Jun 11 2024

NETMANAGEIT

Intelligence Report

Dissecting SSLoad Malware: A Comprehensive Technical Analysis

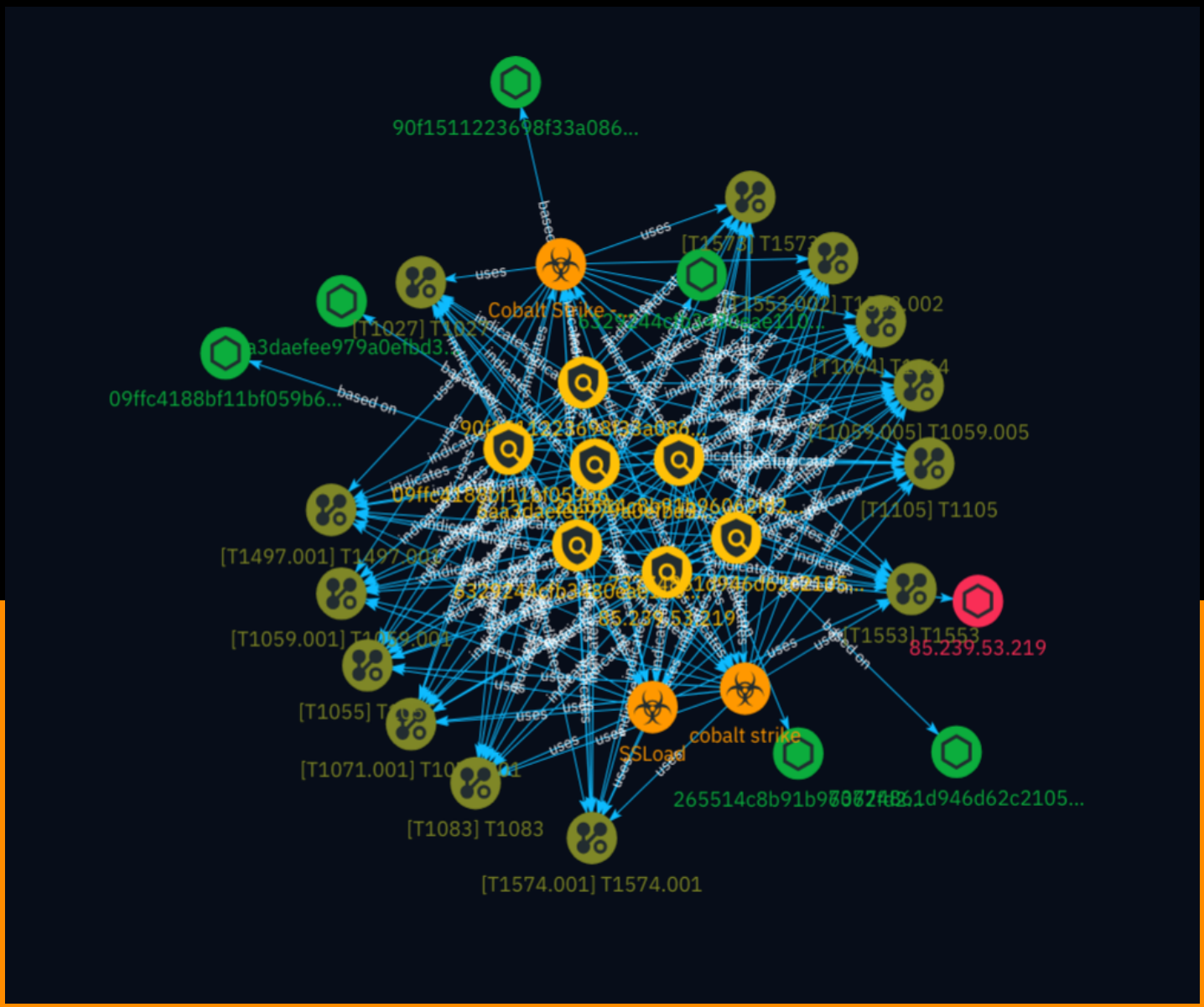

Table of contents

Overview

Entities

Observables

External References

Overview

Description

This in-depth analysis explores the intricate inner workings of SSLoad, a stealthy and adaptable malware known for its sophisticated delivery methods and evasion techniques. The comprehensive investigation unravels the malware's multistage infection chain, dissecting the various loaders, decryption algorithms, and payloads employed across different campaigns. The analysis highlights SSLoad's ability to gather reconnaissance, evade detection, and deploy additional malicious components, underscoring its versatility and ever-evolving nature.

Confidence

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

100 / 100

Content

N/A

Attack-Pattern

Name
T1064
ID
T1064
Description
This technique has been deprecated. Please use [Command and Scripting Interpreter](https://attack.mitre.org/techniques/T1059) where appropriate. Adversaries may use scripts to aid in operations and perform multiple actions that would otherwise be manual. Scripting is useful for speeding up operational tasks and reducing the time required to gain access to critical resources. Some scripting languages may be used to bypass process monitoring mechanisms by directly interacting with the operating system at an API level instead of calling other programs. Common scripting languages for Windows include VBScript and [PowerShell](https://attack.mitre.org/techniques/T1086) but could also be in the form of command-line batch scripts. Scripts can be embedded inside Office documents as macros that can be set to execute when files used in [Spearphishing Attachment](https://attack.mitre.org/techniques/T1193) and other types of spearphishing are opened. Malicious embedded macros are an alternative means of execution than software exploitation through [Exploitation for Client Execution](https://attack.mitre.org/techniques/T1203), where adversaries will rely on macros being allowed or that the user will accept to activate them. Many popular offensive frameworks exist which use forms of scripting for security testers and adversaries alike. Metasploit (Citation: Metasploit_Ref), Veil (Citation: Veil_Ref), and PowerSploit (Citation: Powersploit) are three examples that are popular among penetration testers for exploit and post-compromise operations and include many features for evading defenses. Some adversaries are known to use PowerShell. (Citation: Alperovitch 2014)

Name

T1055

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

T1573

ID

T1573

Description

Adversaries may employ a known encryption algorithm to conceal command and control traffic rather than relying on any inherent protections provided by a communication protocol. Despite the use of a secure algorithm, these implementations may be vulnerable to reverse engineering if secret keys are encoded and/or generated within malware samples/configuration files.

Name

T1071.001

ID

T1071.001

Description

Adversaries may communicate using application layer protocols associated with web traffic to avoid detection/network filtering by blending in with existing traffic. Commands to the remote system, and often the results of those commands, will be embedded within the protocol traffic between the client and server. Protocols such as HTTP/S(Citation: CrowdStrike Putter Panda) and WebSocket(Citation: Brazking-Websockets) that carry web traffic may be very common in environments. HTTP/S packets have many fields and headers in which data can be concealed. An adversary may abuse these protocols to communicate with systems under their control within a victim network while also mimicking normal, expected traffic.

Name

T1059.001

ID

T1059.001

Description

Adversaries may abuse PowerShell commands and scripts for execution. PowerShell is a powerful interactive command-line interface and scripting environment included in the Windows operating system.(Citation: TechNet PowerShell) Adversaries can use PowerShell to perform a number of actions, including discovery of information and execution of code. Examples include the `Start-Process` cmdlet which can be used to run an executable and the `Invoke-Command` cmdlet which runs a command locally or on a remote computer (though administrator permissions are required to use PowerShell to connect to remote systems). PowerShell may also be used to download and run executables from the

Internet, which can be executed from disk or in memory without touching disk. A number of PowerShell-based offensive testing tools are available, including [Empire](https://attack.mitre.org/software/S0363), [PowerSploit](https://attack.mitre.org/software/S0194), [PoshC2](https://attack.mitre.org/software/S0378), and PSAttack.(Citation: Github PSAttack) PowerShell commands/scripts can also be executed without directly invoking the `powershell.exe` binary through interfaces to PowerShell's underlying `System.Management.Automation` assembly DLL exposed through the .NET framework and Windows Common Language Interface (CLI).(Citation: Sixdub PowerPick Jan 2016)(Citation: SilentBreak Offensive PS Dec 2015)(Citation: Microsoft PSfromCsharp APR 2014)

Name

T1553

ID

T1553

Description

Adversaries may undermine security controls that will either warn users of untrusted activity or prevent execution of untrusted programs. Operating systems and security products may contain mechanisms to identify programs or websites as possessing some level of trust. Examples of such features would include a program being allowed to run because it is signed by a valid code signing certificate, a program prompting the user with a warning because it has an attribute set from being downloaded from the Internet, or getting an indication that you are about to connect to an untrusted site. Adversaries may attempt to subvert these trust mechanisms. The method adversaries use will depend on the specific mechanism they seek to subvert. Adversaries may conduct [File and Directory Permissions Modification](https://attack.mitre.org/techniques/T1222) or [Modify Registry](https://attack.mitre.org/techniques/T1112) in support of subverting these controls. (Citation: SpectorOps Subverting Trust Sept 2017) Adversaries may also create or steal code signing certificates to acquire trust on target systems.(Citation: Securelist Digital Certificates)(Citation: Symantec Digital Certificates)

Name

T1027

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

T1105

ID

T1105

Description

Adversaries may transfer tools or other files from an external system into a compromised environment. Tools or files may be copied from an external adversary-controlled system to the victim network through the command and control channel or through alternate protocols such as [ftp](https://attack.mitre.org/software/S0095). Once present, adversaries may also transfer/spread tools between victim devices within a compromised environment (i.e. [Lateral Tool Transfer](https://attack.mitre.org/techniques/T1570)). On Windows, adversaries may use various utilities to download tools, such as `copy`, `finger`, [certutil](https://attack.mitre.org/software/S0160), and [PowerShell](https://attack.mitre.org/techniques/T1059/001) commands such as `IEX(New-Object Net.WebClient).downloadString()` and `Invoke-WebRequest`. On Linux and macOS systems, a variety of utilities also exist, such as `curl`, `scp`, `sftp`, `tftp`, `rsync`, `finger`, and `wget`. (Citation: t1105_lolbas) Adversaries may also abuse installers and package managers, such as `yum` or `winget`, to download tools to victim hosts. Files can also be transferred using various [Web Service](https://attack.mitre.org/techniques/T1102)s as well as native or otherwise present tools on the victim system.(Citation: PTSecurity Cobalt Dec 2016) In some cases, adversaries may be able to leverage services that sync between a web-based and an on-premises client, such as Dropbox or OneDrive, to transfer files onto victim systems. For example, by compromising a cloud account and logging into the service's web portal, an adversary may be able to trigger an automatic syncing process that transfers the file onto the victim's machine.(Citation: Dropbox Malware Sync)

Name

T1059.005

ID

T1059.005

Description

Adversaries may abuse Visual Basic (VB) for execution. VB is a programming language created by Microsoft with interoperability with many Windows technologies such as [Component Object Model](https://attack.mitre.org/techniques/T1559/001) and the [Native API](https://attack.mitre.org/techniques/T1106) through the Windows API. Although tagged as legacy with no planned future evolutions, VB is integrated and supported in the .NET Framework and cross-platform .NET Core.(Citation: VB .NET Mar 2020)(Citation: VB Microsoft) Derivative languages based on VB have also been created, such as Visual Basic for Applications (VBA) and VBScript. VBA is an event-driven programming language built into Microsoft Office, as well as several third-party applications.(Citation: Microsoft VBA)

(Citation: Wikipedia VBA) VBA enables documents to contain macros used to automate the execution of tasks and other functionality on the host. VBScript is a default scripting language on Windows hosts and can also be used in place of [JavaScript](https://attack.mitre.org/techniques/T1059/007) on HTML Application (HTA) webpages served to Internet Explorer (though most modern browsers do not come with VBScript support). (Citation: Microsoft VBScript) Adversaries may use VB payloads to execute malicious commands. Common malicious usage includes automating execution of behaviors with VBScript or embedding VBA content into [Spearphishing Attachment](https://attack.mitre.org/techniques/T1566/001) payloads (which may also involve [Mark-of-the-Web Bypass](https://attack.mitre.org/techniques/T1553/005) to enable execution).(Citation: Default VBS macros Blocking)

Name

T1497.001

ID

T1497.001

Description

Adversaries may employ various system checks 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) Specific checks will vary based on the target and/or adversary, but may involve behaviors such as [Windows Management Instrumentation](https://attack.mitre.org/techniques/T1047), [PowerShell](https://attack.mitre.org/techniques/T1059/001), [System Information Discovery](https://attack.mitre.org/techniques/T1082), and [Query Registry](https://attack.mitre.org/techniques/T1012) to obtain system information and search for VME artifacts. Adversaries may search for VME artifacts in memory, processes, file system, hardware, and/or the Registry. Adversaries may use scripting to automate these checks into one script and then have the program exit if it determines the system to be a virtual environment. Checks could include generic system properties such as host/domain name and samples of network traffic. Adversaries may also check the network adapters addresses, CPU core count, and available memory/drive

size. Other common checks may enumerate services running that are unique to these applications, installed programs on the system, manufacturer/product fields for strings relating to virtual machine applications, and VME-specific hardware/processor instructions.(Citation: McAfee Virtual Jan 2017) In applications like VMWare, adversaries can also use a special I/O port to send commands and receive output. Hardware checks, such as the presence of the fan, temperature, and audio devices, could also be used to gather evidence that can be indicative a virtual environment. Adversaries may also query for specific readings from these devices.(Citation: Unit 42 OilRig Sept 2018)

Name

T1553.002

ID

T1553.002

Description

Adversaries may create, acquire, or steal code signing materials to sign their malware or tools. Code signing provides a level of authenticity on a binary from the developer and a guarantee that the binary has not been tampered with. (Citation: Wikipedia Code Signing) The certificates used during an operation may be created, acquired, or stolen by the adversary. (Citation: Securelist Digital Certificates) (Citation: Symantec Digital Certificates) Unlike [Invalid Code Signature](https://attack.mitre.org/techniques/T1036/001), this activity will result in a valid signature. Code signing to verify software on first run can be used on modern Windows and macOS systems. It is not used on Linux due to the decentralized nature of the platform. (Citation: Wikipedia Code Signing)(Citation: EclecticLightChecksonEXECodeSigning) Code signing certificates may be used to bypass security policies that require signed code to execute on a system.

Name

T1083

ID

T1083

Description

Adversaries may enumerate files and directories or may search in specific locations of a host or network share for certain information within a file system. Adversaries may use the information from [File and Directory Discovery](https://attack.mitre.org/techniques/T1083) during automated discovery to shape follow-on behaviors, including whether or not the adversary fully infects the target and/or attempts specific actions. Many command shell utilities can be used to obtain this information. Examples include `dir`, `tree`, `ls`, `find`, and `locate`.(Citation: Windows Commands JPCERT) Custom tools may also be used to gather file and directory information and interact with the [Native API](https://attack.mitre.org/techniques/T1106). Adversaries may also leverage a [Network Device CLI](https://attack.mitre.org/techniques/T1059/008) on network devices to gather file and directory information (e.g. `dir`, `show flash`, and/or `nvram`).(Citation: US-CERT-TA18-106A)

Name

T1574.001

ID

T1574.001

Description

Adversaries may execute their own malicious payloads by hijacking the search order used to load DLLs. Windows systems use a common method to look for required DLLs to load into a program. (Citation: Microsoft Dynamic Link Library Search Order)(Citation: FireEye Hijacking July 2010) Hijacking DLL loads may be for the purpose of establishing persistence as well as elevating privileges and/or evading restrictions on file execution. There are many ways an adversary can hijack DLL loads. Adversaries may plant trojan dynamic-link library files (DLLs) in a directory that will be searched before the location of a legitimate library that will be requested by a program, causing Windows to load their malicious library when it is called for by the victim program. Adversaries may also perform DLL preloading, also called binary planting attacks, (Citation: OWASP Binary Planting) by placing a malicious DLL with the same name as an ambiguously specified DLL in a location that Windows searches before the legitimate DLL. Often this location is the current working directory of the program.(Citation: FireEye fxsst June 2011) Remote DLL preloading attacks occur when a program sets its current directory to a remote location such as a Web share before loading a DLL. (Citation: Microsoft Security Advisory 2269637) Adversaries may also directly modify the search order via DLL redirection, which after being enabled

(in the Registry and creation of a redirection file) may cause a program to load a different DLL.(Citation: Microsoft Dynamic-Link Library Redirection)(Citation: Microsoft Manifests) (Citation: FireEye DLL Search Order Hijacking) If a search order-vulnerable program is configured to run at a higher privilege level, then the adversary-controlled DLL that is loaded will also be executed at the higher level. In this case, the technique could be used for privilege escalation from user to administrator or SYSTEM or from administrator to SYSTEM, depending on the program. Programs that fall victim to path hijacking may appear to behave normally because malicious DLLs may be configured to also load the legitimate DLLs they were meant to replace.

Indicator

Name
90f1511223698f33a086337a6875db3b5d6fbcce06f3195cdd6a8efa90091750
Pattern Type
stix
Pattern
[file:hashes.'SHA-256' = '90f1511223698f33a086337a6875db3b5d6fbcce06f3195cdd6a8efa90091750']
Name
09ffc4188bf11bf059b616491fcb8a09a474901581f46ec7f2c350fbda4e1e1c
Pattern Type
stix
Pattern
[file:hashes.'SHA-256' = '09ffc4188bf11bf059b616491fcb8a09a474901581f46ec7f2c350fbda4e1e1c']
Name

6aa3daefee979a0efbd30de15a1fc7c0d05a6e8e3f439d5af3982878c3901a1c

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = '6aa3daefee979a0efbd30de15a1fc7c0d05a6e8e3f439d5af3982878c3901a1c']

Name

6329244cfb3480eae11070f1aa880bff2fd52b374e12ac37f1eacb6379c72b80

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = '6329244cfb3480eae11070f1aa880bff2fd52b374e12ac37f1eacb6379c72b80']

Name

265514c8b91b96062fd2960d52ee09d67ea081c56ebadd7a8661f479124133e9

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = '265514c8b91b96062fd2960d52ee09d67ea081c56ebadd7a8661f479124133e9']

Name

73774861d946d62c2105fef4718683796cb77de7ed42edaec7affcee5eb0a0ee

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = '73774861d946d62c2105fef4718683796cb77de7ed42edaec7affcee5eb0a0ee']

Name

85.239.53.219

Description

ISP: BlueVPS OU **OS:** - ---------- Services: **80:** ``` HTTP/1.1 200 OK Server: nginx Date: Fri, 07 Jun 2024 04:05:32 GMT Content-Type: text/html Content-Length: 612 Last-Modified: Thu, 04 Apr 2024 13:03:28 GMT Connection: keep-alive ETag: "660ea520-264" Accept-Ranges: bytes ``` ----------

Pattern Type

stix

Pattern

[ipv4-addr:value = '85.239.53.219']

Malware

Name
Cobalt Strike - S0154
Name
SSLoad
Name
cobalt strike
Description
[Cobalt Strike](https://attack.mitre.org/software/S0154) is a commercial, full-featured, remote access tool that bills itself as “adversary simulation software designed to execute targeted attacks and emulate the post-exploitation actions of advanced threat actors”. Cobalt Strike’s interactive post-exploit capabilities cover the full range of ATT&CK tactics, all executed within a single, integrated system.(Citation: cobaltstrike manual) In addition to its own capabilities, [Cobalt Strike](https://attack.mitre.org/software/S0154) leverages the capabilities of other well-known tools such as Metasploit and [Mimikatz](https://attack.mitre.org/software/S0002).(Citation: cobaltstrike manual)

uses

Name
Name
Name
Description
[Cobalt Strike](https://attack.mitre.org/software/S0154) can use self signed Java applets to execute signed applet attacks.(Citation: Talos Cobalt Strike September 2020)(Citation: Cobalt Strike Manual 4.3 November 2020)
Name
Name
Name
Name

Description
[Cobalt Strike](https://attack.mitre.org/software/S0154) can hash functions to obfuscate calls to the Windows API and use a public/private key pair to encrypt Beacon session metadata.(Citation: Talos Cobalt Strike September 2020)(Citation: Cobalt Strike Manual 4.3 November 2020)

Name

Name

Name

Name

Name

Name

Name

Name

Description

[Cobalt Strike](https://attack.mitre.org/software/S0154) can inject a variety of payloads into processes dynamically chosen by the adversary.(Citation: cobaltstrike manual) (Citation: Cobalt Strike Manual 4.3 November 2020)(Citation: DFIR Conti Bazar Nov 2021)

Name

Name

Description

[Cobalt Strike](https://attack.mitre.org/software/S0154) can use VBA to perform execution. (Citation: Cobalt Strike TTPs Dec 2017)(Citation: CobaltStrike Daddy May 2017)

Name

Description

[Cobalt Strike](https://attack.mitre.org/software/S0154) can execute a payload on a remote host with PowerShell. This technique does not write any data to disk.(Citation: cobaltstrike manual)(Citation: Cyberreason Anchor December 2019) [Cobalt Strike](https://attack.mitre.org/software/S0154) can also use [PowerSploit](https://attack.mitre.org/software/S0194) and other scripting frameworks to perform execution.(Citation: Cobalt Strike TTPs Dec 2017)(Citation: CobaltStrike Daddy May 2017)

based-on

Name
Name
Name
Name
Name

indicates

Name
Name
Name
Name
Name
Name
Name
Name
Name

Name
Name
Name
Name
Name
Name
Name
Name
Name
Name
Name
Name

Name
Name
Name
Name
Name
Name
Name
Name
Name
Name
Name
Name

Name
Name
Name
Name
Name
Name
Name
Name
Name
Name
Name
Name

Name
Name
Name
Name
Name

StixFile

Value
09ffc4188bf11bf059b616491fcb8a09a474901581f46ec7f2c350fbda4e1e1c
90f1511223698f33a086337a6875db3b5d6fbcce06f3195cdd6a8efa90091750
6329244cfb3480eae11070f1aa880bff2fd52b374e12ac37f1eacb6379c72b80
6aa3daefee979a0efbd30de15a1fc7c0d05a6e8e3f439d5af3982878c3901a1c
265514c8b91b96062fd2960d52ee09d67ea081c56ebadd7a8661f479124133e9
73774861d946d62c2105fef4718683796cb77de7ed42edaec7affcee5eb0a0ee

IPv4-Addr

Value
85.239.53.219

External References

- https://intezer.com/blog/research/ssload-technical-malware-analysis/
- https://otx.alienvault.com/pulse/66672064dd2deec2c6d74a3c