Recent changes to Babel-HMAC
`draft-ietf-babel-hmac-04`

Juliusz Chroboczek

28 March 2019

Background

Babel-HMAC is an authentication mechanism for the Babel routing protocol.

Joint work with Clara Dô and Weronika Kołodziejak, based on the work of Denis Ovsienko (RFC 7298), with input from many people, notably Markus Stenberg.

- protects both unicast and multicast;
- invulnerable to replay;
- easy to implement:
  - no global clocks;
  - no persistent storage.
- algorithm agility:
  - MAC algorithm;
  - size of indices
  - size of challenge nonces.

Recent changes to the protocol

Babel-HMAC was designed in public, in what was described by an interested onlooker as a open and robust standardisation process.

Except for two recent changes:

- changed the mandatory to implement MAC algorithms :
  - MUST implement SHA-256;
  - SHOULD implement Blake2s;
- must discard cryptographic state after a bounded time with no traffic.

Change to MTI MAC algorithms

RFC 7298 had two MTI algorithms:

- SHA-1;
- RIPEMD-160.

SHA-1 is not recommended in new protocols.
RIPEMD-160 was described as “somewhat niche”.

In draft `draft-ietf-babel-hmac-04`:

- MUST implement SHA-256;
- SHOULD implement Blake2s.

Babeld and BIRD implement both. So should you.

The hash function doesn’t matter

Babel-HMAC is threatened by first preimage attacks,
not by collision attaks.
“Any” hash function is strong enough.

Babel-HMAC only protects control packets.
“Any” hash function is fast enough.

SHA-256 is overkill for our needs, but it doesn’t matter.

Blake2s

SHA-256 is overkill, but it doesn't matter — it's fast enough.

Yet, some people concerned about slow software routers with no hardware assist for SHA-256. They request a faster option.

Toke and Dave suggested Blake2s:

- blazingly fast in software;
- 128-bit length, 64-bit strength;
- integrates keying (no need for HMAC construction);
- based on similar principles as SHA-256.

Babeld and BIRD implement both. So should you.

Delayed packets

Babel-HMAC is invulnerable to replay.
(We haven't proved that yet.)

It is vulnerable to packets being delayed, e.g. by an attacker located at a network switch. No known attacks, but it makes us nervous.

Let's bound the amount of time a packet can be delayed by.

Bounding the delay

A node MAY discard its per-peer cryptographic state at any time, at the cost of one RTT of lost packets. Doing that invalidates any previously sent packets that have been delayed by the network.

New requirement:
A node MUST ensure that, in the absence of correctly authentified network traffic, cryptographic state is discarded after a bounded time. The mechanism is left unspecified, but the draft contains implementation advice.

Conclusion

Two (hopefully non-controversial) recent changes:

- changed the mandatory to implement MAC algorithms :
  - MUST implement SHA-256;
  - SHOULD implement Blake2s;
- must discard cryptographic state after a bounded time with no traffic.