1.1 OVERVIEW / In 2014, PLXsert has observed a trend in new distributed denial of service (DDoS) malware originating from Asia. These binaries have been targeting Linux operating systems principally, but now PLXsert has identified a new malware kit that can also infect Windows systems and embedded devices. Several iterations of the Spike DDoS toolkit can communicate and execute commands to infected Windows, desktop Linux and ARM-based devices running the Linux operating system (OS).

Binary payloads from this toolkit are dropped and executed after the successful compromise of targeted devices, which may include PCs, servers, routers, Internet of Things (IoT) devices (i.e., smart thermostat systems and washer/dryers) and home-based customer premises equipment (CPE) routing devices.

The toolkit has multiple DDoS payloads, including SYN flood, UDP flood, Domain Name System (DNS) query flood, and GET floods. Several campaigns have been reported against hosts in Asia and the U.S. Several Akamai customers have been targeted by DDoS attack campaigns launched from this botnet. One attack peaked at 215 gigabits per second (Gbps) and 150 million packets per second (Mpps).

1.2 INDICATORS OF BINARY INFECTION / The principal indicators of infection are the presence of a series of binaries that infect specific operating systems and architectures. PLXsert analyzed binary payloads associated with the Spike DDoS toolkit that targeted desktop Linux OSs and ARM-based Linux hosts. Russian anti-virus company Doctor Web also reported on what may be iterations of the toolkit, and evidence of the payloads being ported to Windows has surfaced. The binaries associated with the Spike DDoS toolkit consists of one binary, while the iterations found by DrWeb may include several different binaries and other scripts associated with an infection.

1.3 TOOLKIT ANALYSIS / The Spike DDoS toolkit contains components of a typical client-based botnet: a command and control (C2) panel, binary payloads for infection and DDoS payload builders. The C2 and the builders are Windows binaries for use by the malicious actor, while the infectious payloads were designed to target mainly Linux or other embedded devices. The ability of the Spike toolkit to generate an ARM-based payload suggests that the authors of such tools are targeting devices such as routers and IoT devices to expand their botnets for a post-PC era of botnet propagation.

---

1 "Linux Trojan Ported to Windows." Dr. Web Anti-virus. Doctor Web, 14 Aug. 2014.
The C2 panel contains a fairly basic interface written in Mandarin Chinese. It consists of a top panel for listing the connected bots, a bottom tabbed interface for task information, and subsequent tabs for the different types of DDoS payloads it supports. Figure 1 shows the main C2 panel interface with one connected bot in a lab environment.

1.3 Spike Bot Variations / The Spike DDoS toolkit analyzed by PLXsert included a C2 panel and three payload builders. Two of the builders generate 32-bit and 64-bit Linux payloads. The third, called Typhoon builder, generates a 32-bit ARM executable (Figure 2). The addition of the ARM payload suggests that the author of the toolkit may intend to infect devices beyond PCs and servers. Routers, CPEs and other embedded devices running Linux could be the target of such an ARM-compatible payload.

```
plxsert$ file DDoS_arm
DDoS_arm: ELF 32-bit LSB executable, ARM, version 1 (SYSV), statically linked, for GNU/Linux 2.6.14, stripped
```

Figure 2: The output of the `file` command on the ARM DDoS bot payload generated by the Typhoon builder
Figure 3 shows the presence of string data found in the ARM-compatible binary. This string data is also present in the other Linux binary payloads available from the Spike DDoS toolkit.

```
.text:0000A1DC
.text:0000A1E0
.text:0000A1E4
.text:0000A1E8
.text:0000A1EC
.text:0000A1F0
.text:0000A1F4
.text:0000A1F8
.text:0000A1FC
.text:0000A200
.text:0000A204
.text:0000A208
.text:0000A20C
.text:0000A210
.text:0000A214
.text:0000A218
.text:0000A21C
.text:0000A220
.text:0000A224
.text:0000A228
.text:0000A22C
.text:0000A230
.text:0000A234
BL sub_28AF0
ADD R4, SP, #0xBF0+var_BD0
LDR R12, =aMr_black ;"Mr.Black"
SUB R4, R4, #0xC
MOV R5, R0
MOV R3, R7
MOV R1, #0x400
LDR R2, =aVers0nexSDDS ;"VERS0nex:%s|%d|%d|%s"
MOV R0, R4
STMEA SP, {R5,R6,R12}
STR R5, [SP,#0xBF0+var_50]
BL sub_1AAB4
MOV R0, R4
BL sub_2D60
LDR R3, [R10]
ADD R2, R0, R6
MOV R1, R4
MOV R0, R3
BL sub_DCD0
ADD R6, SP, #0xBF0+var_140
ADD R9, SP, #0xBF0+var_40
ADD R6, R6, #8
ADD R9, R9, #8
```

Figure 3: A disassembly of the ARM bot payload
PLXsert tested the ARM bot using a Raspberry PI, a popular programmable embedded device. Figure 4 and Figure 5 demonstrate a proof-of-concept of an ARM-based bot running on the embedded device.

The introduction of a multi-platform DDoS toolkit such as the Spike DDoS toolkit indicates the direction that malicious actors are taking. The ARM payload for Linux could be used to target popular embedded devices, CPEs and Internet of Things devices; at least the subset of those devices that can be exploited and on which remote code execution can be attained.
1.3B BOT INITIALIZATION / The Spike DDoS toolkit analyzed by PLXsert is capable of creating three types of binary payloads: a 32-bit Linux binary, a 64-bit Linux binary and a 32-bit ARM-based binary. Initial analysis showed that all the payloads have the same capabilities. Once a binary is configured and built, it can be run on a victim host and will call back home to its C2. The initial call-home packet, shown in Figure 6, contains information about the infected machine such as its kernel version and CPU information. It also contains an extra string that may identify the author of the tool as Mr. Black.

Figure 6: The initial call-home packet to identify the victim machine to the C2

1.4 DDoS PAYLOADS / The Spike DDoS toolkit can be used to launch several types of denial of service attacks. The C2 panel indicates the capability to launch four DDoS attacks: SYN, DNS, UDP and GET floods. These also appear in the bot payloads in a function named DealwithDDoS(), which is shown in Figure 7. This function receives the argument from the C2 and then parses the commands. Once it determines which attack to perform, the function creates a thread and calls the corresponding flood function.
int __cdecl DealWithDDoS(void *arg)
{
    int result; // eax@1
    int i; // [sp+1Ch] [bp-Ch]@3
    int j; // [sp+1Ch] [bp-Ch]@7
    int k; // [sp+1Ch] [bp-Ch]@11
    int l; // [sp+1Ch] [bp-Ch]@15
    int m; // [sp+1Ch] [bp-Ch]@19

    result = StopFlag;
    if ( StopFlag )
    {
        StopFlag = 0;
        result = *(((DWORD *)arg) + 65);
        switch ( result )
        {
            case 1:
                for ( i = 0; ; ++i )
                {
                    result = *(((DWORD *)arg) + 66);
                    if ( result <= i )
                        break;
                    pthread_create((pthread_t *)&id[4 * i], 0, SYN_Flood, arg);
                }
                break;
            case 2:
                for ( j = 0; ; ++j )
                {
                    result = *(((DWORD *)arg) + 66);
                    if ( result <= j )
                        break;
                    pthread_create((pthread_t *)&id[4 * j], 0, UDP_Flood, arg);
                }
                break;
            case 5:
                for ( k = 0; ; ++k )
                {
                    result = *(((DWORD *)arg) + 66);
                    if ( result <= k )
                        break;
                    pthread_create((pthread_t *)&id[4 * k], 0, (void (*)(void *))GET_Flood, arg);
                }
                break;
            case 3:
                for ( l = 0; ; ++l )
                {
                    result = *(((DWORD *)arg) + 66);
                    if ( result <= l )
                        break;
                    pthread_create((pthread_t *)&id[4 * l], 0, ICMP_Flood, arg);
                }
                break;
            case 4:
                for ( m = 0; ; ++m )
                {
                    result = *(((DWORD *)arg) + 66);
                    if ( result <= m )
                        break;
                    pthread_create((pthread_t *)&id[4 * m], 0, DNS_Flood, arg);
                }
                break;
            default:
                return result;
        }
    return result;
}
1.4A SYN FLOOD / The C2 panel provides options to control the target, size and bandwidth of a SYN flood payload. The default packet size is 40, which accounts for basic TCP/IP headers and a payload size of zero. However, the packet size can be adjusted by an attacker, as shown in Figure 8.

Figure 9 shows a SYN flood payload with the default size of 40 bytes and a payload with a user-defined size of 1024 bytes. The first payload only includes the IP and TCP headers with no data payload, while the second includes a data payload.

Default 0-byte payload flood
19:59:40.257221 IP 23.184.239.66.51517 > 192.168.20.80.80: Flags [S], seq 3376218112, win 512, length 0
E.="........B...P.=.P.=......P...?$..........

1024-byte payload flood
19:59:44.713925 IP 219.174.239.68.5685 > 192.168.20.80.80: Flags [S], seq 372572160:372573184, win 512, length 1024
E.="........B...P.=.P.=......P...?$..........
192.168.20.1..........................|.......@...@...@...$
.............................................|=..h...2...A..
...............{......=......n.@..................................
........................................................................
........................................................................
........................................................................
1.4b UDP FLOOD / As with the SYN flood, the Spike toolkit allows an attacker to specify a size for a UDP flood. The payload simply consists of hardcoded Xs. Figure 10 shows the C2 panel options for a UDP flood. Figure 11 shows the UDP flood payload.

![Figure 10: UDP flood options in the C2 panel](image)

![Figure 11: Lab-simulated UDP flood payload](image)

1.4c ICMP (PING) FLOOD / The Spike DDoS toolkit claims to implement an ICMP flood. However, an analysis of the DDoS traffic and the code made it evident that the implementation was flawed; the ICMP payload was crafted incorrectly. Figure 12 shows locations in the disassembly of the bot payload where the ICMP flood function is implemented. The protocol number passed to the socket function is 17 (UDP), and a hardcoded buffer size of 12 is used for the send() function call. This causes an ICMP flood command to send a hardcoded UDP flood payload of 12 bytes.
Figure 12: ICMP flood function with questionable implementation details

Figure 13 shows the C2 panel option for the ICMP flood. The field for size (default value of 2048) is unused in the code. The payload size always seems to be 12 bytes, as shown in Figure 14.

Figure 13: ICMP Flood options in the C2 panel

Figure 14 shows the lab generated ICMP flood. The implementation creates a UDP flood of 12 bytes instead of the proposed ICMP flood.

Figure 14: A lab-simulated ICMP flood payload from the Spike DDoS toolkit
1.4d DNS FLOOD / The input options for the DNS flood include a target DNS server, the domain to query and the number of threads and time elapsed, which are shown in Figure 15. The toolkit will then generate random subdomains and random types of DNS queries to the DNS server. Figure 16 shows a DNS query flood payload generated by the Spike DDoS toolkit in a lab environment.

Figure 15: DNS flood options in the C2 panel


Figure 16: A lab-simulated DNS query flood from the Spike DDoS toolkit

1.4e GET FLOOD / The GET flood payload also uses hardcoded values. Figure 17 shows the target IP, port (set at 80) the number of threads (set at 10), and the packets per second (set at 20) in the C2 panel. In this advisory PLXsert provides a SNORT rule to identify and mitigate GET flood payload from the Spike DDoS toolkit. A lab-generated GET flood payload is shown in Figure 18.

Figure 17: GET flood options in the command and control panel

| Time: 10:44:33 | IP: 192.168.20.62.57765 | 192.168.20.80.80: Flags [FF.], seq 1:270, ack 1, win 115, options [nop,nop,TS val 2024120 ecr 29974], length 269 |
| Time: 10:44:33 | | E.A+0,@@..>,7...P...8...8...8...8...8.8...... |
| Time: 10:44:33 | | ......u.GET / HTTP/1.1 |
| Time: 10:44:33 | | Accept: text/html,application/xhtml+xml,*/* |
| Time: 10:44:33 | | Accept-Language: zh-CN |
| Time: 10:44:33 | | User-Agent: Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.1; WOW64; Trident/6.0) |
| Time: 10:44:33 | | Accept-Encoding: gzip, deflate |
| Time: 10:44:33 | | Host: 192.168.20.80 |
| Time: 10:44:33 | | Connection: Keep-Alive |
| Time: 10:44:33 | | Pragma: no-cache |

Figure 18: A lab-simulated GET flood payload from the Spike DDoS toolkit
1.5 OBSERVED ATTACK / Figure 19 highlights an actual DDoS attack mitigated by Akamai from a Spike DDoS botnet. The metrics show the distribution of the attack at each of Akamai’s DDoS scrubbing centers. This attack peaked at 215 Gbps and 150 Mpps.

<table>
<thead>
<tr>
<th>Akamai Scrubbing Center</th>
<th>San Jose</th>
<th>London</th>
<th>Hong Kong</th>
<th>WashingtonDC</th>
<th>Frankfurt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peak bits per second (bps)</td>
<td>25 Gbps</td>
<td>40 Gbps</td>
<td>30 Gbps</td>
<td>70 Gbps</td>
<td>50 Gbps</td>
</tr>
<tr>
<td>Peak packets per second (pps)</td>
<td>14 Mpps</td>
<td>35 Mpps</td>
<td>28 Mpps</td>
<td>28 Mpps</td>
<td>45 Mpps</td>
</tr>
</tbody>
</table>

Figure 19: Attack campaign distribution by scrubbing center

1.6 DDOS MITIGATION / From the target’s perspective, most of the Layer 3 DDoS attack signatures can be mitigated by implementing access control lists (ACLs). For the layer 7 GET flood, PLXsert has produced a SNORT signature to defend against this attack.

1.6a SNORT RULE FOR DDOS PROTECTION / This rule assumes that the host header and domain are no larger than 58 bytes.

```
alert TCP $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS 
(msg: "Spike";
  flow: to_server;
  content: !"Cookie"; depth:306;
  content: "GET / HTTP/1.1"; depth:14;
  content: "Accept": text/html, application/xhtml+xml, */*; distance:2; within:45;
  content: "Accept-Language": zh-CN; distance:2; within:22;
  content: "User-Agent": Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.1;
  Trident/6.0); distance:2; within:83;
  content: "Accept-Encoding": gzip, deflate; distance:2; within:30;
  content: "Host": distance:2; within:4;
  content: "Connection": Keep-Alive; distance:2; within:70;
  content: "Pragma": no-cache; distance:2; within:16;
  classtype: GET-Flood;
  sid:201400000; rev:1;)
```

Figure 20: A PLXsert SNORT rule to stop GET floods from the Spike DDoS toolkit
1.6B SYSTEM HARDENING / The multi-architecture malware code found in this kit increases the threat’s complexity and sophistication and makes it necessary to apply hardening measures to each of the targeted operating systems and platforms. The following organizations provide guidance for system hardening:

- SANS Institute checklist for Linux server hardening
- Microsoft list of baseline Windows hardening guidelines to secure windows servers
- The National Security Agency guidelines for securing various operating systems
- Tripwire security recommendations for small office, home office (SOHO) routers
- NIST guide to secure mobile devices in the enterprise
- University of Michigan guide to securing and managing Android security phones/tablet
- OWASP list of the 10 most critical Internet of Things (IoT) security risks

---

9 "OWASP Internet of Things Top Ten Project." Open Web Application Security Project (OWASP). HP Fortify on Demand.
1.6c YARA RULE / YARA is an open source tool designed to identify and classify malware threats. It is typically used as a host-based detection mechanism and provides a strong Perl-compatible regular expressions (PCRE) engine to match identifying features of threats at a binary level or more. PLXsert utilizes YARA rules to classify threats that persist across many campaigns and over time. Figure 21 contains a YARA rule provided by PLXsert to identify the bot payloads associated with the Spike DDoS Toolkit.

```bash
rule Spikev4
{
    meta:
        author = "PLXsert"
        description = "Rule to detect the Spike v4.0 Bot payloads"
    strings:
        $st0 = "Int Server...
        $st1 = "INFO:%d\r"
        $st2 = "VERSION:%s\r\d\s"
        $st3 = "Mr.Black"
        $st4 = "DealwithDDoS"
    condition:
        all of them
}
```

Figure 21: PLXsert YARA rule to detect the Spike DDoS toolkit bot payloads

1.7 CONCLUSION / There is a rising trend in botnet activity from Asia that has targeted Linux servers primarily, but is now diversifying and targeting Windows hosts, routers, CPE and ARM-compatible Linux distributions as well. DDoS attackers can gain additional resources by extending the range of devices that can be harnessed by a botnet. As a result, these botnets can produce significant DDoS attack campaigns. New DDoS kits, such as the Spike DDoS toolkit, are incorporating malware designed to target multiple devices and platforms, thus requiring system administrators to thoroughly check and harden devices that may not have been targeted or thought to be at risk for botnet infection in the past.

The Spike DDoS toolkit does not use new types of DDoS attacks. Most of its payloads are typical in any DDoS toolkit. In addition, its payload implementations are either fairly simplistic or implemented incorrectly. What Spike does bring, however, is diversity in infection with the introduction of ARM-based binary payloads.

These botnets will likely be used eventually in attack campaigns against targets in regions beyond Asia, and against a variety of verticals. Unless there are significant community cleanup efforts, this infestation is likely to spread and there is likely to be a surge in the number of new iterations that incorporate new payloads and signatures.

PLXsert will continue researching these binaries and provide further advisories should it be warranted.
The Prolexic Security Engineering and Research Team (PLXsert) monitors malicious cyber threats globally and analyzes these attacks using proprietary techniques and equipment. Through research, digital forensics and post-event analysis, PLXsert is able to build a global view of security threats, vulnerabilities and trends, which is shared with customers and the security community. By identifying the sources and associated attributes of individual attacks, along with best practices to identify and mitigate security threats and vulnerabilities, PLXsert helps organizations make more informed, proactive decisions.

Akamai® is a leading provider of cloud services for delivering, optimizing and securing online content and business applications. At the core of the company’s solutions is the Akamai Intelligent Platform™ providing extensive reach, coupled with unmatched reliability, security, visibility and expertise. Akamai removes the complexities of connecting the increasingly mobile world, supporting 24/7 consumer demand, and enabling enterprises to securely leverage the cloud. To learn more about how Akamai is accelerating the pace of innovation in a hyperconnected world, please visit www.akamai.com or blogs.akamai.com, and follow @Akamai on Twitter.

Akamai is headquartered in Cambridge, Massachusetts in the United States with operations in more than 40 offices around the world. Our services and renowned customer care enable businesses to provide an unparalleled Internet experience for their customers worldwide. Addresses, phone numbers and contact information for all locations are listed on www.akamai.com/locations

©2014 Akamai Technologies, Inc. All Rights Reserved. Reproduction in whole or in part in any form or medium without express written permission is prohibited. Akamai and the Akamai wave logo are registered trademarks. Other trademarks contained herein are the property of their respective owners. Akamai believes that the information in this publication is accurate as of its publication date; such information is subject to change without notice. Published 09/14.