CVE-2021-33912 and CVE-2021-33913: Heap overflows in email validation library LibSPF2

18 Jan 2022

Abstract

Two bugs related to the parsing of SPF (Sender Policy Framework) records have been found in LibSPF2, a library commonly used to determine the validity of email received to a server. One of these bugs allows for relatively flexible memory corruption in the heap, while the other affects only up to four bytes past the end of an allocated buffer. Both bugs have the potential to be used to carry out 0-click remote code execution, though modern memory defenses may reduce the possibility of such an attack. These vulnerabilities are especially important to note for servers designed to receive or exchange email, as a successful exploit would compromise the privacy and security of email accounts on a given email server. This could additionally lead to the compromise of other online accounts (such as those that would send password reset links to the compromised email account), or serve as a starting point for more sophisticated phishing/spoofing attacks. Both heap overflows can be triggered remotely by first publishing a particular TXT record for an arbitrary domain and then sending an email from that domain to an affected email server.

Recommendations

Versions of LibSPF2 before 1.2.11 are affected by both of these vulnerabilities.

If you are a mail exchange, you should check to see whether your servers use a vulnerable version of LibSPF2. You should check any anti-SPAM devices you may run in addition to mail server software.

If you are a vendor of mail server software or anti-SPAM services, or if you manage packages for an operating system, you should determine whether LibSPF2 is used in any of your configurations and migrate to LibSPF2 1.2.11. This version can be found at:

https://github.com/shevek/libspf2/tree/8131fe140704eaae695e76b5cd09e39bd1dd220b

If you use another implementation of SPF, it is unlikely that you will be affected by this issue– the parsing bug is due to specific errors made in the LibSPF2 codebase.

Details

The Sender Policy Framework specification (RFC 7208) provides several different mechanisms and macros to provide some level of flexibility for SPF policies. SPF macros are expanded at the time an SPF policy is retrieved by an email server, and they are meant to represent information specific to a given email exchange (such as the HELO domain or IP address that email is being received from).

SPF macros include several indicators that can alter the way macro expansions occur, including options that reverse and/or truncate portions of the macro expansion and an option to

URL-encode the resulting string. Both vulnerabilities in LibSPF2 were found in code implementing these macro options.

CVE-2021-33912 is the result of incorrect parsing in the URL-encode phase of macro expansion. During this phase, any non-ASCII or special characters are converted into their hexadecimal representation (' ' -> '%20', for instance); LibSPF2 does this through a call to `sprintf`. The library implicitly assumes the function will only ever output 4 characters, but certain inputs (0x80-0xff) will cause `sprintf` to overwrite 8 characters, thereby overflowing the buffer allocated to store the expansion.

As an example, the following TXT record and SMTP query could be used to trigger a buffer overflow:

TXT record for 'example.com': `v=spf1 %{L}.example.com -all`

SMTP query:

```
from smtplib import SMTP

TARGET_IP = '192.0.2.1'
TARGET_EMAIL = 'example@nathanielbennett.com'

with SMTP() as s:
    s.connect(host=TARGET_IP, port=25)
    s.ehlo(name="example.com")
    s.send(b'MAIL FROM:\xff@example.com>\r\n')
    s.docmd("RCPT", "TO:<%s>" % (TARGET_EMAIL,))
    s.docmd("DATA")
    s.send(bytes("\r\n\r\n.\r\n", 'ascii'))
    s.quit()
```

Note: some email server and spam filter implementations reject non-ASCII input in SMTP commands; triggering CVE-2021-33912 is a much more involved process when this is the case, though it is still possible.

CVE-2021-33913 involves an error during the domain label reversal/truncation phase of macro expansion. In the event that a macro is reversed, truncated *and* URL-encoded during expansion, the buffer of the macro output is erroneously sized based on the length of the leftmost label in the domain, rather than the size of the whole domain. This means that an attacker may use a domain such as "b.AAAAAAAAAAAAAAAAAAAAA.example.com" to overflow the heap with 'A' characters.

The following TXT record and SMTP query could be used to trigger a buffer overflow:

TXT record for 'example.com': `v=spf1 %{H5r}.example.com -all`

SMTP query:

```
from smtplib import SMTP

TARGET_IP = '192.0.2.1'
TARGET_EMAIL = 'example@nathanielbennett.com'

with SMTP() as s:
    s.connect(host=TARGET_IP, port=25)
    s.ehlo(name="b.AAAAAAAAAAAAAAAAAAA.example.com")
    s.docmd("MAIL", "FROM:<test.example.com>")
    s.docmd("RCPT", "TO:<%s>" % (TARGET_EMAIL,))
    s.docmd("DATA")
    s.send(b'\r\n\r\n.\r\n')
    s.quit()
```

CVE-2021-33913 provides greater control on the size of the allocated buffer, the amount of data that can be written into the heap and the byte values that can be used; as such, it is of greater concern than CVE-2021-33912.

Conclusion

Email has inadvertantly become a centralized location for backup authentication–most online accounts require an email address for the purpose of having somewhere to communicate password reset links to. As such, the risks of a compromised email server are no longer limited to the loss of privacy of users' emails. LibSPF2 is not used by any of the biggest email providors (Gmail, Microsoft, Yahoo and the like), though many smaller providers have been measured to be affected. Multiple services (including Exim) make use of the library.

There are options for SPF validation written in memory-safe code (such as pyspf/python-policyd-spf and Mail::SPF, written in Python and Perl respectively). Both of these are up-to-date with the latest RFC. LibSPF2 hasn't been under active development/maintenence for a couple years now–the last three commits to the code before recent CVE fixes were in 2017, 2016 and 2015. The sensible recommendation may be to switch configurations to one of the above implementations if your email server currently uses LibSPF2. It's just too easy for vulnerabilities to pop up in C code, especially when it comes to string parsing.

Special thanks goes to shevek, Paulo Smorigo and the IMAAL lab for all the help with patching and measuring these vulnerabilities!