Feb 19 2024

NETMANAGEIT

Intelligence Report

Technical Analysis of DarkVNC

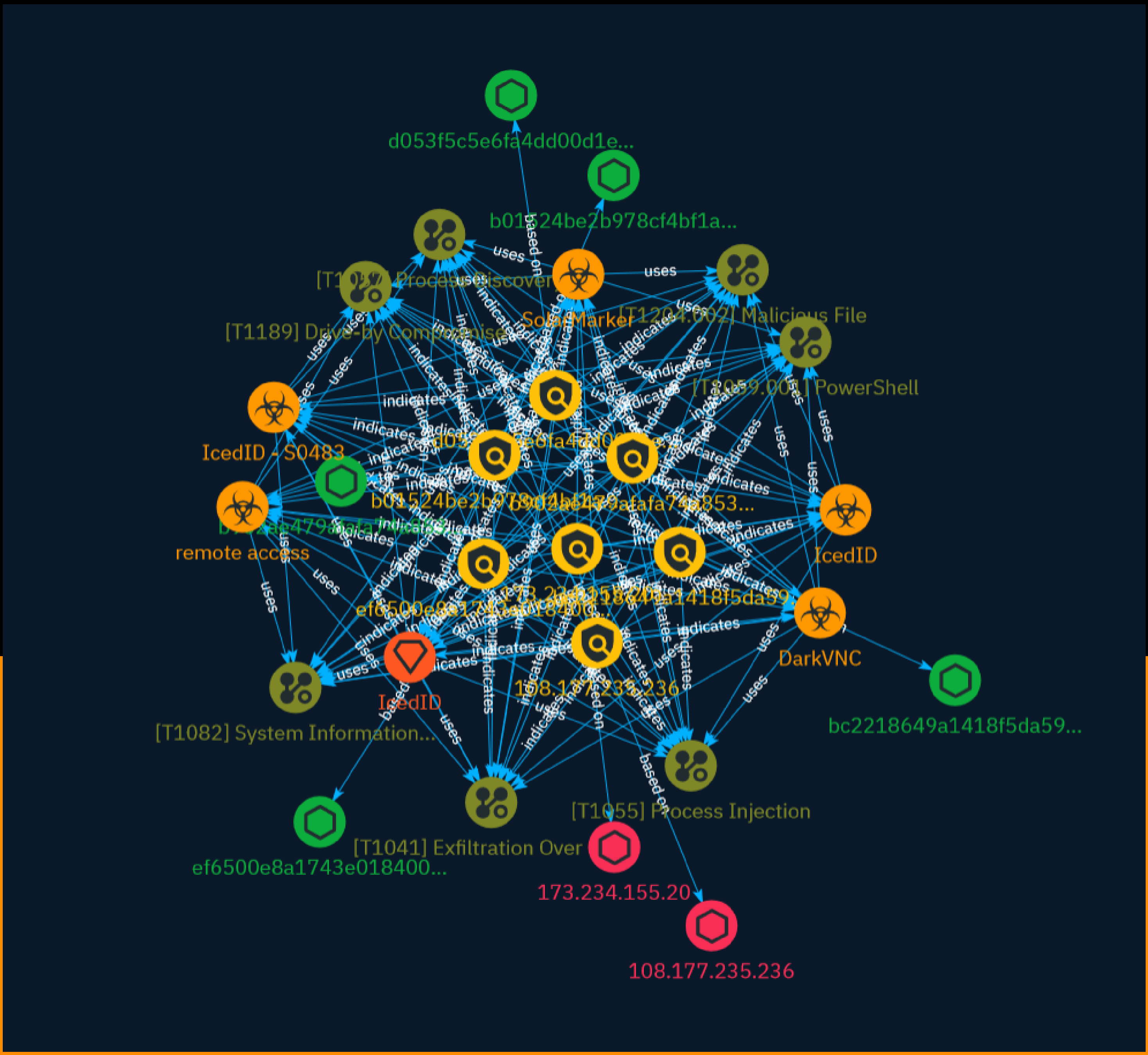

Table of contents

Overview

Entities

Observables

External References

Overview

Description

DarkVNC is a hidden utility based on VNC technology, used for stealthy remote access. It was advertised in 2016 and received updates until 2017. DarkVNC has been used by threat actors associated with IcedID and SolarMarker campaigns. This analysis focuses on a DarkVNC sample that uses 'vncdll64.dll' for exporting functions. It generates a unique ID to send to the C2 server along with system info. DarkVNC can search for and manipulate windows related to the desktop environment. It can also control the state of devices like keyboard and mouse, and block user input. The malware gathers details on the Chrome browser install and runs cmd prompts. Detection and prevention controls like EDR solutions and training programs are recommended.

Confidence

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

15 / 100

Content

N/A

Indicator

Name
173.234.155.20
Description
- **Zip Code:** N/A - **ISP:** Leaseweb New York - **ASN:** 396362 - **Organization:** Leaseweb New York - **Is Crawler:** False - **Timezone:** America/New_York - **Mobile:** False - **Host:** 173.234.155.20 - **Proxy:** True - **VPN:** True - **TOR:** False - **Active VPN:** False - **Active TOR:** False - **Recent Abuse:** True - **Bot Status:** True - **Connection Type:** Premium required. - **Abuse Velocity:** Premium required. - **Country Code:** US - **Region:** New York - **City:** New York - **Latitude:** 40.74 - **Longitude:** -73.97
Pattern Type
stix
Pattern
[ipv4-addr:value = '173.234.155.20']
Name
ef6500e8a1743e01840063544cd4e880abcfe489283c0b32920f9347a77ac4e6
Description

#Lowfi:SIGATTR:VirTool:Win32/Injector.gen!BR SHA256 of 2d84aff562319b25bbef718dde079d43

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = 'ef6500e8a1743e01840063544cd4e880abcfe489283c0b32920f9347a77ac4e6']

Name

d053f5c5e6fa4dd00d1e2dcb1e43b21e64ce99e6606c248f6fffd44cf8328c0e

Description

Backdoor:Win32/Silasilsap.STE SHA256 of f031a1ba221d29f52d16397560ae801b

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = 'd053f5c5e6fa4dd00d1e2dcb1e43b21e64ce99e6606c248f6fffd44cf8328c0e']

Name

bc2218649a1418f5da596a60ca08f030948a42a39c00818eed68e3eb922c7b94

Description

Win.Dropper.Miner-7086570-0 SHA256 of b8a9215b1d7e35698f757e20e1fc47bc

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = 'bc2218649a1418f5da596a60ca08f030948a42a39c00818eed68e3eb922c7b94']

Name

b01524be2b978cf4bf1a8c19ff0d60fc83f24d256a099efbe58fd15037326d41

Description

Win.Dropper.Miner-7086570-0 SHA256 of f15eefe467952b3946c35a578308bbda

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = 'b01524be2b978cf4bf1a8c19ff0d60fc83f24d256a099efbe58fd15037326d41']

Name

b902ae479afafa74a85305859661798bce8aa704b2bbdde5ea86cc16e7327bf8

Description

Win.Dropper.Miner-7086570-0 SHA256 of f85ae229fe7a4fde53c3b624dca754ad

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' = 'b902ae479afafa74a85305859661798bce8aa704b2bbdde5ea86cc16e7327bf8']

Name

108.177.235.236

Description

Cobalt Strike botnet C2 server (confidence level: 100%)

Pattern Type

stix

Pattern

[ipv4-addr:value = '108.177.235.236']

Malware

Name

IcedID - S0483

Name

DarkVNC

Name

SolarMarker

Name

remote access

Name

IcedID

Description

[IcedID](https://attack.mitre.org/software/S0483) is a modular banking malware designed to steal financial information that has been observed in the wild since at least 2017. [IcedID](https://attack.mitre.org/software/S0483) has been downloaded by [Emotet](https://attack.mitre.org/software/S0367) in multiple campaigns.(Citation: IBM IcedID November 2017)(Citation: Juniper IcedID June 2020)

Intrusion-Set

Name
IcedID

Attack-Pattern

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

PowerShell

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

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.

Name

Process Discovery

ID

T1057

Description

Adversaries may attempt to get information about running processes on a system. Information obtained could be used to gain an understanding of common software/applications running on systems within the network. Adversaries may use the information from [Process Discovery](https://attack.mitre.org/techniques/T1057) during automated discovery to shape follow-on behaviors, including whether or not the adversary fully infects the target and/or attempts specific actions. In Windows environments, adversaries could obtain details on running processes using the [Tasklist](https://attack.mitre.org/software/S0057) utility via [cmd](https://attack.mitre.org/software/S0106) or `Get-Process` via [PowerShell](https://attack.mitre.org/techniques/T1059/001). Information about processes can also be extracted from the output of [Native API](https://attack.mitre.org/techniques/T1106) calls such as `CreateToolhelp32Snapshot`. In Mac and Linux, this is accomplished with the `ps` command. Adversaries may also opt to enumerate processes via /proc. On network devices, [Network Device CLI](https://attack.mitre.org/techniques/T1059/008) commands such as `show processes` can be used to display current running processes.(Citation: US-CERT-TA18-106A)(Citation: show_processes_cisco_cmd)

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

System Information Discovery

ID

T1082

Description

An adversary may attempt to get detailed information about the operating system and hardware, including version, patches, hotfixes, service packs, and architecture. Adversaries may use the information from [System Information Discovery](https://attack.mitre.org/techniques/T1082) during automated discovery to shape follow-on behaviors, including whether or not the adversary fully infects the target and/or attempts specific actions. Tools such as [Systeminfo](https://attack.mitre.org/software/S0096) can be used to gather detailed system information. If running with privileged access, a breakdown of system data can be gathered through the `systemsetup` configuration tool on macOS. As an example, adversaries with user-level access can execute the `df -aH` command to obtain currently mounted disks and associated freely available space. Adversaries may also leverage a [Network Device CLI](https://attack.mitre.org/techniques/T1059/008) on network devices to gather detailed system information (e.g. `show version`).(Citation: US-CERT-TA18-106A) [System Information Discovery](https://attack.mitre.org/techniques/T1082) combined with information gathered from other forms of discovery and reconnaissance can drive payload development and concealment.(Citation: OSX.FairyTale)(Citation: 20 macOS Common Tools and Techniques) Infrastructure as a Service (IaaS) cloud providers such as AWS, GCP, and Azure allow access to instance and virtual machine information via APIs. Successful authenticated API calls can return data such as the operating system platform and status of a particular instance or the model view of a virtual machine.(Citation: Amazon Describe Instance)(Citation: Google Instances Resource)(Citation: Microsoft Virutal Machine API)

Name

Malicious File

ID

T1204.002

Description

An adversary may rely upon a user opening a malicious file in order to gain execution. Users may be subjected to social engineering to get them to open a file that will lead to code execution. This user action will typically be observed as follow-on behavior from [Spearphishing Attachment](https://attack.mitre.org/techniques/T1566/001). Adversaries may use several types of files that require a user to execute them, including .doc, .pdf, .xls, .rtf, .scr, .exe, .lnk, .pif, and .cpl. Adversaries may employ various forms of [Masquerading](https://attack.mitre.org/techniques/T1036) and [Obfuscated Files or Information](https://attack.mitre.org/techniques/T1027) to increase the likelihood that a user will open and successfully execute a malicious file. These methods may include using a familiar naming convention and/or password protecting the file and supplying instructions to a user on how to open it.(Citation: Password Protected Word Docs) While [Malicious File](https://attack.mitre.org/techniques/T1204/002) 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).

IPv4-Addr

Value
173.234.155.20
108.177.235.236

StixFile

Value
ef6500e8a1743e01840063544cd4e880abcfe489283c0b32920f9347a77ac4e6
d053f5c5e6fa4dd00d1e2dcb1e43b21e64ce99e6606c248f6fffd44cf8328c0e
bc2218649a1418f5da596a60ca08f030948a42a39c00818eed68e3eb922c7b94
b902ae479afafa74a85305859661798bce8aa704b2bbdde5ea86cc16e7327bf8
b01524be2b978cf4bf1a8c19ff0d60fc83f24d256a099efbe58fd15037326d41

External References

- https://www.esentire.com/blog/technical-analysis-of-darkvnc
- https://otx.alienvault.com/pulse/65d348c6927ea8aae1bee945