

Happy Eyeballs, Version 3: *Better Connectivity Using Concurrency*

draft-ietf-happy-happyeyeballs-v3-01

Tommy Pauly, David Schinazi, Nidhi Jaju, Kenichi Ishibashi
HAPPY - IETF 123 Madrid - July 2025



Agenda

Updates in -01

Open Issues

Updates in -01

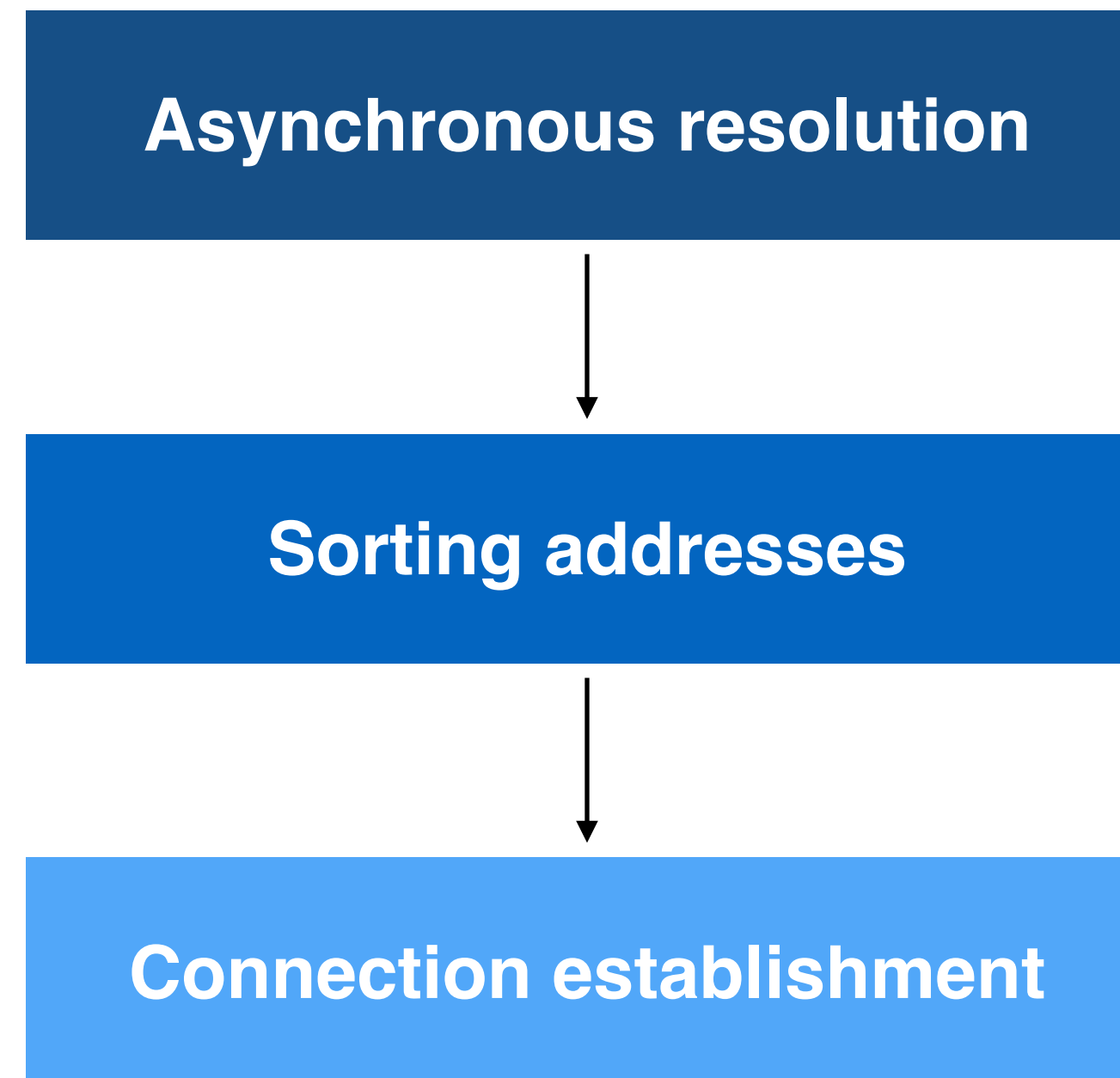
Changes to asynchronous resolution

Changes to address sorting

Handling of IPv6-mostly and PREF64

Editorial cleanup

Algorithm Recap



Asynchronous resolution

With just A/AAAA it was simple!

Adding SVCB/HTTPS makes it more complex

- Which queries to send?
 - A + AAAA, A + AAAA + HTTPS, A + HTTPS, AAAA + HTTPS, etc.
- When to move onto the sorting step

Asynchronous resolution: Queries

Query A on v4-only networks, A+AAAA on dual-stack networks,
A or A+AAAA on v6-only networks

Query HTTPS if you're an HTTP or WebSocket app

Send in order:

1. SVCB or HTTPS query
2. AAAA query
3. A query

Asynchronous resolution: When to move on

- Some positive (non-empty) address answers have been received

AND

- A positive (non-empty) or negative (empty) answer has been received for the preferred address family that was queried

AND

- SVCB/HTTPS service information has been received (or has received a negative response)

OR

- Some positive (non-empty) address answers have been received

AND

- A resolution time delay (50ms) has passed after which other answers have not been received

Note that SVCB address hints count as positive answers

Address sorting: Problem

At IETF 122 we discussed different sources of preferences

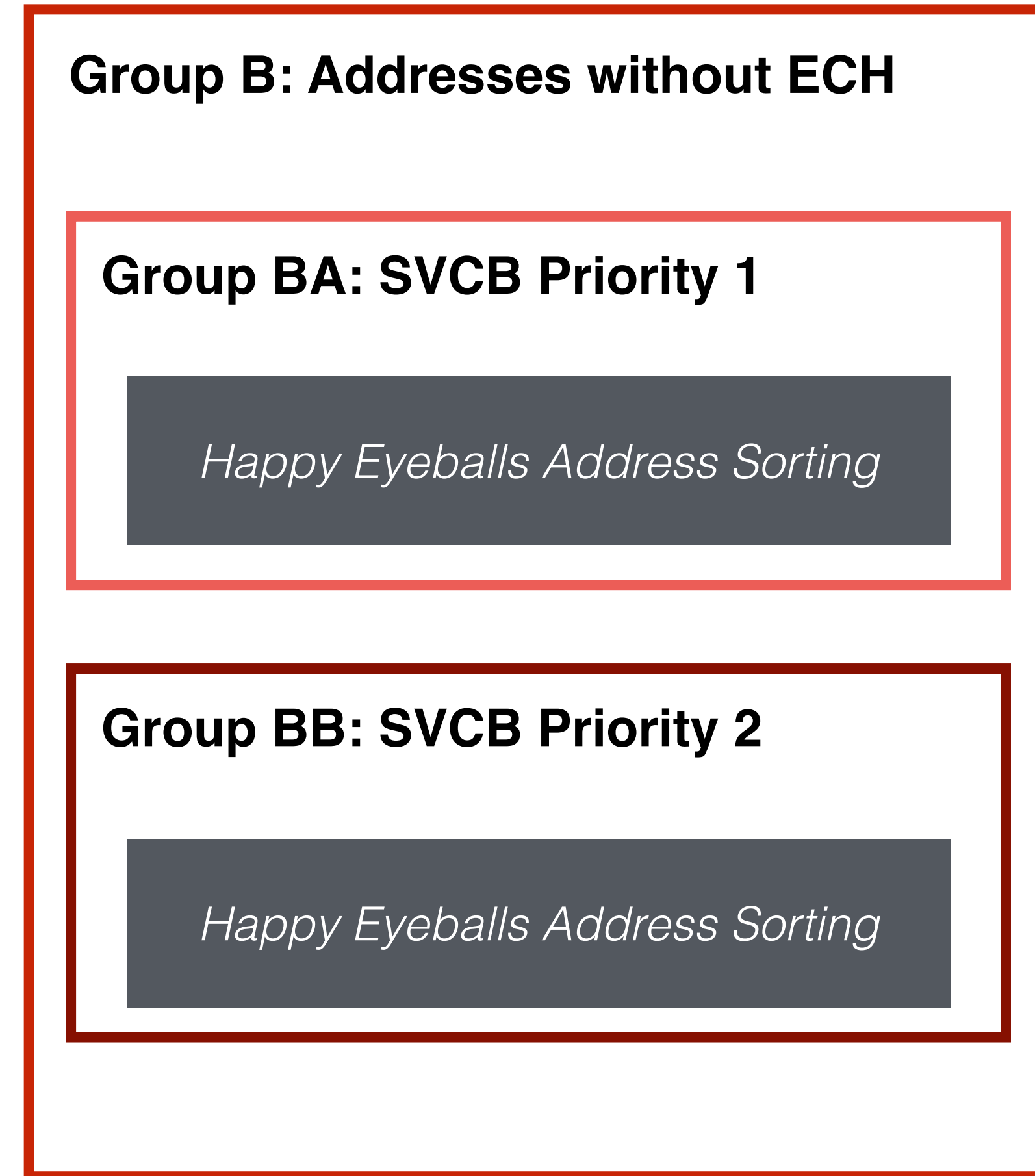
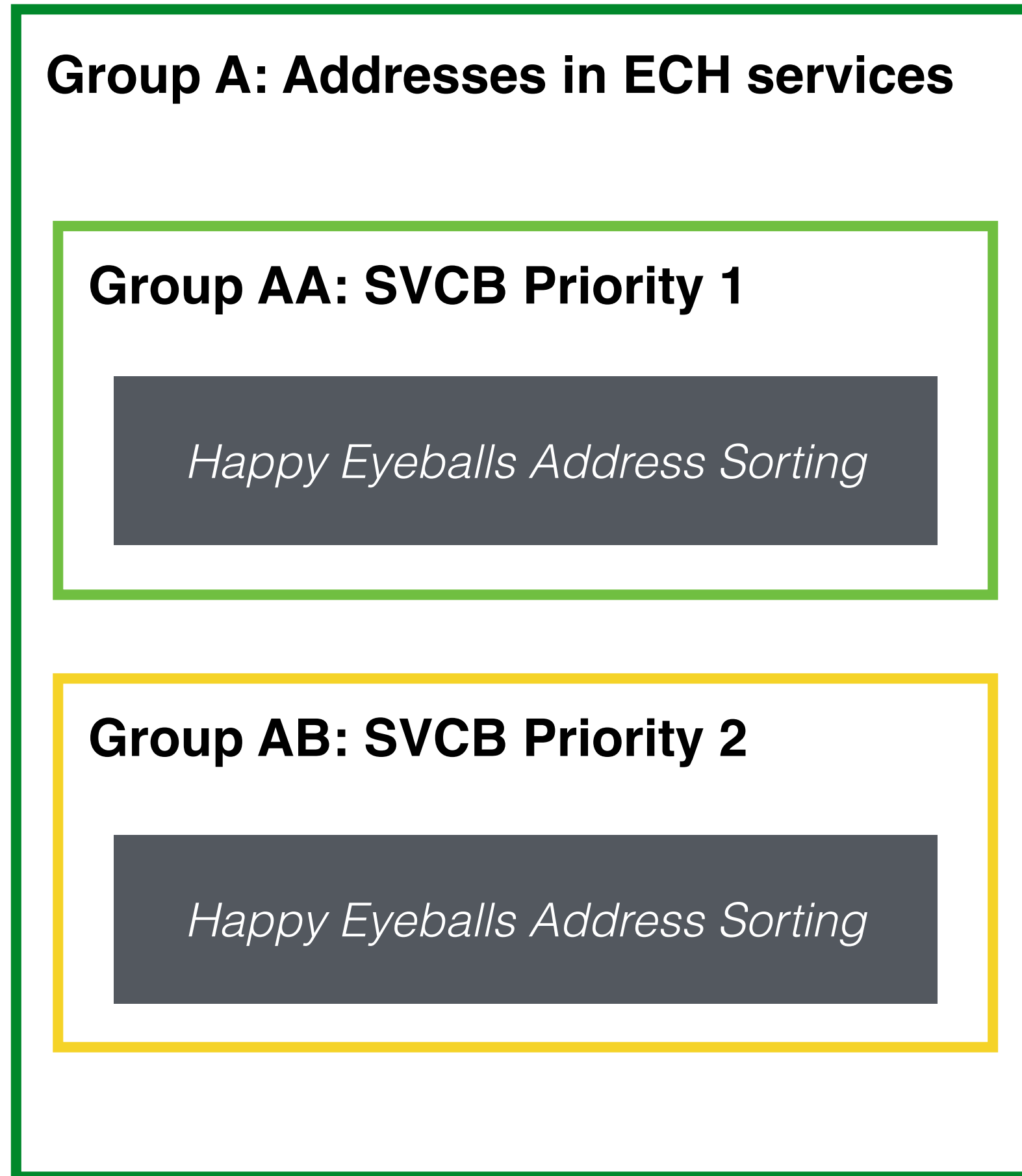
- Clients can prefer specific addresses (IPv6 preference, historically faster or successful options)
- Servers can publish explicit priorities (primary vs backup services)
- Applications may have requirements around security or protocol usage (TLS Encrypted Client Hello, QUIC vs TCP)

Address sorting: Solution

Group and sort in three phases:

1. Grouping and sorting by application protocol and security requirements
2. Grouping and sorting by service priorities
3. Sorting by destination address preferences

Address sorting: Solution



IPv6-only and IPv6-mostly

Explain IPv6-mostly

Make PREF64 behavior work without DNS64

Query both AAAA and A records

Translate SVCB address hints

Open Issues & Discussion Topics

ECH guidance conflicts with ECH-SVCB (#71)

Current text allows clients that prefer/require ECH to group addresses based on ECH-capability, potentially overriding SVCB priorities

Section 5.1:

"Clients first group based on which application protocols the destination endpoints support and which security features those endpoints offer. These are based on information from SVCB/HTTPS records about application-layer protocols ("alpn" values) and other parameters like TLS Encrypted Client Hello configuration ("ech" values, see [SVCB-ECH])."

Are we okay with the current ordering (protocol/security requirements take precedence), or do we want to allow servers to override application requirements?

Synthesized NAT64 over Native IPv4? (#42)

If an IPv6-mostly network provides DNS64, then a dual-stack host talking to an IPv4-only destination would have a choice between:

- a synthesized IPv6 address which will be routed via NAT64
- a native IPv4 path

Synthesized NAT64 over Native IPv4? (#42)

Dual Stack

Native IPv6
Native IPv4

OR

IPv6 Only, NAT64

Native IPv6
Synthesized IPv6

OR

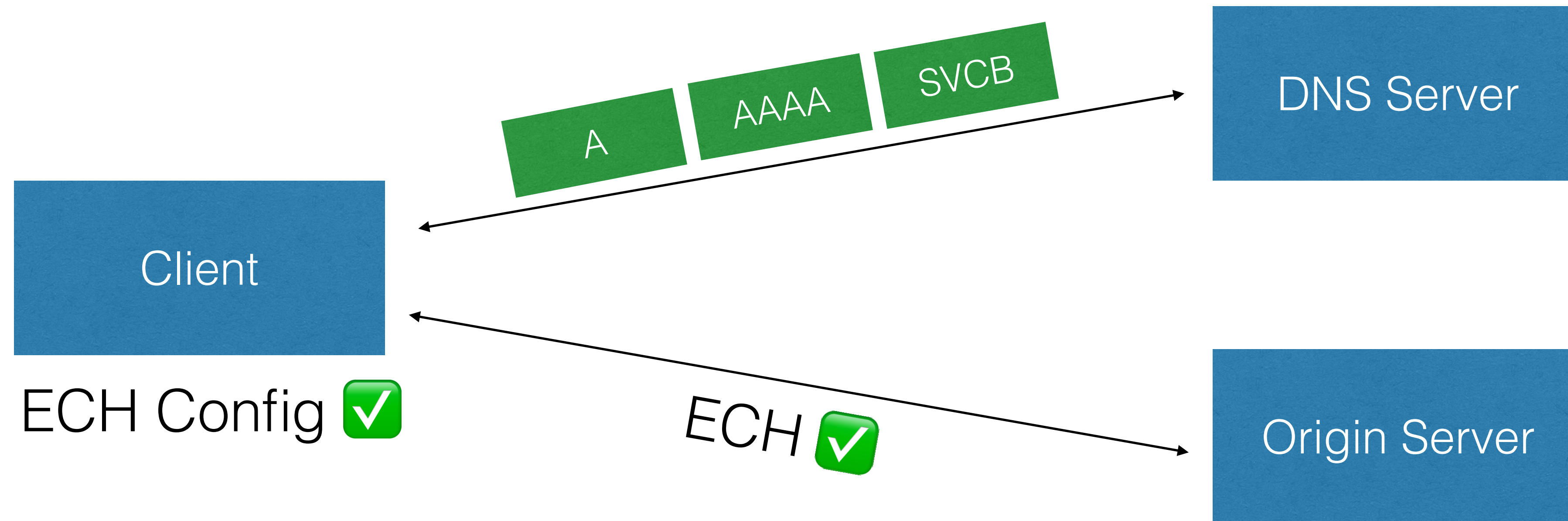
IPv6 Only, CLAT

Native IPv6
Synthesized IPv4

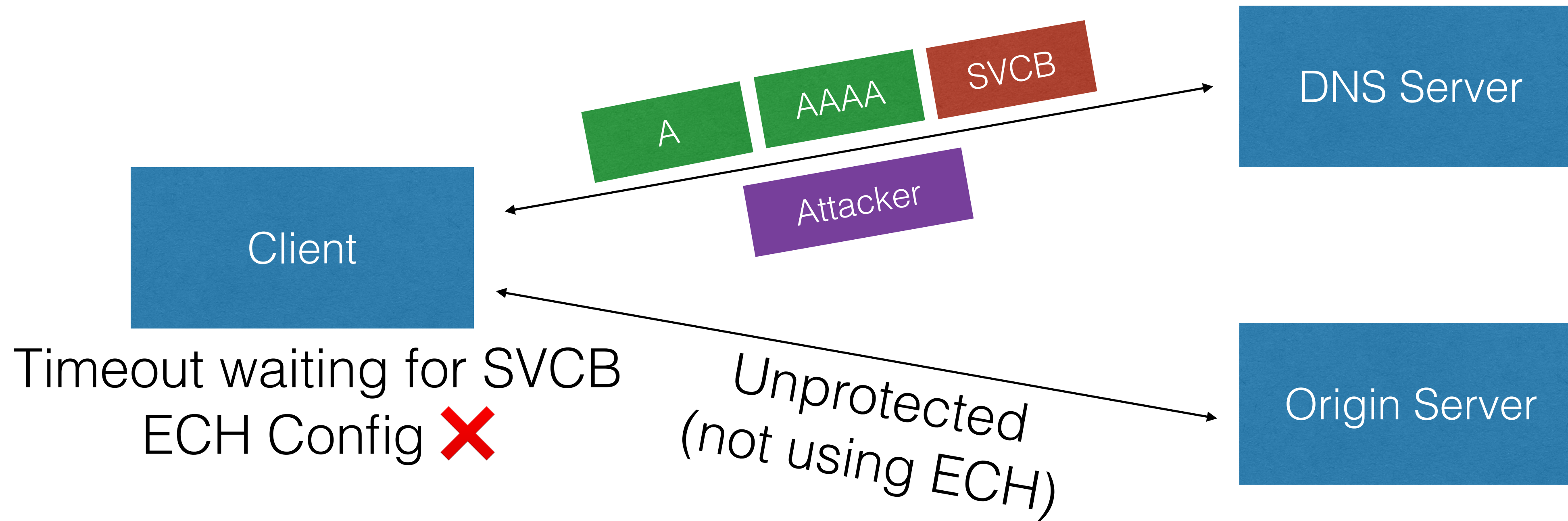
Downgrade attacks (#6, #24)

If attempts proceed without SVCB information, an attacker can target ECH (#6) or HTTPS (#24) information and effectively downgrade the client

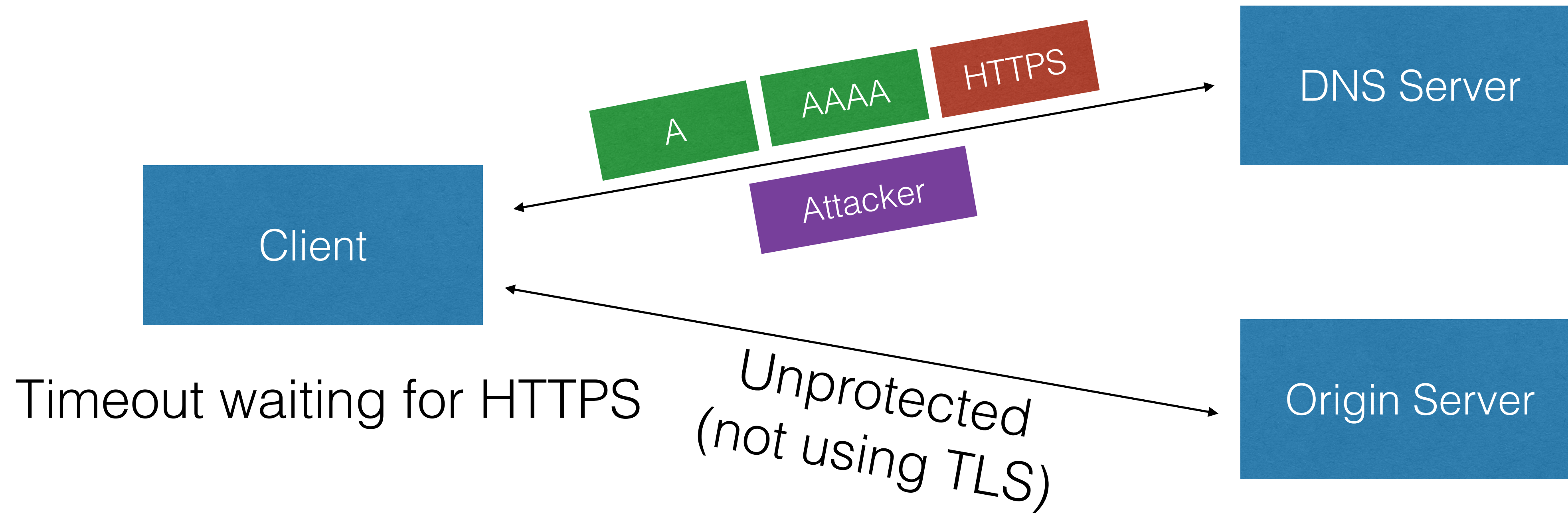
Downgrade attacks: Good case



Downgrade attacks: ECH downgrade (#6)



Downgrade attacks: HTTPS downgrade (#24)



Downgrade attacks (#6, #24)

Proposal:

- If a SVCB request is sent via secure transport (DoH or any other encrypted DNS transport), require a response before proceeding
 - Is this safe, or can this lead to indefinite stalls?
 - Option: Introduce a longer timeout, so we don't indefinitely stall
- If using cleartext DNS, don't wait beyond existing timer (50ms); the response is already unprotected

Retransmissions and next connection attempt timing (#1)

Section 6:

"delay for how long to wait before starting the next connection attempt. This delay is referred to as the "Connection Attempt Delay". ... Note that this algorithm should only try to approximate the time of the first handshake packet retransmission"

Should the first retransmission and next connection attempt happen at the same time? This could exacerbate packet loss.

Options:

- A. Send the first retransmission first
- B. Send the next connection attempt first

SvcPriority and complex clients (#69)

1. Can we race connections across multiple SvcPriorities, or should we wait until all higher priority endpoints fail?

Current text in Section 6:

"If grouping addresses by service (Section 5.2) produced multiple groups, all of the addresses of the first group SHOULD be started before starting attempts using the next group. Attempts across service groups SHOULD be allowed to continue in parallel; in effect, the groups are flattened into a single list."

Options:

- A. Keep current text: Allow lower priority attempts to proceed, sorting ensures higher priority attempts start first
- B. Change: Wait until failure because higher priorities take precedence within their group

SvcPriority and complex clients (#69)

2. How should historical information (slowness or previous failure) interact with SvcPriority? If all addresses in a priority group failed previously, what should the client do?

Options:

- A. Don't skip the group, run through attempts like normal
- B. Sort the group to the end
- C. Something else?

Target Fixation (#77)

Section 5.3:

"If the client keeps track of which addresses it used in the past, it SHOULD add another Destination Address Selection rule between the RTT rule and rule 9, which prefers used addresses over unused ones."

Concern: Causes stickiness when there are no higher order decision factors

Latest text groups by SvcPriority, so we would only have stickiness *within* a group of addresses with the same priority

Servers can adjust priorities to shift clients

Is this sufficient, given that priorities are a "higher order decision factor"?

Thank you!