Can You Trust a File's Digital Signature? New Zloader Campaign exploits Microsoft's Signature Verification putting users at risk

By etal

Published: 2022-01-05 · Archived: 2026-04-23 02:17:28 UTC

Research by: Golan Cohen

Introduction

Last seen in August 2021, Zloader, a banking malware designed to steal user credentials and private information, is back with a simple yet sophisticated infection chain. Previous Zloader campaigns, which were seen in 2020, used malicious documents, adult sites and Google ads to infect systems.
Evidence of the new campaign was first seen around early November 2021. The techniques incorporated in the infection chain include the use of legitimate remote management software (RMM) to gain initial access to the target machine.

FIGURE 1 – SIMPLIFIED INFECTION CHAIN

The malware then exploits Microsoft's digital signature verification method to inject its payload into a signed system DLL to further evade the system's defenses. This evidence shows that the Zloader campaign authors put great effort into defense evasion and are still updating their methods on a weekly basis.

Infection Chain

The infection starts with the installation of Atera software on the victim's machine. Atera is a legitimate, enterprise remote monitoring and management software, designed for IT use. Atera can install an agent and assign the endpoint to a specific account using a unique .msi file that includes the owner's email address. The campaign authors created this installer (b9d403d17c1919ee5ac6f1475b645677a4c03fe9) with a temporary email address: 'Antik.Corp@mailto.plus'. The file imitates a Java installation, just like in previous Zloader campaigns. As of this moment, the exact distribution method for this file is not fully understood.

Figure 2 – The malicious installer

Once the agent is installed on the machine, the attacker has full access to the system and is able to upload/download files, run scripts, etc. Atera offers a free 30-day trial for new users, which is enough time for the attacker to stealthily gain initial access. Previously, Atera was used by the Conti ransomware group to gain persistence and remote access.

Figure 3 – Create custom Atera installer

Figure 4 – The email used in the malicious installer

Figure 5 – Atera Functions

Following the agent installation, the attacker then uploads and runs two .bat files onto the device using the “Run Script” function:

- defenderr.bat is used to modify Windows Defender preferences.

- load.bat is used to load the rest of the malware.

Figure 6 – defenderr.bat

The rest of the files are hosted on the domain teamworks455[.]com and are downloaded from there.

Figure 7 – load.bat

The load.bat script downloads and runs new.bat, which checks for admin privileges and requests them using the BatchGotAdmin script. It then continues to download another bat file (new1.bat). This new script adds more exclusions to Windows Defender for different folders, disables different tools on the machine that could be used for detection and investigation such as cmd.exe and the task manager. It also downloads other files into the %appdata% folder:

- 9092.dll – The main payload, Zloader.

- adminpriv.exe – Nsudo.exe. Enables running programs with elevated privileges.
- appContast.dll – Used to run 9092.dll and new2.bat.
- reboot.dll – Also used to run 9092.dll.
- new2.bat – Disables “Admin Approval Mode” and shuts down the computer.
- auto.bat – Placed in the Startup folder for boot persistence.

Figure 8 – New.BAT

Figure 9 – New1.BAT

Next, the script runs mshta.exe with file appContast.dll as the parameter. When we took a closer look at the DLL, we noticed that the file is signed by Microsoft with a valid signature (see below for further explanation) and its

original filename is AppResolver.dll. Comparing the two files, we see that in the malicious DLL, the author appended a script to the file.

Figure 10 – Valid Signature

Figure 11 – Original Filename

Figure 12 – Comparison of appResolver.dll and appContast.dll

This script then enters a sleeping phase using the file WScriptSleeper.vbs which is written to the %temp% directory. Next, it runs 9092.dll (the main Zloader payload) using regsvr32.exe.

A full technical analysis of Zloader was published by Malwarebytes in May 2020. Ultimately, the malware calls msiexec.exe and injects its payload into the running process. Msiexec then communicates with the C2 server at the

domain lkjhgfgsdshja[.]com.

Figure 13 – Communication to the C2 server

Figure 14 – Strings extracted from msiexec memory

Finally, the new2.bat script edits the registry SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System to disable the “administrator in Admin Approval Mode” user type, which runs all applications by default with full administrator privileges, and then shuts down the computer for the changes to take effect.

Persistence

When the malware initially runs, it places an auto.bat script under the Startup folder which runs mshta.exe with reboot.dll as a parameter. Similar to appContast.dll, and then the script deletes itself. In the figure below, we see that regsvr32.exe is called with zoom.dll and 9092.dll. The file zoom.dll is missing, which indicates that this campaign might still be under development and we will see it in the future.
After injecting msiexec.exe with the malicious code, a random registry key value is created under HKCU\Software\Microsoft\Windows\CurrentVersion\Run. This runs regsvr32.exe with a copy of 9092.dll, which is placed in a newly created folder in %appdata%. This is how the malware persists the next time the system reboots.

Figure 15 – reboot.dll

File Signature

As mentioned above, the file appContast.dll has a valid signature by Microsoft but the file has been modified and injected with a malicious script. This begs the question – how was it done?
If we compare the malicious DLL with the original one on a byte level, we can see the file was modified in a few places: File checksum and two places that match the signature size.

Figure 16 – A (benign) is appResolver.dll, B (malicious) is appContast.dll

Figure 17 – appResolver.dll CheckSum

Figure 18 – appResolver.dll signature size

figure 19 – appResolver.dll signature size (2)

These simple modifications to a signed file maintain the signature's validity, yet enables us to append data to the signature section of a file. As we can't run compiled code from the signature section of a file, placing a script written in VBscript or JavaScript and running the file using mshta.exe is an easy solution that could evade some EDRs.
As a sanity check, we created our own signed PE file with an appended script (A6ED1667BB4BB9BAC35CE937FF08C7216D63EBB4) that opens the calculator app when run as a parameter to mshta.exe.
This gap is apparently a known issue mentioned in the following CVEs: CVE-2020-1599, CVE-2013-3900, and CVE-2012-0151. Microsoft addressed the issue in 2013 with a Security Bulletin and pushed a fix. However, they stated after implementing it that they "determined that impact to existing software could be high." Therefore, in July 2014, they pulled the stricter file verification and changed it to an opt-in update.

In other words, this fix is disabled by default, which is what enables the malware author to modify the signed file.

Further explanation about how to enable the strict file verification is available here, which includes modifying the registry keys:

Figure 20 – Keys needed to change to mitigate the issue

We note that reboot.dll is also signed in the same way. After applying the fix, both DLLs have an invalid signature.

Campaign Victims

During our analysis, we found an open directory, hosted at teamworks455[.]com, that holds some of the files that are downloaded and used. Every few days, the author makes changes to the files and the check.php script returns a different DLL file with the same behavior, but a different hash. In the file `entries', we can see a list of victims that are infected with Zloader and their country of origin.

Figure 21 – teamworks455[.]com/_country

As of January 2, 2022, there are **2170 unique victim IPs** that downloaded the malicious DLL file. This graph shows the number of victims from each country ("Other" category includes countries with less than 15 victims). As you can see, most of the victims reside in the United States and Canada

Figure 22 – Downloads per country

Campaign Authors

Due to a few similarities with previous campaigns by MalSmoke, we believe that they are the cybercriminals behind this campaign:

- Malware in previous campaigns by MalSmoke are known to masquerade as Java plugins, which is occurring in this case.
- There is a connection between the registrar information of the domain teamworks455[.]com, where the current campaign files are hosted, and the domain pornislife[.]online which was linked to a MalSmoke campaign in 2020

Figure 24 – teamworks455[.]com/racoon

Finally, when looking through the ‘entries’ file, we found two IP addresses that might be related to the attackers.

Figure 25 – Possible addresses related to the campaign

The first address, 185[.]191[.]34[.]223, was spotted in an IP blacklist that is categorized as “cybercrime.” The second address, 185[.]191[.]34[.]209, can be seen attempting to download the payload multiple times, using different user-agents. This could indicate that the authors were testing their payload. Both addresses are found in AbuseIPDB:

Figure 26 – Abuseipdb 185[.]191[.]34[.]223

Figure 27 – Abuseipdb 185[.]191[.]34[.]209

Conclusion

Zloader campaigns have been previously spotted in the wild in multiple forms. In this particular case, we see that the authors put a lot of effort into the evasion methods. Two noteworthy ways seen here are using legitimate RMM software as an initial access to a target machine, and appending code to a file's signature while still maintaining the signature's validity and running it using mshta.exe.
The ability to append code to a file's signature has been known for many years and multiple CVEs were assigned as mentioned above. To mitigate the issue, all vendors should conform to the new Authenticode specifications to have these settings as default, instead of an opt-in update. Until that happens, we can never be sure if we can truly trust a file's signature.

Safety Tips

We recommend that users apply Microsoft's update for strict Authenticode verification. To do so, paste these lines into Notepad and save the file with .reg extension before running it.

Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\Software\Microsoft\Cryptography\Wintrust\Config]

"EnableCertPaddingCheck"="1"

[HKEY_LOCAL_MACHINE\Software\Wow6432Node\Microsoft\Cryptography\Wintrust\Config]

“EnableCertPaddingCheck”=”1”

We should also note that after applying the fix, some signatures of legitimate benign installers will show up with an invalid signature. In addition, if mshta.exe is not relevant in your environment, you may disable it and mitigate the execution of scripts that are inserted into such files.

Check Point Threat Emulation and Harmony endpoint provides protection against this threat:

- Exploit.Wins.CVE-2013-3900.A
- Trojan-Downloader.Win.Zloader.E
  Trojan-Downloader.Win.Zloader.F

MITRE ATT&CK

IOCs

AteraAgent Scripts:

Defenderr.bat – 1CA89010E866FB97047383A7F6C83C00C3F31961

Load.bat – F3D73BE3F4F5393BE1BC1CF81F3041AAD8BE4F8D

www.teamworks455[.]com

C2 Servers:

https://asdfghdsajkl[.]com/gate.php

https://iasudjghnasd[.]com/gate.php

https://kdjwhqejqwij[.]com/gate.php

https://kjdhsasghjds[.]com/gate.php

https://dkisuaggdjhna[.]com/gate.php

https://dquggwjhdmq[.]com/gate.php

https://lkjhgfgsdshja[.]com/gate.php

https://daksjuggdhwa[.]com/gate.php

https://eiqwuggejqw[.]com/gate.php

https://djshggadasj[.]com/gate.php

Files:

Java.msi – B9D403D17C1919EE5AC6F1475B645677A4C03FE9

new.bat – 0926F8DF5A40B58C6574189FFB5C170528A6A34D

new1.bat – 9F1C72D2617B13E591A866196A662FEA590D5677

new2.bat – DE0FA1529BC652FF3C10FF16871D88F2D39901A0

9092.dll – A25D33F3F8C2DA6DC35A64B16229D5F0692FB5C5, 7A57118EE3122C9BDB45CF7A9B2EFD72FE258771, 2C0BC274BC2FD9DAB82330B837711355170FC606

Adminpriv.exe – 3A80A49EFAAC5D839400E4FB8F803243FB39A513

appContast.dll – 117318262E521A66ABA4605262FA2F8552903217

reboot.dll – F3B3CF03801527C24F9059F475A9D87E5392DAE9

auto.bat – 3EA3B79834C2C2DBCE0D24C73B022A2FF706B4C6
