QUIC Token Misdirection

Mike Bishop

Akamai

When a server receives an Initial packet with an address validation token, it MUST attempt to validate the token, unless it has already completed address validation. If the token is invalid, then the server SHOULD proceed as if the client did not have a validated address, including potentially sending a Retry packet. **Tokens provided with NEW_TOKEN frames and Retry packets can be distinguished by servers** (see Section 8.1.1), and the latter can be validated more strictly.

When a server receives an Initial packet with an address validation token, it MUST attempt to validate the token, unless it has already completed address validation. If the token is invalid, then the server SHOULD proceed as if the client did not have a validated address, including potentially sending a Retry packet. **Tokens provided with NEW_TOKEN frames and Retry packets can be distinguished by servers** (see Section 8.1.1), and the latter can be validated more strictly.

A token sent in a NEW_TOKEN frame or a Retry packet MUST be constructed in a way that allows the server to identify how it was provided to a client. These tokens are carried in the same field but require different handling from servers.

If a server receives a client Initial that contains an invalid Retry token but is otherwise valid[...] the server SHOULD immediately close (Section 10.2) the connection with an INVALID_TOKEN error.

Initial

NEW_TOKEN

Initial w/token

LGTM!

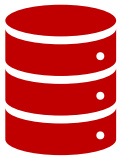
Initial

NEW_TOKEN

...

Initial w/token

CLOSE_CONNECTION
(INVALID_TOKEN)



Potential Fixes

- Shared token format, so that any server can discriminate NEW_TOKEN vs. Reset tokens
- Implementation-internal “actually mine” verification before trusting result
  - Potentially cryptographic, but remember servers do reboot

Is this an erratum? Draft? Tribal knowledge?