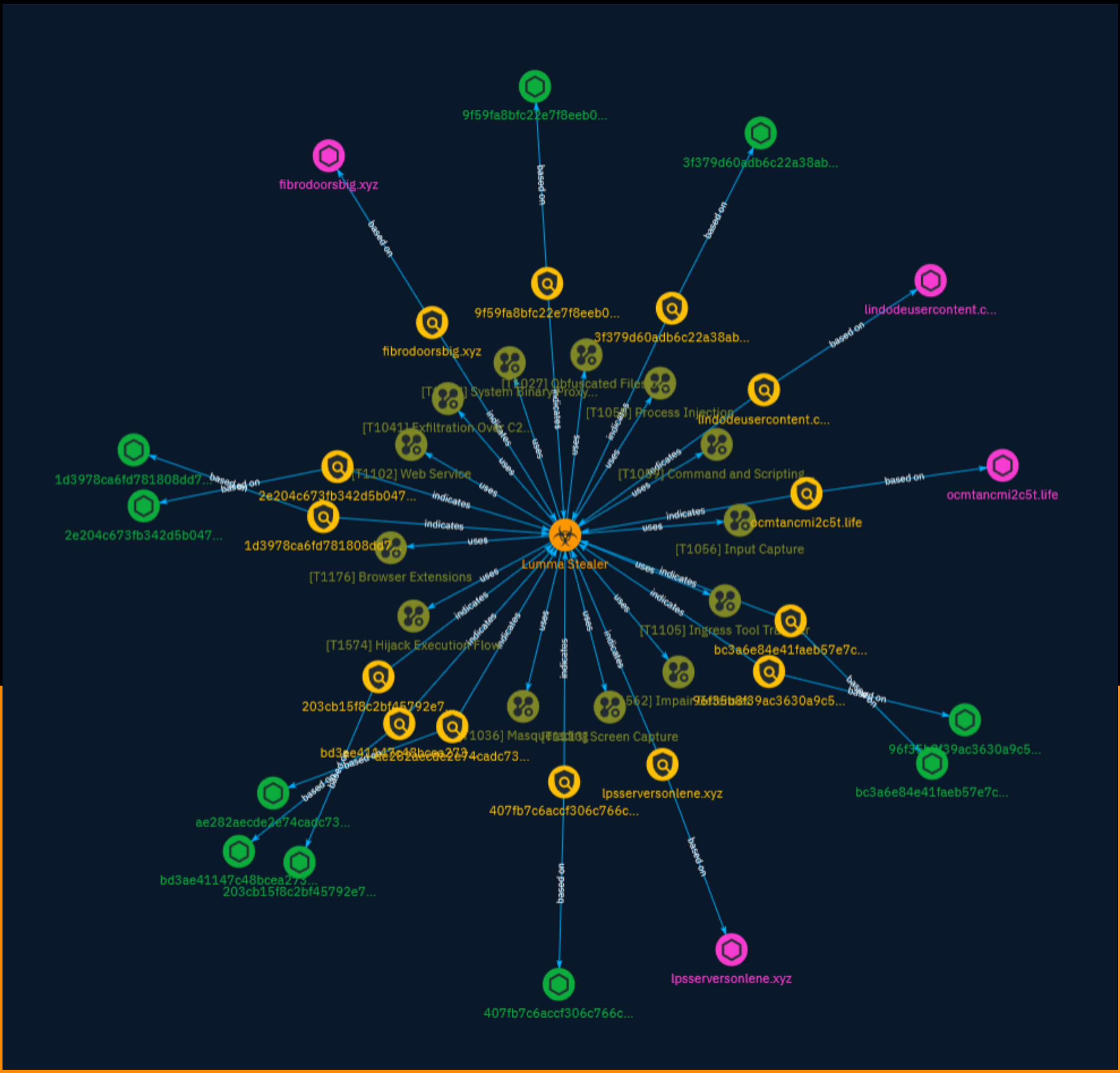
Oct 29 2023

NETMANAGEIT

# Intelligence Report

# An iLUMMAnation on LummaStealer



# Table of contents

## Overview

## Entities

## Observables

## External References

# Overview

## Description

Lumma Stealer is a malware that is being sold widely across the Dark Web and hacker forums, according to research by security researchers from the VMware Carbon Black team and the UK-based security firm.

## Confidence

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

15 / 100

# Content

N/A

# Attack-Pattern

**Name**

Input Capture

**ID**

T1056

**Description**

Adversaries may use methods of capturing user input to obtain credentials or collect information. During normal system usage, users often provide credentials to various different locations, such as login pages/portals or system dialog boxes. Input capture mechanisms may be transparent to the user (e.g. [Credential API Hooking](https://attack.mitre.org/techniques/T1056/004)) or rely on deceiving the user into providing input into what they believe to be a genuine service (e.g. [Web Portal Capture](https://attack.mitre.org/techniques/T1056/003)).

**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**

Impair Defenses

**ID**

T1562

## Description

Adversaries may maliciously modify components of a victim environment in order to hinder or disable defensive mechanisms. This not only involves impairing preventative defenses, such as firewalls and anti-virus, but also detection capabilities that defenders can use to audit activity and identify malicious behavior. This may also span both native defenses as well as supplemental capabilities installed by users and administrators. Adversaries may also impair routine operations that contribute to defensive hygiene, such as blocking users from logging out of a computer or stopping it from being shut down. These restrictions can further enable malicious operations as well as the continued propagation of incidents.(Citation: Emotet shutdown) Adversaries could also target event aggregation and analysis mechanisms, or otherwise disrupt these procedures by altering other system components.

## Name

Browser Extensions

## ID

T1176

## Description

Adversaries may abuse Internet browser extensions to establish persistent access to victim systems. Browser extensions or plugins are small programs that can add functionality and customize aspects of Internet browsers. They can be installed directly or through a browser's app store and generally have access and permissions to everything that the browser can access.(Citation: Wikipedia Browser Extension)(Citation: Chrome Extensions Definition) Malicious extensions can be installed into a browser through malicious app store downloads masquerading as legitimate extensions, through social engineering, or by an adversary that has already compromised a system. Security can be limited on browser app stores so it may not be difficult for malicious extensions to defeat automated scanners.(Citation: Malicious Chrome Extension Numbers) Depending on the browser, adversaries may also manipulate an extension's update url to install updates from an adversary controlled server or manipulate the mobile configuration file to silently install additional extensions. Previous to macOS 11, adversaries could silently install browser extensions via the command line using the `profiles` tool to install malicious `.mobileconfig` files. In macOS 11+, the use of the `profiles` tool can no longer install configuration profiles, however `.mobileconfig` files can be planted and installed with user interaction.(Citation: xorrior chrome extensions macOS) Once the extension is installed, it can browse to websites in the background, steal all information that a user enters into a browser (including credentials), and be used as an installer for a RAT for persistence. (Citation: Chrome Extension Crypto Miner)(Citation: ICEBRG Chrome Extensions)(Citation: Banker Google Chrome Extension Steals Creds)(Citation: Catch All Chrome Extension) There have also been instances of botnets using a persistent backdoor through malicious Chrome extensions.(Citation: Stantinko Botnet) There have also been similar examples of extensions being used for command & control.(Citation: Chrome Extension C2 Malware)

**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**

Ingress Tool Transfer

**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)). 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) 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)

**Name**

Command and Scripting Interpreter

**ID**

T1059

**Description**

Adversaries may abuse command and script interpreters to execute commands, scripts, or binaries. These interfaces and languages provide ways of interacting with computer systems and are a common feature across many different platforms. Most systems come with some built-in command-line interface and scripting capabilities, for example, macOS and Linux distributions include some flavor of [Unix Shell](https://attack.mitre.org/techniques/T1059/004) while Windows installations include the [Windows Command Shell](https://attack.mitre.org/techniques/T1059/003) and [PowerShell](https://attack.mitre.org/techniques/T1059/001). There are also cross-platform interpreters such as [Python](https://attack.mitre.org/techniques/T1059/006), as well as those commonly associated with client applications such as [JavaScript](https://attack.mitre.org/techniques/T1059/007) and [Visual Basic](https://attack.mitre.org/techniques/T1059/005). Adversaries may abuse these technologies in various ways as a means of executing arbitrary commands. Commands and scripts can be embedded in [Initial Access](https://attack.mitre.org/tactics/TA0001) payloads delivered to victims as lure documents or as secondary payloads downloaded from an existing C2. Adversaries may also execute commands through interactive terminals/shells, as well as utilize various [Remote Services](https://attack.mitre.org/techniques/T1021) in order to achieve remote Execution.(Citation: Powershell Remote Commands)(Citation: Cisco IOS Software Integrity Assurance - Command History)(Citation: Remote Shell Execution in Python)

**Name**

Web Service

**ID**

T1102

**Description**

Adversaries may use an existing, legitimate external Web service as a means for relaying data to/from a compromised system. Popular websites and social media acting as a mechanism for C2 may give a significant amount of cover due to the likelihood that hosts within a network are already communicating with them prior to a compromise. Using common services, such as those offered by Google or Twitter, makes it easier for adversaries to hide in expected noise. Web service providers commonly use SSL/TLS encryption, giving adversaries an added level of protection. Use of Web services may also protect back-end C2 infrastructure from discovery through malware binary analysis while also enabling operational resiliency (since this infrastructure may be dynamically changed).

**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) |
| **Name** |
| Screen Capture |
| **ID** |
| T1113 |
| **Description** |
| Adversaries may attempt to take screen captures of the desktop to gather information over the course of an operation. Screen capturing functionality may be included as a feature of a remote access tool used in post-compromise operations. Taking a screenshot is also typically possible through native utilities or API calls, such as `CopyFromScreen`, `xwd`, or `screencapture`.(Citation: CopyFromScreen .NET)(Citation: Antiquated Mac Malware) |
| **Name** |
| Exfiltration Over C2 Channel |
| **ID** |
| T1041 |
| **Description** |
| Adversaries may steal data by exfiltrating it over an existing command and control channel. Stolen data is encoded into the normal communications channel using the same protocol as command and control communications. |

# Indicator

**Name**

fibrodoorsbig.xyz

**Pattern Type**

stix

**Pattern**

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

**Name**

lpsserversonlene.xyz

**Pattern Type**

stix

**Pattern**

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

**Name**

203cb15f8c2bf45792e72bf75366e3eacf563a7470c66acce935c15f498c1806

**Pattern Type**

stix

**Pattern**

[file:hashes.'SHA-256' = '203cb15f8c2bf45792e72bf75366e3eacf563a7470c66acce935c15f498c1806']

**Name**

lindodeusercontent.com

**Pattern Type**

stix

**Pattern**

[domain-name:value = 'lindodeusercontent.com']

**Name**

407fb7c6accf306c766ccb68fb2247d3340fe5363cf991d16613af2ea46f8d0c

**Pattern Type**

stix

**Pattern**

[file:hashes.'SHA-256' = '407fb7c6accf306c766ccb68fb2247d3340fe5363cf991d16613af2ea46f8d0c']

**Name**

3f379d60adb6c22a38ab81052458d7ced3361185d92ea7afe6d7b5d812080b95

**Description**

stack_string

**Pattern Type**

stix

**Pattern**

[file:hashes.'SHA-256' =
'3f379d60adb6c22a38ab81052458d7ced3361185d92ea7afe6d7b5d812080b95']

**Name**

96f35b8f39ac3630a9c58f2621bb0cfce873b69c5a1c2a40612130076e07a533

**Pattern Type**

stix

**Pattern**

[file:hashes.'SHA-256' =
'96f35b8f39ac3630a9c58f2621bb0cfce873b69c5a1c2a40612130076e07a533']

**Name**

bd3ae41147c48bcea273f742ae19c229ad76c4a75895253e01a58bd4f3c2b9d1

**Description**

stack_string

| Pattern Type |
|---|
| stix |

| Pattern |
|---|
| [file:hashes.'SHA-256' = 'bd3ae41147c48bcea273f742ae19c229ad76c4a75895253e01a58bd4f3c2b9d1'] |

| Name |
|---|
| 9f59fa8bfc22e7f8eeb0aabf9ccea130eecd3a825822abd5e8e5347ba0c1402d |

| Pattern Type |
|---|
| stix |

| Pattern |
|---|
| [file:hashes.'SHA-256' = '9f59fa8bfc22e7f8eeb0aabf9ccea130eecd3a825822abd5e8e5347ba0c1402d'] |

| Name |
|---|
| ae282aecde2e74cadc73ee114e6760959686dc5ee99c608e5e4047766b5137d1 |

| Pattern Type |
|---|
| stix |

| Pattern |
|---|
| [file:hashes.'SHA-256' = 'ae282aecde2e74cadc73ee114e6760959686dc5ee99c608e5e4047766b5137d1'] |

| **Name** |
|---|
| 2e204c673fb342d5b0472a765ce8576487d83ed25957365f9df744d12ac04893 |
| **Pattern Type** |
| stix |
| **Pattern** |
| [file:hashes.'SHA-256' = '2e204c673fb342d5b0472a765ce8576487d83ed25957365f9df744d12ac04893'] |
| **Name** |
| bc3a6e84e41faeb57e7c21aa3b60c2a64777107009727c5b7c0ed8fe658909e5 |
| **Pattern Type** |
| stix |
| **Pattern** |
| [file:hashes.'SHA-256' = 'bc3a6e84e41faeb57e7c21aa3b60c2a64777107009727c5b7c0ed8fe658909e5'] |
| **Name** |
| 1d3978ca6fd781808dd7cea4cc31e622324a1f0ababd5020f597d2fcc9d378fd |
| **Pattern Type** |
| stix |
| **Pattern** |

[file:hashes.'SHA-256' = '1d3978ca6fd781808dd7cea4cc31e622324a1f0ababd5020f597d2fcc9d378fd']

**Name**

ocmtancmi2c5t.life

**Pattern Type**

stix

**Pattern**

[domain-name:value = 'ocmtancmi2c5t.life']

# Malware

| Name |
|---|
| Lumma Stealer |

# Domain-Name

| Value |
|---|
| ocmtancmi2c5t.life |
| fibrodoorsbig.xyz |
| lindodeusercontent.com |
| lpsserversonlene.xyz |

# StixFile

| Value |
|---|
| 3f379d60adb6c22a38ab81052458d7ced3361185d92ea7afe6d7b5d812080b95 |
| 2e204c673fb342d5b0472a765ce8576487d83ed25957365f9df744d12ac04893 |
| 1d3978ca6fd781808dd7cea4cc31e622324a1f0ababd5020f597d2fcc9d378fd |
| 203cb15f8c2bf45792e72bf75366e3eacf563a7470c66acce935c15f498c1806 |
| 9f59fa8bfc22e7f8eeb0aabf9ccea130eecd3a825822abd5e8e5347ba0c1402d |
| 407fb7c6accf306c766ccb68fb2247d3340fe5363cf991d16613af2ea46f8d0c |
| bc3a6e84e41faeb57e7c21aa3b60c2a64777107009727c5b7c0ed8fe658909e5 |
| ae282aecde2e74cadc73ee114e6760959686dc5ee99c608e5e4047766b5137d1 |
| 96f35b8f39ac3630a9c58f2621bb0cfce873b69c5a1c2a40612130076e07a533 |
| bd3ae41147c48bcea273f742ae19c229ad76c4a75895253e01a58bd4f3c2b9d1 |

# External References

- https://otx.alienvault.com/pulse/6537eac5a590aa7e2fbd6429
- https://blogs.vmware.com/security/2023/10/an-ilummanation-on-lummastealer.html