Token Binding for 0-RTT TLS 1.3 Connections

draft-nharper-0-rtt-token-binding
Nick Harper
IETF 97

Problem: RFC 5705 exporter only available after TLS handshake completes

0-RTT mode sends application data before the TLS handshake completes.

We’d like to put a TokenBindingMessage in this application data.

This means we need to sign a different exporter value in 0-RTT data.

How to choose exporter

- Must use early exporter in 0-RTT data
- If 0-RTT data accepted, use early exporter for the rest of the connection
- If 0-RTT data rejected, use 5705-style exporter for the rest of the connection

How to negotiate

Initial connection: Negotiate TB as in TBNEGO; server sends NewSessionTicket to use for 0-RTT connection

0-RTT connection:
Client sends same TBNEGO extension as previous connection, sends TokenBinding with signature of early exporter using key of same type as previously negotiated.
Server sends TBNEGO extension negotiating same type, or rejects 0-RTT data if it negotiates a different key type

Checking for replay protection

Client can send replay TLS indication extension to query if server has implemented some form of replay protection.

Server echoes it back if it does.

```
enum {
    token_binding_replay_indication(TBD), (65535)
} ExtensionType;
```

This extension always has zero length.

Opting out of TB+0-RTT

(Assuming both peers support TLS 0-RTT without TB)

Client opt-out: doesn’t send 0-RTT data

Server opt-out: rejects 0-RTT data

Security considerations

Pure PSK

- Attacker on-path with PSK can hijack an existing connection using Token Binding and send its own application messages using a TokenBindingMessage sniffed from earlier in the connection
  - This applies to connections that aren't 0-RTT
- With 0-RTT, this attacker can now get a TokenBindingMessage from an existing connection and replay it (with different application data) on new connections
  - Unless the server prevents the client from reusing a ClientHello.random value

PSK+(EC)DHE replay concerns

On-path attacker with PSK (but not (EC)DHE private value) can sniff TokenBindingMessage. This can be replayed with new application data in 0-RTT data, but requires sending the same ClientHello. The server can process the 0-RTT application data, but the attacker can’t complete the handshake.

Mitigations: Server-side

- Prevent all 0-RTT replay
  - Implement NewSessionTicket as session cache instead of encrypted tickets
  - Require (ClientHello.random, PSK) pair is unique for all time
- Prevent replay of TokenBinding
  - Allows moving replay protection from TLS layer to application layer
- Limit time window for ticket age

Mitigations: Client side

- Strongly protect TB key, and protect PSK just as strongly
  - Doesn’t prevent replay
  - Changes incentive for attacker from stealing PSK to stealing TB key