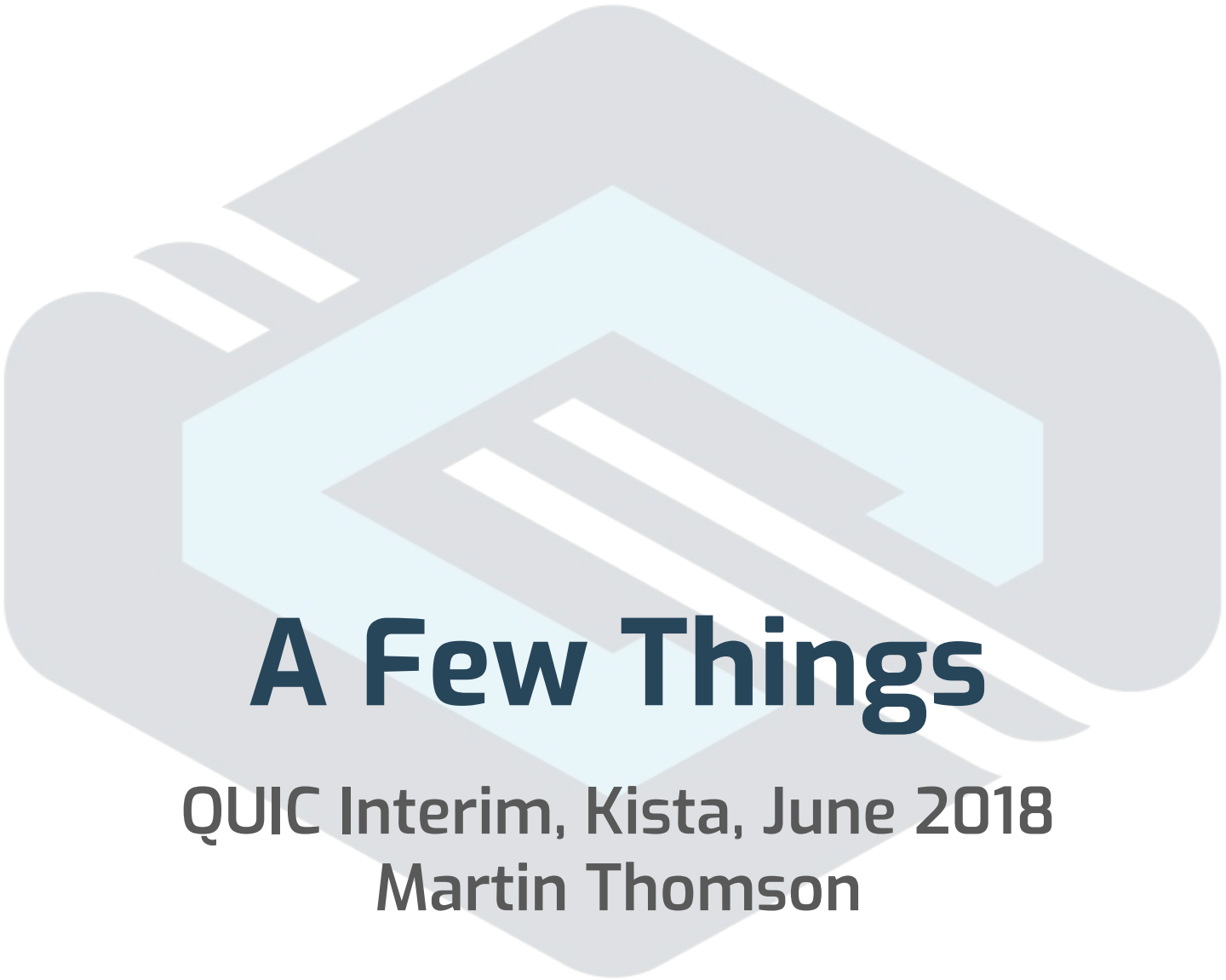


A Few Things

QUIC Interim, Kista, June 2018

Martin Thomson

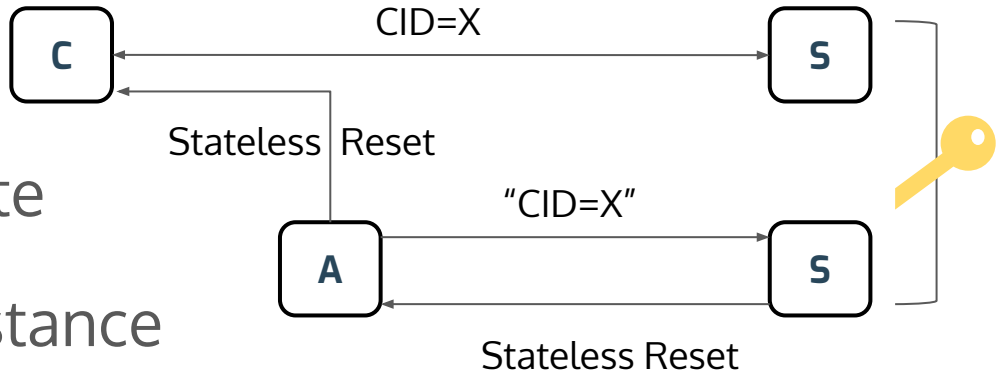
Stateless Reset Oracles

Server loses state

Or there is a change in route

Packet arrives at wrong instance

Server sends a stateless reset



If an attacker can send a packet toward a server that doesn't have state, it gets a stateless reset that the client will accept

Simple fix

Advise that configuring a cluster with a shared stateless reset key requires ensuring that routing uses connection ID

...and nothing that an attacker might control, such as source IP address or network location

Better fix?

Add proof of receipt to the stateless reset?

Simple change:

- last 16 octets is the stateless reset token
- preceding 16 octets is the last 16 octets of the packet that triggered the reset

Problems:

- endpoints probably don't remember packet ciphertext
- man on the side wins

Better, better fix?

Stateless reset is formed from:

- last 16 octets is H(token)
- preceding 16 octets is H(token||last 16 octets of received)

Defends against on-path attacker who never learns token

Doesn't fix the remembering issue though, but endpoints that don't remember can verify, like with RFC 5961

#1342 - Implicit Open

Yes

Consistent order of opening on both sides

No gaps

No

Just use identifiers

More code?

Can write on bidirectional streams that are “skipped”

Can’t write on bidirectional streams that are “skipped”

Our oldest issue

#58 - Frame Type Extensibility

No shortage of choices

1. Extension frame that embeds a type field
2. Registry for remaining types
3. Registry for remaining types + varint frame type
4. Each endpoint advertises frame types it accepts as a pair of type+semantic with (16+ bit) registry for semantics
5. Attach a list of frame types to every transport parameter, listing the frame types used

Stated principles

Extension use needs to be negotiated

All extensions should have access to a one octet encoding

- not just extensions that the IETF produces
- this rules out 1, 2, and 3

So, if we agree that these are desirable properties,

...can we choose from the drawbacks of 4 and 5?

Pick the least-worst

Against 4

An extension that has parameters and frames uses two transport parameters and has two registrations

Against 5

A transport parameter that isn't associated with the use of extension frames carries an empty list of frame types

Observation: extensions that want confidential negotiation can't use transport parameters anyway

#1016 - initial_max_stream_data

We have one value that governs all streams of all types:

bidirectional (mine)

bidirectional (yours)

unidirectional (yours)

How many transport parameters do we want? 1, 2, or 3.

#1296 - Negotiating Packet Number Protection

Goal: disable packet number protection when it isn't needed

Goal (?): have servers (only) make this decision

Options:

1. Transport parameter
2. New version

Both workable, but with transport parameters, Initial, 0-RTT, and Handshake can't be negotiated**

Clients could be forced to offer either, so servers can choose