executed modified version in the VirtualBox to deobfuscate itself.
- For more detailed image reconstruction, please check task_3_notes.txt file.

Malware functionality:
Function names are in Chinese and this malware family was spreading back in 2007, so it is possible that the author really is from that part of the world. But maybe it's just camouflage.

After execution, VBScript copies itself into %System Root% and %System% directories.
In my case C:\Windows\System32 and C:\ on Windows 7 was used. I had to execute it as an Admin.
Files copied are:
main.txt
main.vbe
main.vbs

Each of them contains lightly obfuscated version of the initial script that is executed from the USB image. All copied files are hidden.

Script creates file with the date of the infection, in my case VirtualBox time:
C:\date.bin - content: "11/8/2016 5:18:15 AM"

Script sets Windows setting "show hidden files" to False, so the files are not visible in the Explorer.
"HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced\"
Type_Name="REG_DWORD"
Key_Name="ShowSuperHidden"
Key_Data="00000000"

Script creates registry entries (if possible) for reboot survival:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\
policies\Explorer\run
explorer = "main.vbe"
```

VB script writes AUTORUN.INF file to the various folders (C:\, system32, every drive...) in order to get executed. It also copies itself to every attached disk, removable media, network drives.

Content of the Autorun.inf:

```
daxian3.1
[AutoRun]
daxianbiyeliunian 2007.7.15
open=WScript.exe .\main.vbs
3.1
shell\open=����(&O)
http://hgz.dinghui123.cn/wan.asp
shell\open\Command=WScript.exe .\main.vbs
shell\open\Default=1
1/6/2009 10:29:12 PM
```

Finally, this script is checking url address in order to download additional malicious executable from the Internet. The executable is then executed. Requested URL is not working anymore.

URL: hXXp://hgz.dinghui123.cn/wan.asp   (http string changed)