IOActive Security Advisory

Title	pppd vulnerable to buffer overflow due to a flaw in EAP packet processing (CVE-2020-8597)
Severity	High
Discovered by	Ilja Van Sprundel, Director of Penetration Testing
Advisory Date	March 4, 2020

Affected Products

Cisco, Debian GNU/Linux, Fedora Project, NetBSD, OpenWRT, Red Hat, SUSE Linux, Synology, TP-LINK, Ubuntu

Impact

By sending an unsolicited EAP packet to a vulnerable ppp client or server, an unauthenticated remote attacker could cause memory corruption in the pppd process, which may allow for arbitrary code execution.

Background

pppd (Point to Point Protocol Daemon) versions 2.4.2 through 2.4.8 are vulnerable to buffer overflow due to a flaw in Extensible Authentication Protocol (EAP) packet processing in eap_request and eap_response subroutines.

Technical Details

PPP is the protocol used for establishing internet links over dial-up modems, DSL connections, and many other types of point-to-point links including Virtual Private Networks (VPN) such as Point to Point Tunneling Protocol (PPTP). The pppd software can also authenticate a network connected peer and/or supply authentication information to the peer using multiple authentication protocols including EAP.

Due to a flaw in the Extensible Authentication Protocol (EAP) packet processing in the Point-to-Point Protocol Daemon (pppd), an unauthenticated remote attacker may be able to cause a stack buffer overflow, which may allow arbitrary code execution on the target system. This vulnerability is due to an error in validating the size of the input before copying the supplied data into memory. As the validation of the data size is incorrect, arbitrary data can be copied into memory and cause memory corruption possibly leading to execution of unwanted code.

The vulnerability is in the logic of the eap parsing code, specifically in the eap_request() and eap_response() functions in eap.c that are called by a network input handler. These functions take a pointer and length as input using the the first byte as a type. If the type is EAPT_MD5CHAP(4), it looks at an embedded 1-byte length field. The logic in this code is intended to make sure that embedded length is smaller than the whole packet length. After this verification, it tries to copy provided data (hostname) that is located after the embedded length field into a local stack buffer. This bounds check is incorrect and allows for memory copy to happen with an arbitrary length of data.

An additional logic flaw causes the eap_input() function to not check if EAP has been negotiated during the Line Control Protocol (LCP) phase. This allows an unauthenticated attacker to send an EAP packet even if ppp refused the authentication negotiation due to lack of support for EAP or due to mismatch of an agreed pre-shared passphrase in the LCP phase. The vulnerable pppd code in eap_input will still process the EAP packet and trigger the stack buffer overflow. This unverified data with an unknown size can be used to corrupt memory of the target system. The pppd often runs with high privileges (system or root) and works in conjunction with kernel drivers. This makes it possible for an attacker to potentially execute arbitrary code with system or root level privileges.

The pppd software is also adopted into lwIP (lightweight IP) project to provide pppd capabilities for small devices. The default installer and packages of lwIP are not vulnerable to this buffer overflow. However, if you have used the lwIP source code and configured specifically to enable EAP at compile time, your software is likely vulnerable to the buffer overflow. The recommended update is available from Git repoistory http://git.savannah.nongnu.org/cgit/lwip.git.

This type of weakness is commonly associated in Common Weakness Enumeration (CWE) with CWE-120 Buffer Copy without Checking Size of Input ('Classic Buffer Overflow').

Suggested Fixes

Update your software with the latest available patches provided by your software vendor. It is incorrect to assume that pppd is not vulnerable if EAP is not enabled or EAP has not been negotiated by a remote peer using a secret or passphrase. This is due to the fact that an authenticated attacker may still be able to send unsolicited EAP packet to trigger the buffer overflow.

If your software is packaged and created from the ppp source code, please obtain the latest software from github pppd repository.
https://github.com/paulusmack/ppp

Patch referenced:
https://github.com/paulusmack/ppp/commit/8d7970b8f3db727fe798b65f3377fe6787575426

In case of lwIP package that is compiled from source with EAP enabled at compile time, obtain the latest software from github
http://git.savannah.nongnu.org/cgit/lwip.git

Patch referenced:
http://git.savannah.nongnu.org/cgit/lwip.git/commit/?id=2ee3cbe69c6d2805e64e7cac2a1c1706e49ff
d86

Note: the latest software also includes ignoring out-of-order or unsolicited EAP packets from being processed as an additional precautionary measure. It is recommended that you use the latest available software from the appropriate Git repository that includes this fix.

There is no viable work around except to patch the software with updated software made available by the software vendors.

References

- https://nvd.nist.gov/vuln/detail/CVE-2020-8597
- https://vulners.com/cve/CVE-2020-8597
- https://github.com/paulusmack/ppp/commit/8d45443bb5c9372b4c6a362ba2f443d41c5636af
- http://git.savannah.nongnu.org/cgit/lwip.git/commit/?id=2ee3cbe69c6d2805e64e7cac2a1c1706e49ffd86
- http://git.savannah.nongnu.org/cgit/lwip.git/commit/?id=d281d3e9592a3ca2ad0c3b7840f8036facc02f7b

Special Thanks

Special thanks to Vijay S. Sarvepalli from cert/cc and Paul Mackerras (pppd author) for the work and contributions in helping resolve the issue.

Timeline

- 12/16/2019 - vulnerability discovery
- 12/17/2019 - contacted cert/cc
- 12/18/2019 - cert responds, suggest making POC and reaching out to to ppp author directly
- 12/19/2019 - created poc, reached out to pppd author, responded to cert/cc
- 12/20/2019 - response from cert/cc says a lot of staff is out for holidays, could cause some delayed responses
- 01/05/2020 - no response from ppp author yet, reaching out again. this time also on a 2nd email address
- 01/06/2020 - response from pppd author, acknowledges the issue. Informed cert/cc we made contact with pppd author
- 01/15/2020 - sync up email threads between cert/cc and pppd author
- 01/16/2020 - pppd author bug is only reachable if CHAP or SRP secret is configured
- 01/16/2020 - IOActive is skeptical about this claim. shows some evidence and asks pppd author to look into it
- 01/16/2020 - pppd author acknowledges that the previous claim is not actually implemented. unsolicited eap messages will get parsed and can trigger the issue
- 02/03/2020 - pppd author asks to review patch
- 02/03/2020 - IOActive reviewed patch, responds saying the fix looks good
- 02/03/2020 - got message saying CVE 'CVE-2020-8597' was assigned for this issue
- 02/03/2020 - patch to fix issue is publicly committed. same for path to ignore unsolicited eap messages
- 02/04/2020 - got message from cert/cc, saying they're about to start communications soon with vendors and linux distros
- 03/04/2020 - cert/cc publishes Vulnerability Note VU#782301 covering this issue