The Case of LummaC2 v4.0

By eSentire Threat Response Unit (TRU)

Archived: 2026-04-10 02:23:05 UTC

Adversaries don’t work 9-5 and neither do we. At eSentire, our 24/7 SOCs are staffed with Elite Threat Hunters and Cyber Analysts who hunt, investigate, contain and respond to threats within minutes.

We have discovered some of the most dangerous threats and nation state attacks in our space – including the Kaseya MSP breach and the more_eggs malware.

Our Security Operations Centers are supported with Threat Intelligence, Tactical Threat Response and Advanced Threat Analytics driven by our Threat Response Unit – the TRU team.

In TRU Positives, eSentire’s Threat Response Unit (TRU) provides a summary of a recent threat investigation. We outline how we responded to the confirmed threat and what recommendations we have going forward.

Here’s the latest from our TRU Team…

What did we find?

Since the beginning of August 2023, the eSentire Threat Response Unit (TRU) has observed 5 cases of Lumma Stealer infections across manufacturing, retail, and business industries.

The infostealer was delivered via drive-by downloads disguised as fake installers such as Chrome and Edge browser installers. and some of them were distributed via PrivateLoader (a loader known to drop malware such as Redline, DCRat, and RaccoonStealer in the past).

eSentire TRU recently published a TRU Positive on LummaC2 delivering Amadey and PrivateLoader; you can access the article here.

Lumma Stealer, also known as LummaC2, first appeared on Russian-speaking forums at the end of 2022. The stealer is written in C++, and the build size is approximately 150-396 KB. The pricing for the stealer is divided into three categories:

- **Experienced - $250/month**: Includes capabilities such as downloading logs in bulk and the ability to filter the logs by specific parameters.
- **Professional - $500/month**: Includes capabilities such as creating an unlimited number of rules for the stealer, sharing the statistics with other people, non-residential loader, etc.
- **Corporate - $1000/month**: The developer claims that with the corporate plan, the user gets a random build by the morpher, which means each generated build is different from the other, implemented Heaven’s Gate (a technique used by malware to transition from 32-bit code execution to 64-bit code execution on x64 Windows systems, it’s used to bypass dynamic analysis by sandboxes and debuggers).

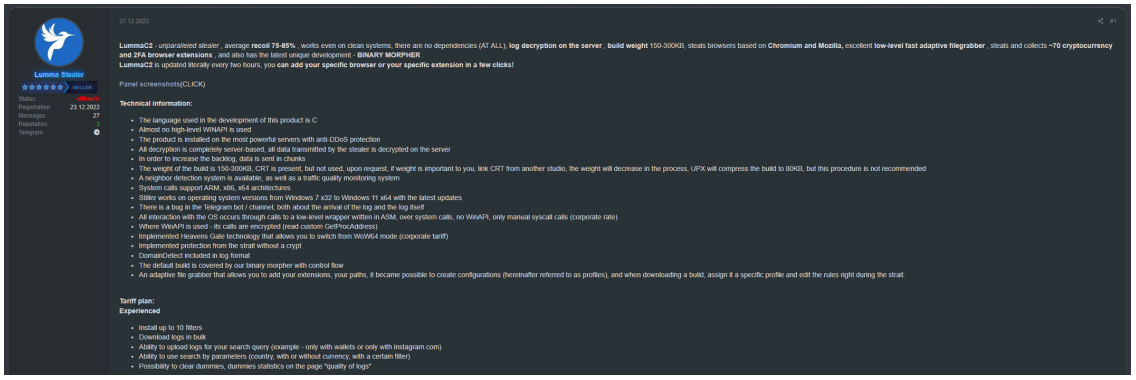


Figure 1: Stealer advertisement on Russian speaking forums

The stealer features a non-resident loader that is capable of delivering additional payloads via EXE, DLL, and PowerShell:

Figure 2: Non-resident loader

Some other notable features of the stealer include:

- **Control Flow Flattening –** code obfuscation technique used to make the program's control flow more complex and harder to analyze.
- "**Spreading without the crypt protection" –** if the generated stealer build is not crypted, the pop-up message will show up for the user upon running the build, as shown in Figure 3.
- **Flexible file grabber –** the ability to directly change the rules in the profile to grab specific data.

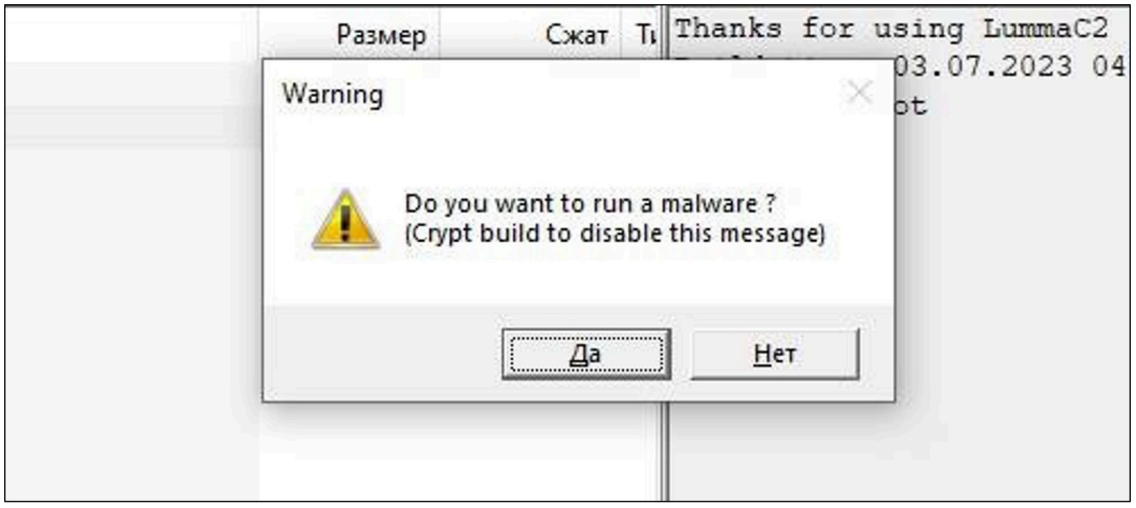


Figure 3: Pop-up message if the build is not crypted

Technical Analysis

From the previous versions of the stealer (the sample we observed in June, MD5: a4e04604c178fe4fd19996b4d48b8885), as an obfuscation method, the stealer appends strings such as "576xe" to the data.

```
.rdata:0044D59B aNdowsServer201 db 'ndows Server 2012 R2',0
.rdata:0044D5B0 aWindows10      db 'Windows 10',0
.rdata:0044D5BB aWindowsServer2_3 db 'Windows Server 2016',0
.rdata:0044D5CF                 align 10h
.rdata:0044D5D0 aUser32Dll_0:
.rdata:0044D5D0                 text "UTF-16LE", 'user32.dll',0
.rdata:0044D5E6 aLum576xedmac2B db 'Lum576xedmaC2, Build 20233101',0Ah,0
.rdata:0044D605                 db  4Ch ; L
.rdata:0044D606                 db  49h ; I
.rdata:0044D607                 db  44h ; D
.rdata:0044D608                 db  28h ; (
.rdata:0044D609                 db  4Ch ; L
.rdata:0044D60A                 db  75h ; u
.rdata:0044D60B                 db  35h ; 5
.rdata:0044D60C                 db  37h ; 7
.rdata:0044D60D                 db  36h ; 6
.rdata:0044D60E                 db  78h ; x
.rdata:0044D60F                 db  65h ; e
.rdata:0044D610                 db  64h ; d
.rdata:0044D611                 db  6Dh ; m
.rdata:0044D612                 db  6Dh ; m
.rdata:0044D613                 db  61h ; a
.rdata:0044D614                 db  20h
.rdata:0044D615                 db  49h ; I
.rdata:0044D616                 db  44h ; D
.rdata:0044D617 dword_44D617    dd 203A29h              ; DATA XREF: sub_40BE8A:loc_40D2
.rdata:0044D61B aSDDD           db '%s (%d.%d.%d)',0
.rdata:0044D629 aHw576xedid     db '- HW576xedID: ',0
.rdata:0044D638 aScreenResoluto db '- Screen Resoluton: ',0
.rdata:0044D64D aCp576xeduName  db '- CP576xedU Name: ',0
.rdata:0044D660 aPhys576xedical db '- Phys576xedical Ins576xedtalled Memor576xedy: ',0
```

Figure 4: String obfuscation for previous versions of LummaC2

But with the latest versions (version 4.0), the stealer obfuscates the strings with an XOR algorithm. It converts the hexadecimal characters in the string into the decimal value. The first four bytes of the byte array of the string are used as the XOR key. Each subsequent byte in the array is XORed with the mentioned four bytes in a cyclic manner. The decryption is shown algorithm is shown in the screenshot below.

```
_WORD *__cdecl mw_str_decrypt(int enc_str)
{
  size_t str_len; // edi
  size_t v3; // esi
  _DWORD *v4; // eax
  _DWORD *v5; // ebp
  int v6; // esi
  char v7; // bl
  _WORD *v8; // edi
  int v9; // ecx
  int str_dec[5]; // [esp+8h] [ebp-14h]

  str_len = -1;
  while ( *(_WORD *)(enc_str + 2 * str_len++ + 2) != 0 )// get the length by iterating over the string two chars at a time
    ;
  v3 = str_len >> 1;
  v4 = malloc(str_len >> 1);
  v5 = v4;
  if ( str_len )
  {
    v6 = 0;
    do
    {
      v7 = 16 * mw_convert_hex_to_dec(*(_BYTE *)(enc_str + 4 * v6));
      *((_BYTE *)v5 + v6) = v7 | mw_convert_hex_to_dec(*(_BYTE *)(enc_str + 4 * v6 + 2));
      ++v6;
    }
    while ( ((str_len - 1) >> 1) + 1 != v6 );
    v4 = (_DWORD *)*v5;
    v3 = str_len >> 1;
  }
  str_dec[0] = (int)v4;
  v8 = calloc(1u, str_len);
  if ( v3 != 4 )
  {
    v9 = 0;
    do
    {
      v8[v9] = (char)(*((_BYTE *)v5 + v9 + 4) ^ *((_BYTE *)str_dec + (v9 & 3)));// 1. first 4 bytes is the xor [:4]
                                                // 2. goes through the bytes, starting from the 5th byte.
      ++v9;
    }
    while ( v3 - 4 != v9 );
  }
  v8[v3 - 4] = 0;
```

Figure 5: LummaC2 decryption algorithm

```python
def hex_char_to_byte(char):
    ascii_val = ord(char)

    if '0' <= char <= '9':
        return ascii_val - ord('0')
    elif 'a' <= char <= 'f':
        return ascii_val - ord('a') + 10
    elif 'A' <= char <= 'F':
        return ascii_val - ord('A') + 10
    else:
        return 0

def deobfuscate_str(input_str):
    byte_array = []
    for i in range(0, len(input_str), 2):
        first_byte = hex_char_to_byte(input_str[i]) * 16
        second_byte = hex_char_to_byte(input_str[i+1])
        byte_array.append(first_byte | second_byte)

    # XOR decryption
    key = byte_array[:4]
    decrypted_bytes = [byte_array[i] ^ key[i % 4] for i in range(4, len(byte_array))]

    decrypted_str = ''.join([chr(b) for b in decrypted_bytes])

    return decrypted_str

enc_string = "aab58e5185f0f625cfdbfd38c5dbfd7e" # input the encrypted string
deobfuscated_str = deobfuscate_str(enc_string)
print(deobfuscated_str)
```

Figure 6: String decrypter implementation in Python

The stealer uses MurmurHash2 hashing algorithm to resolve the APIs, in our example the seed is 0x20. The API hashing function is obfuscated with control flow flattening as the rest of the code (Figure 8).

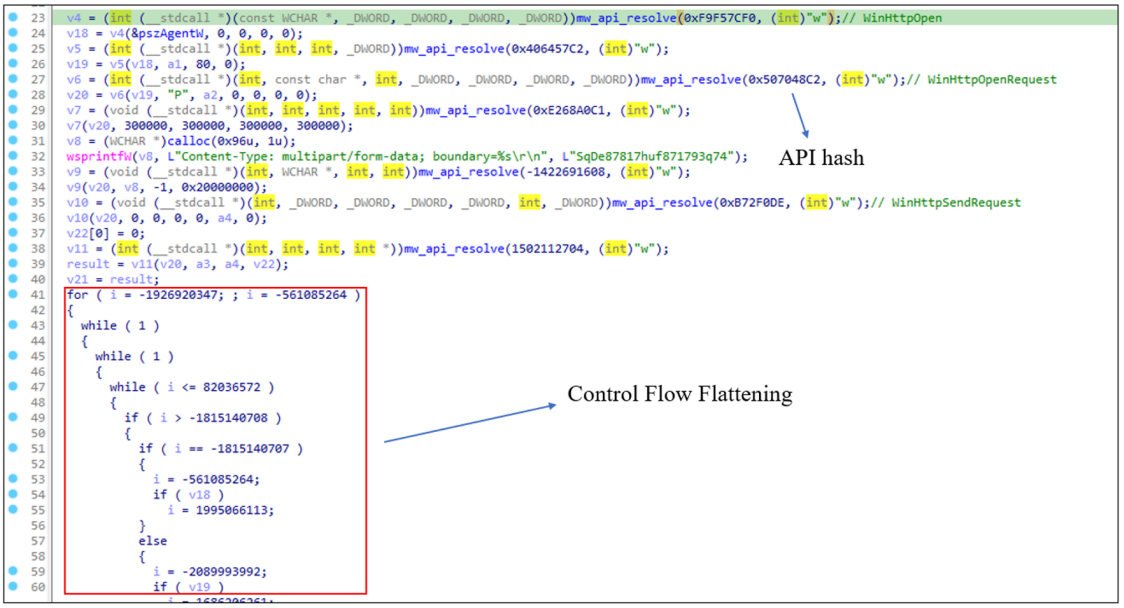
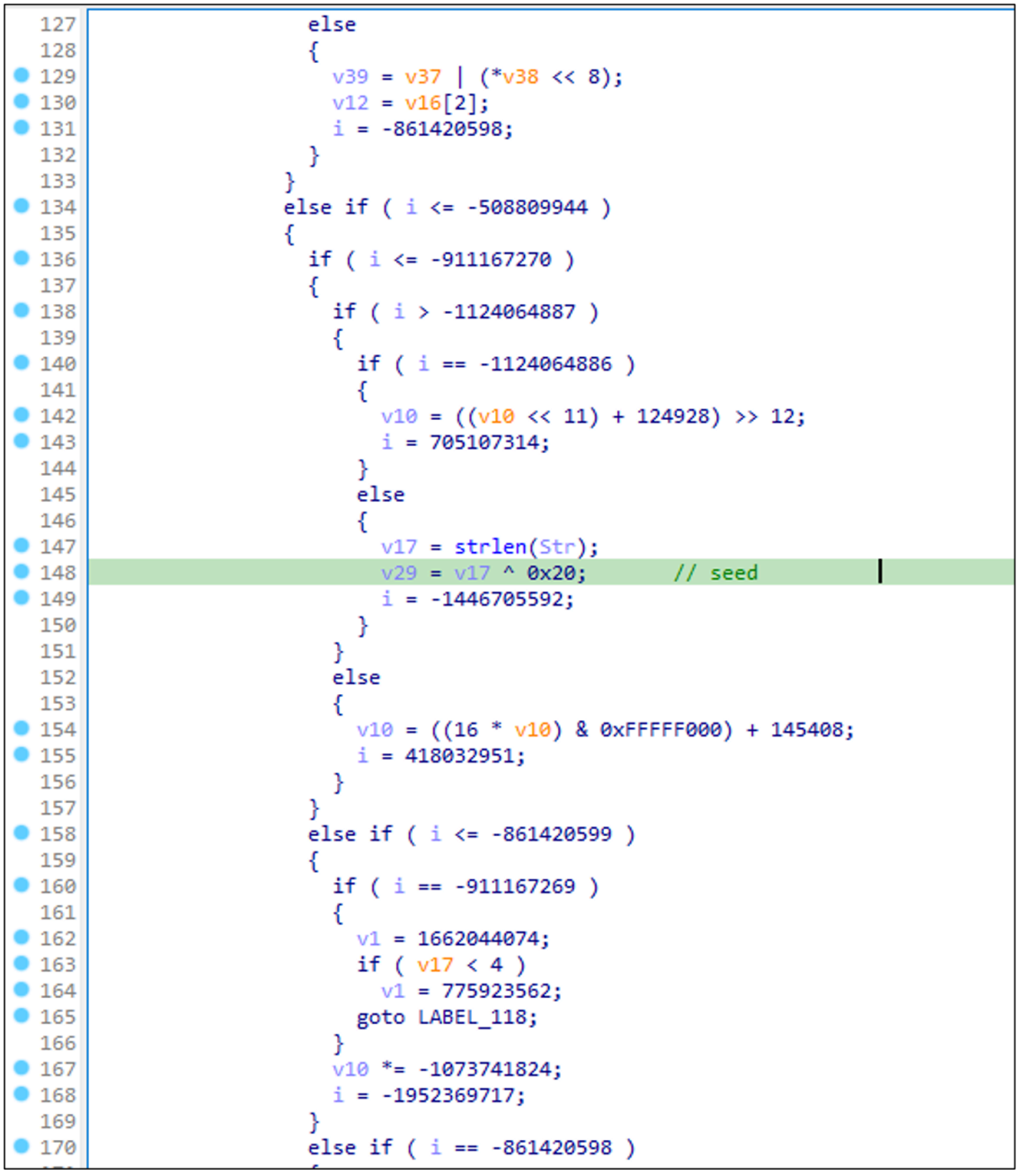


Figure 7: Snippet of code with control flow flattening



```
127                   else
128                   {
129                     v39 = v37 | (*v38 << 8);
130                     v12 = v16[2];
131                     i = -861420598;
132                   }
133                 }
134                 else if ( i <= -508809944 )
135                 {
136                   if ( i <= -911167270 )
137                   {
138                     if ( i > -1124064887 )
139                     {
140                       if ( i == -1124064886 )
141                       {
142                         v10 = ((v10 << 11) + 124928) >> 12;
143                         i = 705107314;
144                       }
145                       else
146                       {
147                         v17 = strlen(Str);
148                         v29 = v17 ^ 0x20;       // seed
149                         i = -1446705592;
150                       }
151                     }
152                     else
153                     {
154                       v10 = ((16 * v10) & 0xFFFFF000) + 145408;
155                       i = 418032951;
156                     }
157                   }
158                   else if ( i <= -861420599 )
159                   {
160                     if ( i == -911167269 )
161                     {
162                       v1 = 1662044074;
163                       if ( v17 < 4 )
164                         v1 = 775923562;
165                       goto LABEL_118;
166                     }
167                     v10 *= -1073741824;
168                     i = -1952369717;
169                   }
170                   else if ( i == -861420598 )
```

Figure 8: API hashing function

We wrote the script to produce the hash values based on the seed for each API call within the DLL library found in the stealer.

C2 Communication

Upon the initial infection, the stealer retrieves the configuration from the C2 (/c2conf). @Jane_0sint provided insights on decrypting the configuration using the first 32-byte of the base64-encoded string as the key. That means that the crypto wallet extensions, applications, and list of browsers are not embedded in the stealer build anymore compared to previous versions of the stealer.

Here is the example of the decrypted configuration retrieved from C2 (JSON format):

```
{
   "v": 1,
   "c": [
       {
           "t": 0,
           "p": "%userprofile%",
           "m": "*.txt",
           "z": "Important Files/Profile",
           "d": 1
       },
       {
           "t": 0,
           "p": "%userprofile%",
           "m": "*key*",
           "z": "Important Files/Profile",
           "d": 1
       },
       {
           "t": 0,
           "p": "%userprofile%",
           "m": "*bitcoin*",
           "z": "Important Files/Profile",
           "d": 3
       },
       {
           "t": 0,
           "p": "%userprofile%",
           "m": "*binance*",
           "z": "Important Files/Profile",
           "d": 3
       },
       {
           "t": 0,
           "p": "%userprofile%",
           "m": "*exodus*",
           "z": "Important Files/Profile",
           "d": 3
       },
       {
           "t": 0,
           "p": "%userprofile%",
           "m": "*coinbase*",
           "z": "Important Files/Profile",
           "d": 3
       },
       {
```

```
        "t": 0,
        "p": "%userprofile%",
        "m": "*wallet*",
        "z": "Important Files/Profile",
        "d": 3
    },
    {
        "t": 0,
        "p": "%userprofile%",
        "m": "*seed*",
        "z": "Important Files/Profile",
        "d": 3
    },
    {
        "t": 0,
        "p": "%userprofile%",
        "m": "*pass*",
        "z": "Important Files/Profile",
        "d": 3
    },
    {
        "t": 0,
        "p": "%userprofile%",
        "m": "*ledger*",
        "z": "Important Files/Profile",
        "d": 3
    },
    {
        "t": 0,
        "p": "%userprofile%",
        "m": "*trezor*",
        "z": "Important Files/Profile",
        "d": 3
    },
    {
        "t": 0,
        "p": "%userprofile%",
        "m": "*metamask*",
        "z": "Important Files/Profile",
        "d": 3
    },
    {
        "t": 0,
        "p": "%userprofile%",
        "m": "*crypto*",
        "z": "Important Files/Profile",
        "d": 1
```

```
        },
        {
            "t": 0,
            "p": "%appdata%\\Binance",
            "m": "app-store.json",
            "z": "Wallets/Binance",
            "d": 1
        },
        {
            "t": 0,
            "p": "%appdata%\\Binance",
            "m": ".finger-print.fp",
            "z": "Wallets/Binance",
            "d": 1
        },
        {
            "t": 0,
            "p": "%appdata%\\Binance",
            "m": "simple-storage.json",
            "z": "Wallets/Binance",
            "d": 1
        },
        {
            "t": 0,
            "p": "%appdata%\\Electrum\\wallets",
            "m": "*",
            "z": "Wallets/Electrum",
            "d": 1
        },
        {
            "t": 0,
            "p": "%appdata%\\Ethereum",
            "m": "keystore",
            "z": "Wallets/Ethereum",
            "d": 1
        },
        {
            "t": 0,
            "p": "%appdata%\\Exodus\\exodus.wallet",
            "m": "*",
            "z": "Wallets/Exodus",
            "d": 2
        },
        {
            "t": 0,
            "p": "%appdata%\\Ledger Live",
            "m": "*",
```

```
            "z": "Wallets/Ledger Live",
            "d": 2
        },
        {
            "t": 0,
            "p": "%appdata%\\atomic\\Local Storage\\leveldb",
            "m": "*",
            "z": "Wallets/Atomic",
            "d": 2
        },
        {
            "t": 0,
            "p": "%localappdata%\\Coinomi\\Coinomi\\wallets",
            "m": "*",
            "z": "Wallets/Coinomi",
            "d": 2
        },
        {
            "t": 0,
            "p": "%appdata%\\Authy Desktop\\Local Storage\\leveldb",
            "m": "*",
            "z": "Wallets/Authy Desktop",
            "d": 2
        },
        {
            "t": 0,
            "p": "%appdata%\\Bitcoin\\wallets",
            "m": "*",
            "z": "Wallets/Bitcoin core",
            "d": 2
        },
        {
            "t": 0,
            "p": "%appdata%\\com.liberty.jaxx\\IndexedDB",
            "m": "*.leveldb",
            "z": "Wallets/JAXX New Version",
            "d": 2
        },
        {
            "t": 0,
            "p": "%appdata%\\Electrum\\wallets",
            "m": "*",
            "z": "Wallets/Electrum",
            "d": 2
        },
        {
            "t": 0,
```

```
            "p": "%appdata%\\AnyDesk",
            "m": "*.conf",
            "z": "Applications/AnyDesk",
            "d": 2
        },
        {
            "t": 0,
            "p": "%appdata%\\FileZilla",
            "m": "recentservers.xml",
            "z": "Applications/FileZilla",
            "d": 2
        },
        {
            "t": 0,
            "p": "%appdata%\\FileZilla",
            "m": "sitemanager.xml",
            "z": "Applications/FileZilla",
            "d": 2
        },
        {
            "t": 0,
            "p": "%userprofile%",
            "m": "*.kbdx",
            "z": "Applications/KeePass",
            "d": 2
        },
        {
            "t": 0,
            "p": "%programfiles%\\Steam",
            "m": "ssfn*",
            "z": "Applications/Steam",
            "d": 2
        },
        {
            "t": 0,
            "p": "%programfiles%\\Steam\\config",
            "m": "*",
            "z": "Applications/Steam/config",
            "d": 2
        },
        {
            "t": 0,
            "p": "%appdata%\\Telegram Desktop",
            "m": "*s",
            "z": "Applications/Telegram",
            "d": 2
        },
```

```
{
    "t": 1,
    "e": [
        {
            "en": "ejbalbakoplchlghecdalmeeeajnimhm",
            "ez": "MetaMask"
        },
        {
            "en": "nkbihfbeogaeaoehlefnkodbefgpgknn",
            "ez": "MetaMask"
        },
        {
            "en": "egjidjbpglichdcondbcbdnbeeppgdph",
            "ez": "Trust Wallet"
        },
        {
            "en": "ibnejdfjmmkpcnlpebklmnkoeoihofec",
            "ez": "TronLink"
        },
        {
            "en": "fnjhmkhhmkbjkkabndcnnogagogbneec",
            "ez": "Ronin Wallet"
        },
        {
            "en": "fhbohimaelbohpjbbldcngcnapndodjp",
            "ez": "Binance Chain Wallet"
        },
        {
            "en": "ffnbelfdoeiohenkjibnmadjiehjhajb",
            "ez": "Yoroi"
        },
        {
            "en": "jbdaocneiiinmjbjlgalhcelgbejmnid",
            "ez": "Nifty"
        },
        {
            "en": "afbcbjpbpfadlkmhmclhkeeodmamcflc",
            "ez": "Math"
        },
        {
            "en": "hnfanknocfeofbddgcijnmhnfnkdnaad",
            "ez": "Coinbase"
        },
        {
            "en": "hpglfhgfnhbgpjdenjgmdgoeiappafln",
            "ez": "Guarda"
        },
```

```
{
    "en": "blnieiiffboillknjnepogjhkgnoapac",
    "ez": "EQUA"
},
{
    "en": "cjelfplplebdjjenllpjcblmjkfcffne",
    "ez": "Jaxx Liberty"
},
{
    "en": "fihkakfobkmkjojpchpfgcmhfjnmnfpi",
    "ez": "BitApp"
},
{
    "en": "kncchdigobghenbbaddojjnnaogfppfj",
    "ez": "iWlt"
},
{
    "en": "kkpllkodjeloidieedojogacfhpaihoh",
    "ez": "EnKrypt"
},
{
    "en": "amkmjjmmflddogmhpjloimipbofnfjih",
    "ez": "Wombat"
},
{
    "en": "nlbmnnijcnlegkjjpcfjclmcfggfefdm",
    "ez": "MEW CX"
},
{
    "en": "nanjmdknhkinifnkgdcggcfnhdaammmj",
    "ez": "Guild"
},
{
    "en": "nkddgncdjgjfcddamfgcmfnlhccnimig",
    "ez": "Saturn"
},
{
    "en": "cphhlgmgameodnhkjdmkpanlelnlohao",
    "ez": "NeoLine"
},
{
    "en": "nhnkbkgjikgcigadomkphalanndcapjk",
    "ez": "Clover"
},
{
    "en": "kpfopkelmapcoipemfendmdcghnegimn",
    "ez": "Liquality"
```

```
    },
    {
        "en": "aiifbnbfobpmeekipheeijimdpnlpgpp",
        "ez": "Terra Station"
    },
    {
        "en": "dmkamcknogkgcdfhhbddcghachkejeap",
        "ez": "Keplr"
    },
    {
        "en": "fhmfendgdocmcbmfikdcogofphimnkno",
        "ez": "Sollet"
    },
    {
        "en": "cnmamaachppnkjgnildpdmkaakejnhae",
        "ez": "Auro"
    },
    {
        "en": "jojhfeoedkpkglbfimdfabpdfjaoolaf",
        "ez": "Polymesh"
    },
    {
        "en": "flpiciilemghbmfalicajoolhkkenfe",
        "ez": "ICONex"
    },
    {
        "en": "nknhiehlklippafakaeklbeglecifhad",
        "ez": "Nabox"
    },
    {
        "en": "hcflpincpppdclinealmandijcmnkbgn",
        "ez": "KHC"
    },
    {
        "en": "ookjlbkiijinhpmnjffcofjonbfbgaoc",
        "ez": "Temple"
    },
    {
        "en": "mnfifefkajgofkcjkemidiaecocnkjeh",
        "ez": "TezBox"
    },
    {
        "en": "lodccjjbdhfakaekdiahmedfbieldgik",
        "ez": "DAppPlay"
    },
    {
        "en": "ijmpgkjfkbfhoebgogflfebnmejmfbm",
```

```
        "ez": "BitClip"
    },
    {
        "en": "lkcjlnjfpbikmcmbachjpdbijejflpcm",
        "ez": "Steem Keychain"
    },
    {
        "en": "onofpnbbkehpmmoabgpcpmigafmmnjh",
        "ez": "Nash Extension"
    },
    {
        "en": "bcopgchhojmggmffilplmbdicgaihlkp",
        "ez": "Hycon Lite Client"
    },
    {
        "en": "klnaejjgbibmhlephnhpmaofohgkpgkd",
        "ez": "ZilPay"
    },
    {
        "en": "aeachknmefphepccionboohckonoeemg",
        "ez": "Coin98"
    },
    {
        "en": "bhghoamapcdpbohphigoooaddinpkbai",
        "ez": "Authenticator"
    },
    {
        "en": "dkdedlpgdmmkkfjabffeganieamfklkm",
        "ez": "Cyano"
    },
    {
        "en": "nlgbhdfgdhgbiamfdfmbikcdghidoadd",
        "ez": "Byone"
    },
    {
        "en": "infeboajgfhgbjpjbeppbkgnabfdkdaf",
        "ez": "OneKey"
    },
    {
        "en": "cihmoadaighcejopammfbmddcmdekcje",
        "ez": "Leaf"
    },
    {
        "en": "gaedmjdfmmahhbjefcbgaolhhanlaolb",
        "ez": "Authy"
    },
    {
```

```
                "en": "oeljdldpnmdbchonielidgobddfffla",
                "ez": "EOS Authenticator"
            },
            {
                "en": "ilgcnhelpchnceeipipijaljkblbcob",
                "ez": "GAuth Authenticator"
            },
            {
                "en": "imloifkgjagghnncjkhggdhalmcnfklk",
                "ez": "Trezor Password Manager"
            },
            {
                "en": "bfnaelmomeimhlpmgjnjophhpkkoljpa",
                "ez": "Phantom"
            },
            {
                "en": "ppbibelpcjmhbdihakflkdcoccbgbkpo",
                "ez": "UniSat"
            }
        ],
        "n": [
            {
                "p": "%localappdata%\\Google\\Chrome\\User Data",
                "z": "Chrome"
            },
            {
                "p": "%localappdata%\\Chromium\\User Data",
                "z": "Chromium"
            },
            {
                "p": "%localappdata%\\Microsoft\\Edge\\User Data",
                "z": "Edge"
            },
            {
                "p": "%localappdata%\\Kometa\\User Data",
                "z": "Kometa"
            },
            {
                "p": "%appdata%\\Opera Software\\Opera Stable",
                "z": "Opera Stable"
            },
            {
                "p": "%appdata%\\Opera Software\\Opera GX Stable",
                "z": "Opera GX Stable"
            },
            {
                "p": "%appdata%\\Opera Software\\Opera Neon\\User Data",
```

```
                    "z": "Opera Neon"
                },
                {
                    "p": "%localappdata%\\BraveSoftware\\Brave-Browser\\User Data",
                    "z": "Brave Software"
                },
                {
                    "p": "%localappdata%\\Comodo\\Dragon\\User Data",
                    "z": "Comodo"
                },
                {
                    "p": "%localappdata%\\CocCoc\\Browser\\User Data",
                    "z": "CocCoc"
                }
            ]
        },
        {
            "t": 2,
            "p": "%appdata%\\Mozilla\\Firefox\\Profiles",
            "z": "Mozilla Firefox"
        }
    ]
}
```

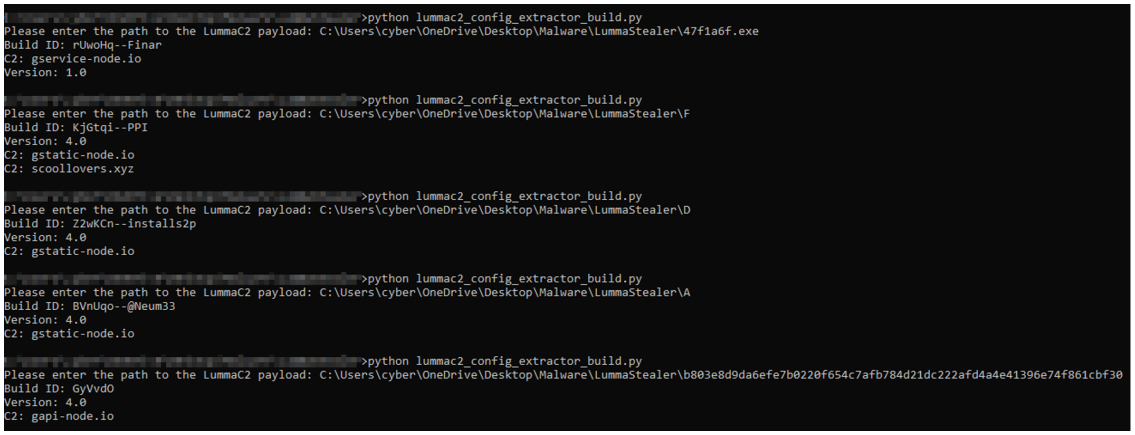


Figure 9: Output of the configuration extractor script

Example of additional data obtained from C2: *lid=KjGtqi--VIKKY&j=e799236f7a828928688bbd10d343328e&ver=4.0*, where the *lid* is the LummaC2 ID, the value “VIKKY” is likely the build ID assigned and the stealer version (ver=). The logs are exfiltrated over the ZIP archive, as shown in the screenshot below (/c2sock).

Figure 10: Exfiltrated logs

```
data:00456892 aPc              db '- PC: ',0
data:00456899 aUser            db '- User: ',0
data:004568A2 aDomain          db '- Domain: ',0
data:004568AD aWorkgroup       db '- Workgroup: ',0
data:004568BB aComputernamedn  db '- ComputerNameDnsHostname: ',0
data:004568BB                                  ; DATA XREF: sub_428E94+211B↑o
data:004568D7 aComputernamene  db '- ComputerNameNetBIOS: ',0
data:004568D7                                  ; DATA XREF: sub_428E94+3174↑o
data:004568EF aOsVersion       db '- OS Version: ',0
data:004568FE aHwid            db '- HWID: ',0
data:00456907 aScreenResoluto  db '- Screen Resoluton: ',0
data:00456907                                  ; DATA XREF: sub_428E94+275C↑o
data:0045691C aLanguage        db '- Language: ',0
data:00456929 aCpuName         db '- CPU Name: ',0
data:00456936 aPhysicalInstal  db '- Physical Installed Memory: ',0
```

Figure 11: Log formatting (System.txt)

```
1100                i = 1529246359;
1101              }
1102            }
1103            else if ( i <= -2065731177 )
1104            {
1105              if ( i == -2137542257 )
1106              {
1107                i = -688594717;
1108              }
1109              else if ( i == -2133076369 )
1110              {
1111                memmove(v128, L"System.txt", 0x16u);
1112                v52 = strlen(v102);
1113                sub_41988D(a1, (LPCWCH)v128, (int)v102, v52);
1114                i = -125749283;
1115              }
1116              else
1117              {
1118                i = 562358012;
1119              }
1120            }
1121            else if ( i > -2053089015 )
1122            {
1123              if ( i == -2053089014 )
1124              {
1125                i = 372301985;
1126              }
1127              else
1128              {
1129                EnumDisplayDevicesA(0, 0, v106, 0);
1130                i = 1422926142;
1131              }
1132            }
1133            else
1134            {
1135              *v104 = 256;
1136              GetComputerNameExA(ComputerNameDnsDomain, v112, v104);
1137              i = 370367479;
1138            }
1139          }
1140          else if ( i <= -1806034577 )
1141          {
1142            if ( i <= -1876740375 )
1143            {
1144              if ( i != -1903342481 )
```

Figure 12: Function responsible for collecting system information and parsing it to the System.txt file

Имя	Размер	Сжат	Тип	Изменён	CRC32
..			Папка с файлами		
Wallets	133 759	132 909	Папка с файлами	05.07.2023 15:43	
Thunderbird	689	471	Папка с файлами	05.07.2023 15:43	
Opera GX Stable	5 154	2 601	Папка с файлами	05.07.2023 15:43	
Important Files	1 273 090	489 841	Папка с файлами	05.07.2023 15:43	
Edge	20 074	9 767	Папка с файлами	05.07.2023 15:43	
Cookies	155 945	78 307	Папка с файлами	05.07.2023 15:43	
Chrome	93 546	37 297	Папка с файлами	05.07.2023 15:43	
Brave Software	562 418	138 898	Папка с файлами	05.07.2023 15:43	
System.txt	382	323	Текстовый докуме...	20.05.2023 2:39	3791585F
Software.txt	1 024	588	Текстовый докуме...	20.05.2023 2:39	B3D708E4
Screen.png	14 745 654	847	Файл "PNG"	20.05.2023 2:39	74A53E14
DomainDetect.txt	241	155	Текстовый докуме...	20.05.2023 2:39	DB61C11E
Brute.txt	935	642	Текстовый докуме...	20.05.2023 2:39	AD53BFE9
All Passwords.txt	38 793	9 737	Текстовый докуме...	20.05.2023 2:39	A562BD30

Figure 13: The contents of the ZIP archive (Source: Dark2Web)

What did we do?

- Our team of 24/7 SOC Cyber Analysts isolated affected host to contain the infection.
- We also provided remediation recommendations and support to the customer.

What can you learn from this TRU positive?

- LummaC2, which is written in C++, is delivered via drive-by downloads.
- The new version of the stealer (4.0) uses the XOR algorithm for string encryption.
- It applies control flow flattening to complicate the analysis as well as API hashing using the MurmurHash2 hashing algorithm.
- LummaC2 Stealer is sold in three different tiers: Experienced, Professional, and Corporate. The last one comes with a Heaven's Gate technique.

Recommendations from our Threat Response Unit (TRU):

Protecting against information stealers requires a multi-layered defense approach to defend endpoints from malware and detect or block unauthorized login activity against applications and remote access services. Therefore, we recommend:

- Protecting endpoints against malware.
    - Ensure antivirus signatures are up to date.
    - Use a Next-Gen AV (NGAV) or Endpoint Detection and Response (EDR) product to detect and contain threats.
    - If an information stealing malware is identified, reset the user's credentials and terminate logon sessions immediately.
- Encouraging good cybersecurity hygiene among your users by using Phishing and Security Awareness Training (PSAT) when downloading software from the Internet.
- Restricting access to enterprise applications from personal devices outside the scope of security monitoring.
- Ensuring adequate logging is in place for remote access services such as VPNs and using modern authentication methods, which support MFA and conditional access.
- Prevent web browsers from automatically saving and storing passwords.
- It is recommended to use password managers instead.

Indicators of Compromise

Name	Indicators
C2	gservice-node[.]io

C2	gstatic-node[.]io
C2	balancelag[.]xyz
C2	doorblu[.]xyz
Payload	c9094685ae4851fd5a5b886b73c7b07efd9b47ea0bdae3f823d035cf1b3b9e48
Payload	3bf4b365d61c1e9807d20e71375627450b8fea1635cb6ddb85f2956e8f6b3ec3
Payload	a9c7c4df7e39b8bc7b03b7d4053066d26154bc76c3e261fff84f2339d44a474e
Payload	b803e8d9da6efe7b0220f654c7afb784d21dc222afd4a4e41396e74f861cbf30

Yara

```
rule LummaC2 {

   meta:
       author = "RussianPanda"
       description = "LummaC2 Detection"

   strings:
       $p1="lid=%s&j=%s&ver"
       $p2= {89 ca 83 e2 03 8a 54 14 08 32 54 0d 04}

   condition:
       all of them and filesize <= 500KB
}
```

Reference

- https://blog.sekoia.io/privateloader-the-loader-of-the-prevalent-ruzki-ppi-service/
- https://twitter.com/Jane_0sint/status/1672374464418394112?s=20

To learn how your organization can build cyber resilience and prevent business disruption with eSentire’s Next Level MDR, connect with an eSentire Security Specialist now.

GET STARTED

ABOUT ESENTIRE'S THREAT RESPONSE UNIT (TRU)

The eSentire Threat Response Unit (TRU) is an industry-leading threat research team committed to helping your organization become more resilient. TRU is an elite team of threat hunters and researchers that supports our 24/7 Security Operations Centers (SOCs), builds threat detection models across the eSentire XDR Cloud Platform, and works as an extension of your security team to continuously improve our Managed Detection and Response service. By providing complete visibility across your attack surface and performing global threat sweeps and proactive hypothesis-driven threat hunts augmented by original threat research, we are laser-focused on defending your organization against known and unknown threats.

Source: https://www.esentire.com/blog/the-case-of-lummac2-v4-0