Sep 02 2023

NETMANAGEIT

Intelligence Report

Fake Update Utilizes New IDAT Loader To Execute StealC and Lumma Infostealers

Table of contents

Overview

Entities

Observables

External References

Overview

Description

Fake browser updates lure users into executing malicious binaries which include a new IDAT loader which is utilized in order to execute infostealers on compromised systems including StealC and Lumma.

Confidence

This value represents the confidence in the correctness of the data contained within this report.

15 / 100

Indicator

Name
be8eb5359185baa8e456a554a091ec54c8828bb2499fe332e9ecd65639c9a75b
Pattern Type
stix
Pattern
[file:hashes.'SHA-256' = 'be8eb5359185baa8e456a554a091ec54c8828bb2499fe332e9ecd65639c9a75b']
Name
3bf4b365d61c1e9807d20e71375627450b8fea1635cb6ddb85f2956e8f6b3ec3
Description
Zeppelin_10
Pattern Type
stix
Pattern

[file:hashes.'SHA-256' = '3bf4b365d61c1e9807d20e71375627450b8fea1635cb6ddb85f2956e8f6b3ec3']

Name

b3d8bc93a96c992099d768beb42202b48a7fe4c9a1e3b391efbeeb1549ef5039

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = 'b3d8bc93a96c992099d768beb42202b48a7fe4c9a1e3b391efbeeb1549ef5039']

Name

buyerbrand.xyz

Pattern Type

stix

Pattern

[domain-name:value = 'buyerbrand.xyz']

Name

94.228.169.55

Description

ISP: AEZA GROUP Ltd **OS:** Ubuntu ---------- Hostnames: - prickly-cats.aeza.network ---------- Domains: - aeza.network ----------

Services: **22:** ``` SSH-2.0-OpenSSH_8.9p1 Ubuntu-3ubuntu0.1 Key type: ecdsa-sha2-nistp256 Key: AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBAs7DXIYf7Tyn/E+gq/l4nYj YVDYK7glNO749/V21he0q50gfXwnN5+NTteezCvfypsb3NdzmGB1EvDlRAsaQWo= Fingerprint: cc:71:77:37:b3:df:a6:00:c7:ba:4c:9b:31:84:ec:19 Kex Algorithms: curve25519-sha256 curve25519-sha256@libssh.org ecdh-sha2-nistp256 ecdh-sha2-nistp384 ecdh-sha2-nistp521 sntrup761x25519-sha512@openssh.com diffie-hellman-group-exchange-sha256 diffie-hellman-group16-sha512 diffie-hellman-group18-sha512 diffie-hellman-group14-sha256 Server Host Key Algorithms: rsa-sha2-512 rsa-sha2-256 ecdsa-sha2-nistp256 ssh-ed25519 Encryption Algorithms: chacha20-poly1305@openssh.com aes128-ctr aes192-ctr aes256-ctr aes128-gcm@openssh.com aes256-gcm@openssh.com MAC Algorithms: umac-64-etm@openssh.com umac-128-etm@openssh.com hmac-sha2-256-etm@openssh.com hmac-sha2-512-etm@openssh.com hmac-sha1-etm@openssh.com umac-64@openssh.com umac-128@openssh.com hmac-sha2-256 hmac-sha2-512 hmac-sha1 Compression Algorithms: none zlib@openssh.com ``` ---------- **80:** ``` HTTP/1.1 200 OK Server: nginx/1.18.0 (Ubuntu) Date: Fri, 25 Aug 2023 03:58:37 GMT Content-Type: text/html; charset=UTF-8 Content-Length: 265 Connection: keep-alive Vary: Accept-Encoding ``` ----------

Pattern Type

stix

Pattern

[ipv4-addr:value = '94.228.169.55']

Name

gapi-node.io

Pattern Type

stix

Pattern

[domain-name:value = 'gapi-node.io']

Name

winextrabonus.life

Pattern Type

stix

Pattern

[domain-name:value = 'winextrabonus.life']

Name

a0319e612de3b7e6fbb4b71aa7398266791e50da0ae373c5870c3dcaa51abccf

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = 'a0319e612de3b7e6fbb4b71aa7398266791e50da0ae373c5870c3dcaa51abccf']

Name

931d78c733c6287cec991659ed16513862bfc6f5e42b74a8a82e4fa6c8a3fe06

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = '931d78c733c6287cec991659ed16513862bfc6f5e42b74a8a82e4fa6c8a3fe06']

Name

d19c166d0846ddaf1a6d5dbd62c93acb91956627e47e4e3cbd79f3dfb3e0f002

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = 'd19c166d0846ddaf1a6d5dbd62c93acb91956627e47e4e3cbd79f3dfb3e0f002']

Name

51cee2de0ebe01e75afdeffe29d48cb4d413d471766420c8b8f9ab08c59977d7

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = '51cee2de0ebe01e75afdeffe29d48cb4d413d471766420c8b8f9ab08c59977d7']

Name

53c3982f452e570db6599e004d196a8a3b8399c9d484f78cdb481c2703138d47

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = '53c3982f452e570db6599e004d196a8a3b8399c9d484f78cdb481c2703138d47']

Name

omdowqind.site

Pattern Type

stix

Pattern

[domain-name:value = 'omdowqind.site']

Name

bgobgogimrihehmxerreg.site

Pattern Type

stix

Pattern

[domain-name:value = 'bgobgogimrihehmxerreg.site']

Name

weomfewnfnu.site

Pattern Type
stix
Pattern
[domain-name:value = 'weomfewnfnu.site']
Name
ocmtancmi2c5t.xyz
Pattern Type
stix
Pattern
[domain-name:value = 'ocmtancmi2c5t.xyz']
Name
c9094685ae4851fd5a5b886b73c7b07efd9b47ea0bdae3f823d035cf1b3b9e48
Description
Zeppelin_10
Pattern Type
stix
Pattern

[file:hashes.'SHA-256' = 'c9094685ae4851fd5a5b886b73c7b07efd9b47ea0bdae3f823d035cf1b3b9e48']

Name

ooinonqnbdqnjdnqwqkdn.space

Pattern Type

stix

Pattern

[domain-name:value = 'ooinonqnbdqnjdnqwqkdn.space']

Name

doorblu.xyz

Pattern Type

stix

Pattern

[domain-name:value = 'doorblu.xyz']

Name

gstatic-node.io

Description

Win32/Lumma

Pattern Type

stix

Pattern

[domain-name:value = 'gstatic-node.io']

Name

5f57537d18adcc1142294d7c469f565f359d5ff148e93a15ccbceb5ca3390dbd

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = '5f57537d18adcc1142294d7c469f565f359d5ff148e93a15ccbceb5ca3390dbd']

Name

8ce0901a5cf2d3014aaa89d5b5b68666da0d42d2294a2f2b7e3a275025b35b79

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = '8ce0901a5cf2d3014aaa89d5b5b68666da0d42d2294a2f2b7e3a275025b35b79']

Name

https://ocmtancmi2c5t.xyz/82z2fn2afo/b3/update.msi

Description

Composite Document File V2 Document, Little Endian, Os: Windows, Version 10.0, MSI Installer, Last Printed: Fri Dec 11 11:47:44 2009, Last Saved Time/Date: Fri Sep 18 14:06:51 2020, Security: 0, Code page: 1252, Revision Number: {94224AA3-26E6-468B-88D7-094689CA5B5B}, Number of Words: 10, Subject: Installation Assistant S54FCF1E7-E6A4-478B-u7tmn7rcpfvzbyy, Author: p9mc6m, Last Saved By: p9mc6m, Name of Creating Application: Installation Assistant S54FCF1E7-E6A4-478B-u7tmn7rcpfvzbyy, Template: ;1 4fc609aab3c404ae776ebdd60f1dbf1d0f0b3aa7aeace20b61b8c64335fd06c9

Pattern Type

stix

Pattern

[url:value = 'https://ocmtancmi2c5t.xyz/82z2fn2afo/b3/update.msi']

Name

https://zeltser.com/media/docs/malware-analysis-lab.pdf

Pattern Type

stix

Pattern

[url:value = 'https://zeltser.com/media/docs/malware-analysis-lab.pdf']

Name

lazagrc3cnk.xyz

Pattern Type

stix

Pattern

[domain-name:value = 'lazagrc3cnk.xyz']

Name

b287c0bc239b434b90eef01bcbd00ff48192b7cbeb540e568b8cdcdc26f90959

Description

stack_string

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = 'b287c0bc239b434b90eef01bcbd00ff48192b7cbeb540e568b8cdcdc26f90959']

Name

costexcise.xyz

Pattern Type

stix

Pattern

```
[domain-name:value = 'costexcise.xyz']
```

Malware

Name
Fake Update
Name
SectopRAT
Name
Stealc
Name
Lumma
Name
SocGholish
Name
Amadey
Description

[Amadey](https://attack.mitre.org/software/S1025) is a Trojan bot that has been used since at least October 2018.(Citation: Korean FSI TA505 2020)(Citation: BlackBerry Amadey 2020)

Attack-Pattern

Name

Virtualization/Sandbox Evasion

ID

T1497

Description

Adversaries may employ various means to detect and avoid virtualization and analysis environments. This may include changing behaviors based on the results of checks for the presence of artifacts indicative of a virtual machine environment (VME) or sandbox. If the adversary detects a VME, they may alter their malware to disengage from the victim or conceal the core functions of the implant. They may also search for VME artifacts before dropping secondary or additional payloads. Adversaries may use the information learned from [Virtualization/Sandbox Evasion](https://attack.mitre.org/techniques/T1497) during automated discovery to shape follow-on behaviors.(Citation: Deloitte Environment Awareness) Adversaries may use several methods to accomplish [Virtualization/Sandbox Evasion](https://attack.mitre.org/techniques/T1497) such as checking for security monitoring tools (e.g., Sysinternals, Wireshark, etc.) or other system artifacts associated with analysis or virtualization. Adversaries may also check for legitimate user activity to help determine if it is in an analysis environment. Additional methods include use of sleep timers or loops within malware code to avoid operating within a temporary sandbox. (Citation: Unit 42 Pirpi July 2015)

Name

Masquerading

ID

T1036

Description

Adversaries may attempt to manipulate features of their artifacts to make them appear legitimate or benign to users and/or security tools. Masquerading occurs when the name or location of an object, legitimate or malicious, is manipulated or abused for the sake of evading defenses and observation. This may include manipulating file metadata, tricking users into misidentifying the file type, and giving legitimate task or service names. Renaming abusable system utilities to evade security monitoring is also a form of [Masquerading](https://attack.mitre.org/techniques/T1036).(Citation: LOLBAS Main Site)

Name

Process Injection

ID

T1055

Description

Adversaries may inject code into processes in order to evade process-based defenses as well as possibly elevate privileges. Process injection is a method of executing arbitrary code in the address space of a separate live process. Running code in the context of another process may allow access to the process's memory, system/network resources, and possibly elevated privileges. Execution via process injection may also evade detection from security products since the execution is masked under a legitimate process. There are many different ways to inject code into a process, many of which abuse legitimate functionalities. These implementations exist for every major OS but are typically platform specific. More sophisticated samples may perform multiple process injections to segment modules and further evade detection, utilizing named pipes or other inter-process communication (IPC) mechanisms as a communication channel.

Name

Phishing

ID

T1566

Description

Adversaries may send phishing messages to gain access to victim systems. All forms of phishing are electronically delivered social engineering. Phishing can be targeted, known as spearphishing. In spearphishing, a specific individual, company, or industry will be targeted by the adversary. More generally, adversaries can conduct non-targeted phishing, such as in mass malware spam campaigns. Adversaries may send victims emails containing malicious attachments or links, typically to execute malicious code on victim systems. Phishing may also be conducted via third-party services, like social media platforms. Phishing may also involve social engineering techniques, such as posing as a trusted source, as well as evasive techniques such as removing or manipulating emails or metadata/headers from compromised accounts being abused to send messages (e.g., [Email Hiding Rules](https://attack.mitre.org/techniques/T1564/008)).(Citation: Microsoft OAuth Spam 2022)(Citation: Palo Alto Unit 42 VBA Infostealer 2014) Another way to accomplish this is by forging or spoofing(Citation: Proofpoint-spoof) the identity of the sender which can be used to fool both the human recipient as well as automated security tools.(Citation: cyberproof-double-bounce) Victims may also receive phishing messages that instruct them to call a phone number where they are directed to visit a malicious URL, download malware,(Citation: sygnia Luna Month)(Citation: CISA Remote Monitoring and Management Software) or install adversary-accessible remote management tools onto their computer (i.e., [User Execution](https://attack.mitre.org/techniques/T1204)).(Citation: Unit42 Luna Moth)

Name

User Execution

ID

T1204

Description

An adversary may rely upon specific actions by a user in order to gain execution. Users may be subjected to social engineering to get them to execute malicious code by, for example, opening a malicious document file or link. These user actions will typically be observed as follow-on behavior from forms of [Phishing](https://attack.mitre.org/techniques/T1566). While [User Execution](https://attack.mitre.org/techniques/T1204) frequently occurs shortly after Initial Access it may occur at other phases of an intrusion, such as when an adversary places a file in a shared directory or on a user's desktop hoping that a user will click on it. This activity may also be seen shortly after [Internal Spearphishing](https://attack.mitre.org/techniques/T1534). Adversaries may also deceive users into performing actions such as enabling [Remote Access Software](https://attack.mitre.org/techniques/T1219), allowing direct control of the system to the adversary, or downloading and executing malware for [User Execution](https://attack.mitre.org/techniques/T1204). For example, tech support scams can be facilitated through [Phishing](https://attack.mitre.org/techniques/T1566), vishing, or various forms of user interaction. Adversaries can use a combination of these methods, such as spoofing and promoting toll-free numbers or call centers that are used to direct victims to malicious websites, to deliver and execute payloads containing malware or [Remote Access Software](https://attack.mitre.org/techniques/T1219).(Citation: Telephone Attack Delivery)

Name

Native API

ID

T1106

Description

Adversaries may interact with the native OS application programming interface (API) to execute behaviors. Native APIs provide a controlled means of calling low-level OS services within the kernel, such as those involving hardware/devices, memory, and processes. (Citation: NT API Windows)(Citation: Linux Kernel API) These native APIs are leveraged by the OS during system boot (when other system components are not yet initialized) as well as carrying out tasks and requests during routine operations. Native API functions (such as `NtCreateProcess`) may be directed invoked via system calls / syscalls, but these features are also often exposed to user-mode applications via interfaces and libraries.(Citation: OutFlank System Calls)(Citation: CyberBit System Calls)(Citation: MDSec System Calls) For example, functions such as the Windows API `CreateProcess()` or GNU `fork()` will allow programs and scripts to start other processes.(Citation: Microsoft CreateProcess)(Citation:

GNU Fork) This may allow API callers to execute a binary, run a CLI command, load modules, etc. as thousands of similar API functions exist for various system operations. (Citation: Microsoft Win32)(Citation: LIBC)(Citation: GLIBC) Higher level software frameworks, such as Microsoft .NET and macOS Cocoa, are also available to interact with native APIs. These frameworks typically provide language wrappers/abstractions to API functionalities and are designed for ease-of-use/portability of code.(Citation: Microsoft NET)(Citation: Apple Core Services)(Citation: MACOS Cocoa)(Citation: macOS Foundation) Adversaries may abuse these OS API functions as a means of executing behaviors. Similar to [Command and Scripting Interpreter](https://attack.mitre.org/techniques/T1059), the native API and its hierarchy of interfaces provide mechanisms to interact with and utilize various components of a victimized system. While invoking API functions, adversaries may also attempt to bypass defensive tools (ex: unhooking monitored functions via [Disable or Modify Tools](https://attack.mitre.org/techniques/T1562/001)).

Name

Obfuscated Files or Information

ID

T1027

Description

Adversaries may attempt to make an executable or file difficult to discover or analyze by encrypting, encoding, or otherwise obfuscating its contents on the system or in transit. This is common behavior that can be used across different platforms and the network to evade defenses. Payloads may be compressed, archived, or encrypted in order to avoid detection. These payloads may be used during Initial Access or later to mitigate detection. Sometimes a user's action may be required to open and [Deobfuscate/Decode Files or Information](https://attack.mitre.org/techniques/T1140) for [User Execution](https://attack.mitre.org/techniques/T1204). The user may also be required to input a password to open a password protected compressed/encrypted file that was provided by the adversary. (Citation: Volexity PowerDuke November 2016) Adversaries may also use compressed or archived scripts, such as JavaScript. Portions of files can also be encoded to hide the plain-text strings that would otherwise help defenders with discovery. (Citation: Linux/Cdorked.A We Live Security Analysis) Payloads may also be split into separate, seemingly benign files that only reveal malicious functionality when reassembled. (Citation: Carbon Black Obfuscation Sept 2016) Adversaries may also abuse [Command Obfuscation](https://attack.mitre.org/techniques/T1027/010) to obscure commands executed from payloads or directly via [Command and Scripting Interpreter](https://attack.mitre.org/techniques/

T1059). Environment variables, aliases, characters, and other platform/language specific semantics can be used to evade signature based detections and application control mechanisms. (Citation: FireEye Obfuscation June 2017) (Citation: FireEye Revoke-Obfuscation July 2017)(Citation: PaloAlto EncodedCommand March 2017)

Name

Hijack Execution Flow

ID

T1574

Description

Adversaries may execute their own malicious payloads by hijacking the way operating systems run programs. Hijacking execution flow can be for the purposes of persistence, since this hijacked execution may reoccur over time. Adversaries may also use these mechanisms to elevate privileges or evade defenses, such as application control or other restrictions on execution. There are many ways an adversary may hijack the flow of execution, including by manipulating how the operating system locates programs to be executed. How the operating system locates libraries to be used by a program can also be intercepted. Locations where the operating system looks for programs/resources, such as file directories and in the case of Windows the Registry, could also be poisoned to include malicious payloads.

Name

Drive-by Compromise

ID

T1189

Description

Adversaries may gain access to a system through a user visiting a website over the normal course of browsing. With this technique, the user's web browser is typically targeted for

exploitation, but adversaries may also use compromised websites for non-exploitation behavior such as acquiring [Application Access Token](https://attack.mitre.org/techniques/T1550/001). Multiple ways of delivering exploit code to a browser exist (i.e., [Drive-by Target](https://attack.mitre.org/techniques/T1608/004)), including: * A legitimate website is compromised where adversaries have injected some form of malicious code such as JavaScript, iFrames, and cross-site scripting * Script files served to a legitimate website from a publicly writeable cloud storage bucket are modified by an adversary * Malicious ads are paid for and served through legitimate ad providers (i.e., [Malvertising](https://attack.mitre.org/techniques/T1583/008)) * Built-in web application interfaces are leveraged for the insertion of any other kind of object that can be used to display web content or contain a script that executes on the visiting client (e.g. forum posts, comments, and other user controllable web content). Often the website used by an adversary is one visited by a specific community, such as government, a particular industry, or region, where the goal is to compromise a specific user or set of users based on a shared interest. This kind of targeted campaign is often referred to a strategic web compromise or watering hole attack. There are several known examples of this occurring.(Citation: Shadowserver Strategic Web Compromise) Typical drive-by compromise process: 1. A user visits a website that is used to host the adversary controlled content. 2. Scripts automatically execute, typically searching versions of the browser and plugins for a potentially vulnerable version. * The user may be required to assist in this process by enabling scripting or active website components and ignoring warning dialog boxes. 3. Upon finding a vulnerable version, exploit code is delivered to the browser. 4. If exploitation is successful, then it will give the adversary code execution on the user's system unless other protections are in place. * In some cases a second visit to the website after the initial scan is required before exploit code is delivered. Unlike [Exploit Public-Facing Application](https://attack.mitre.org/techniques/T1190), the focus of this technique is to exploit software on a client endpoint upon visiting a website. This will commonly give an adversary access to systems on the internal network instead of external systems that may be in a DMZ. Adversaries may also use compromised websites to deliver a user to a malicious application designed to [Steal Application Access Token](https://attack.mitre.org/techniques/T1528)s, like OAuth tokens, to gain access to protected applications and information. These malicious applications have been delivered through popups on legitimate websites.(Citation: Volexity OceanLotus Nov 2017)

Name

Deobfuscate/Decode Files or Information

ID

T1140

Description

Adversaries may use [Obfuscated Files or Information](https://attack.mitre.org/techniques/T1027) to hide artifacts of an intrusion from analysis. They may require separate mechanisms to decode or deobfuscate that information depending on how they intend to use it. Methods for doing that include built-in functionality of malware or by using utilities present on the system. One such example is the use of [certutil](https://attack.mitre.org/software/S0160) to decode a remote access tool portable executable file that has been hidden inside a certificate file.(Citation: Malwarebytes Targeted Attack against Saudi Arabia) Another example is using the Windows `copy /b` command to reassemble binary fragments into a malicious payload.(Citation: Carbon Black Obfuscation Sept 2016) Sometimes a user's action may be required to open it for deobfuscation or decryption as part of [User Execution](https://attack.mitre.org/techniques/T1204). The user may also be required to input a password to open a password protected compressed/encrypted file that was provided by the adversary. (Citation: Volexity PowerDuke November 2016)

Name

System Binary Proxy Execution

ID

T1218

Description

Adversaries may bypass process and/or signature-based defenses by proxying execution of malicious content with signed, or otherwise trusted, binaries. Binaries used in this technique are often Microsoft-signed files, indicating that they have been either downloaded from Microsoft or are already native in the operating system.(Citation: LOLBAS Project) Binaries signed with trusted digital certificates can typically execute on Windows systems protected by digital signature validation. Several Microsoft signed binaries that are default on Windows installations can be used to proxy execution of other files or commands. Similarly, on Linux systems adversaries may abuse trusted binaries such as `split` to proxy execution of malicious commands.(Citation: split man page)(Citation: GTFO split)

Domain-Name

Value
ooinonqnbdqnjdnqwqkdn.space
bgobgogimrihehmxerreg.site
lazagrc3cnk.xyz
doorblu.xyz
omdowqind.site
costexcise.xyz
weomfewnfnu.site
buyerbrand.xyz
gstatic-node.io
ocmtancmi2c5t.xyz
winextrabonus.life
gapi-node.io

StixFile

Value
b3d8bc93a96c992099d768beb42202b48a7fe4c9a1e3b391efbeeb1549ef5039
3bf4b365d61c1e9807d20e71375627450b8fea1635cb6ddb85f2956e8f6b3ec3
a0319e612de3b7e6fbb4b71aa7398266791e50da0ae373c5870c3dcaa51abccf
d19c166d0846ddaf1a6d5dbd62c93acb91956627e47e4e3cbd79f3dfb3e0f002
5f57537d18adcc1142294d7c469f565f359d5ff148e93a15ccbceb5ca3390dbd
51cee2de0ebe01e75afdeffe29d48cb4d413d471766420c8b8f9ab08c59977d7
c9094685ae4851fd5a5b886b73c7b07efd9b47ea0bdae3f823d035cf1b3b9e48
53c3982f452e570db6599e004d196a8a3b8399c9d484f78cdb481c2703138d47
931d78c733c6287cec991659ed16513862bfc6f5e42b74a8a82e4fa6c8a3fe06
be8eb5359185baa8e456a554a091ec54c8828bb2499fe332e9ecd65639c9a75b
b287c0bc239b434b90eef01bcbd00ff48192b7cbeb540e568b8cdcdc26f90959
8ce0901a5cf2d3014aaa89d5b5b68666da0d42d2294a2f2b7e3a275025b35b79

IPv4-Addr

Value
94.228.169.55

Url

Value
https://zeltser.com/media/docs/malware-analysis-lab.pdf
https://ocmtancmi2c5t.xyz/82z2fn2afo/b3/update.msi

External References

- https://otx.alienvault.com/pulse/64f1e91a2dd9db4bd3af8ce4
- https://www.rapid7.com/blog/post/2023/08/31/fake-update-utilizes-new-idat-loader-to-execute-stealc-and-lumma-infostealers/