CoAP Protocol Negotiation

draft-silverajan-core-coap-protocol-negotiation

Bill Silverajan   Tampere Univ of Technology

Background

- Aimed at CoAP endpoints wishing for multiple transports and/or locations to exchange CoAP requests and responses
- Transport availability falls into the following node categories
  - Type T0 nodes have a single transport
  - Type T1 nodes have 1 or more transports, which may be in unreachable/off states but at least 1 active transport
  - Type T2 nodes have multiple active transports

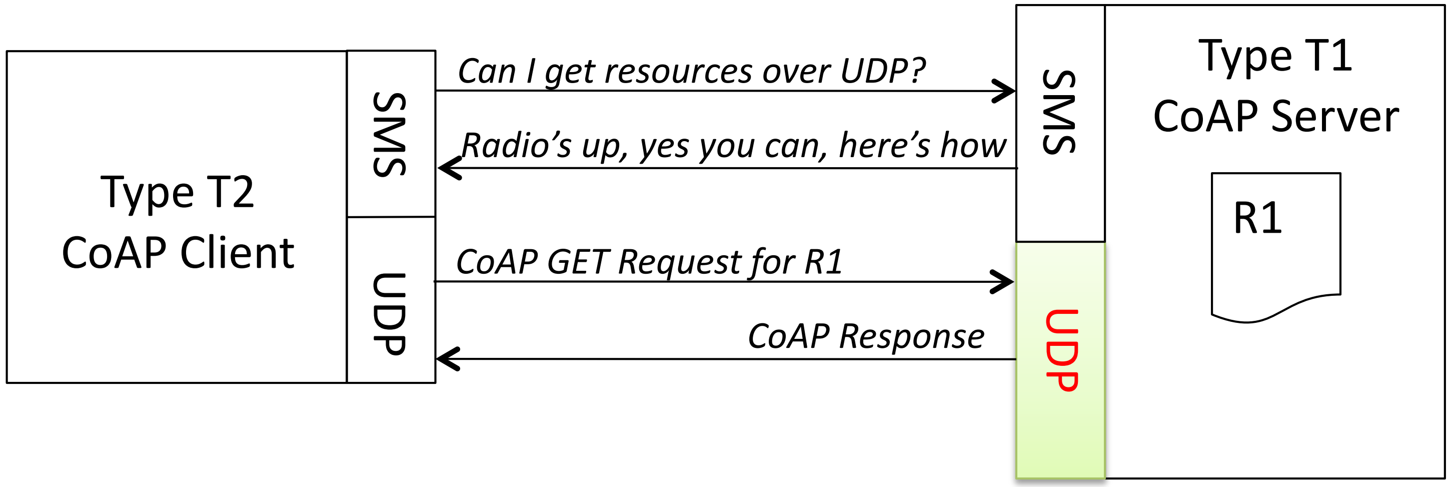
Why we <u>need</u> this

(..and we do ☺)

- Enables client-side discovery of server transports
- Reduces URI aliasing at origin server
- Eliminates URI path complexity

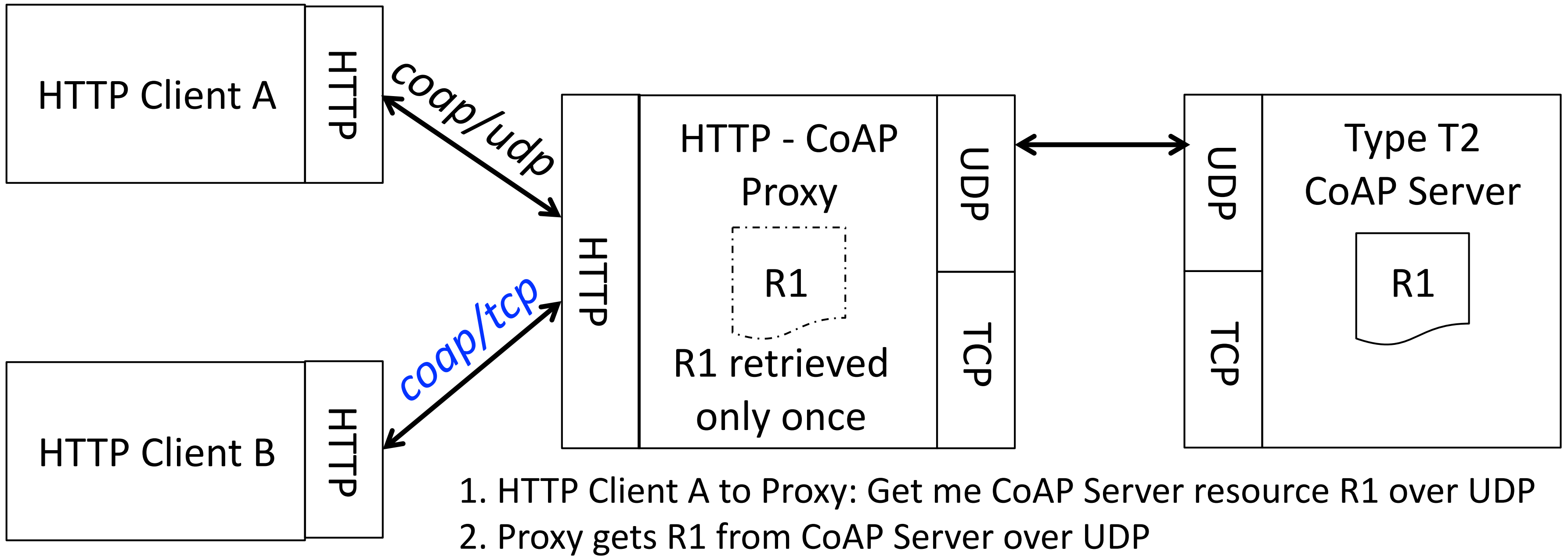
Allow Discovery

- CoAP clients to discover active transports on an origin server



Avoid URI aliasing

- Express same/related resource in alternate transports and locations



1. HTTP Client A to Proxy: Get me CoAP Server resource R1 over UDP
2. Proxy gets R1 from CoAP Server over UDP
3. HTTP Client B to Proxy: Get me CoAP Server resource R1 over TCP
4. Proxy to CoAP Server over UDP: Is it the same resource over TCP?
5. CoAP Server to Proxy over UDP: Yes, it is
6. Proxy Server returns cached R1 to HTTP Client B

Reduce URI path complexity

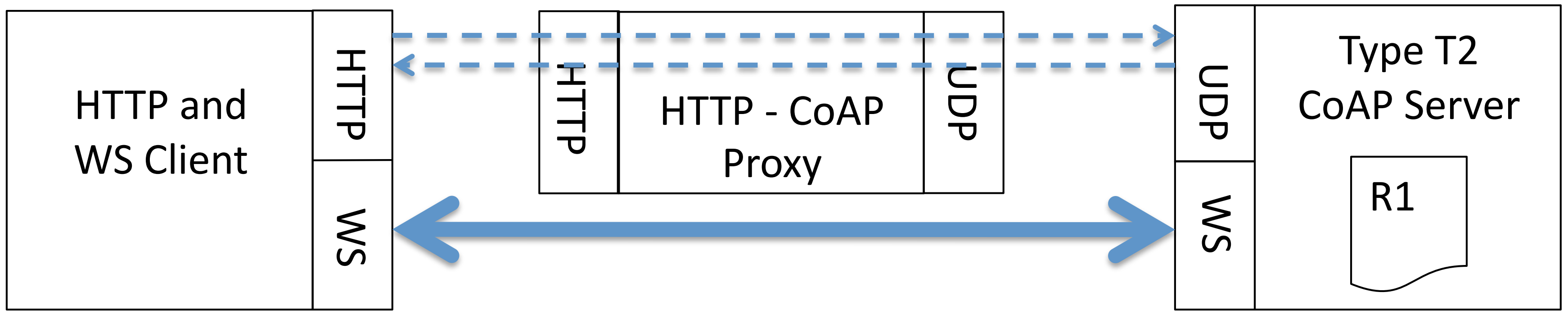
- Separate locator (endpoint subpath) from identifier (resource subpath)

Example CoAP over WebSocket URI from earlier work (discarded owing to complexity):

$$\underbrace{\text{coap-at:ws://www.example.com/WebSocket}}_{\text{WebSocket endpoint locator}}\text{?}\underbrace{\text{/sensors/temperature}}_{\text{CoAP resource Identifier}}$$

Reduce URI path complexity

- Separate locator (endpoint subpath) from identifier (resource subpath)



1. HTTP Client uses proxy to reach CoAP Server at UDP endpoint, server.example.com
2. HTTP Client solicits CoAP Server for WebSocket transport and endpoint info
3. CoAP Server responds giving WebSocket endpoint location as server.example.com/path/to/websocket
4. HTTP Client initiates WebSocket handshake with CoAP Server and negotiates CoAP subprotocol
5. Client switches to CoAP over WebSocket and retrieves resources from CoAP Server

How can this be achieved?

- Origin server simply exposes with .well-known/core:
  - a new link attribute “tt” containing list of priority ordered transport types for *coap* and *coaps* resources
  - a new link relation type “alt-loc” containing alternate *endpoint locations* (and not resource path)

```
REQ: GET /.well-known/core

RES: 2.05 Content </sensors>;ct=40;title="Sensor Index", tt="tcp ws sms”,
</sensors/temp>;rt="temperature-c";if="sensor”,
</sensors/light>;rt="light-lux";if="sensor”,
<coap+tcp://server.example.com/>;rel="altloc”,
<coap+tcp://server.example.net/>;rel="altloc”,
<coap+ws://server.example.com/ws-endpoint/>;rel="altloc”,
<coaps+sms://12147205269/>;rel="altloc"
```

Next Steps to consider

- Still lots of open work, contributions welcome!
- Lifetime value for transport types?
- Observe relationship to detect new / expired CoAP transports?
- Is session continuity/resumption across new transports needed?
- Support alt-loc for Type T0 (single transport) nodes too? (eg sleepy node, pub/sub support, etc)
- Security considerations