Using URIs With Multiple Transport Stacks

draft-thaler-appsawg-multi-transport-uris-01

Dave Thaler <dthaler@microsoft.com>

Some Recent Requests for URI Schemes

- CoRE WG (draft-ietf-core-coap-tcp-tls-07) asked for Permanent registration of coap+tcp, coaps+tcp, coap+ws, coaps+ws (in addition to existing coap and coaps)
- Open Connectivity Foundation supported the CoRE WG request, and requested Provisional assignment if IETF declined to register them itself
- OPC Foundation asked for Permanent registration of opc.tcp, opc.amqp, and opc.wss

- Lots of debate ensued around exposing the same resource over multiple transport stacks, especially since HTTP is taking a different approach
    - This draft documents the arguments, tradeoffs, and use cases discussed so far
    - Goal is Informational RFC

The Problem…

- Lots of cases exist today where two URIs for same resource differ only in URI scheme, or authority, or path
- “Architecture of the WWW” argues for minimizing such cases since interferes with valuation and correlation of links/resources
  - But encourages use in some cases (e.g., secured vs unsecured)
- RFC 3986 (URI syntax) similarly argues for minimizing, but does not disallow
  - Indeed, ladder levels of comparison explicitly allow for it
- RFC 7595 (Scheme registration process) gives list of Requirements for Permanent Schemes, but this topic is not one of them (hence implicitly allowed)

Example Use Case

- Application layer protocol supports multiple transports (COAP, HTTP, Bluetooth?, other), and defines a transport-agnostic URI, e.g.
  - **ocf://<hash of public key>/rest/of/uri**
- But need a way to resolve actual transport endpoints
  - Some transports (e.g., websockets, HTTP, coap, …) already have URIs defined
  - For consistency, *convenient* to express them all as URIs
- Resolution might be via some lookup step, or (as in the case of OCF) learned in the same message as the app-layer URI is learned
- But the same thing can happen at multiple layers (OCF over COAP over TCP …) so general problem is not just one id/locator level split
  - OCF defined discovery one level down from ocf: URI, with no hard dependency on DNS or other servers

Discovery vs Selection

- **Discovery:** resolution of a URI to a set of potential transport endpoints
- **Selection:** process of selecting an appropriate endpoint to use from among the discovered set

- Most of the draft is about *discovery*, but also includes a section on *selection* (sorting algorithms, Happy Eyeballs style algorithms, etc.)

Discussion of 4 discovery approaches (1/2)

1. Specified by URI scheme definition, never custom. Example: tftp:
   - Avoids dependency on any other mechanism for discovery
   - No support for non-default endpoint info
   - Adding a transport later might be difficult due to hard coded assumptions
2. Encoded somewhere in a single URI
   - Avoids dependency on any other mechanism for discovery
   - Ports might be problematic:
     - Ephemeral ports (and in theory IANA ports allocated at different times) can vary by transport protocol
     - No natural place to put a transport-agnostic service name in URI
   - If complex stacks or larger or dynamic sets, problematic to try to encode into a common immutable URI

Discussion of 4 discovery approaches (2/2)

3. Use a set of URIs, one per transport stack
   - Results in multiple “equivalent” URIs so often needs a higher layer URI that acts like an ID where the set of URIs are locators
   - Still problematic if can have complex stacks with multiple layers
   - Only “natural” place is to vary by URI scheme
4. Use a locator format that might not be URI and some mechanism to learn them
   - Disadvantage may be lack of consistent syntax across transports, complicating discovery syntax

Next Steps

- AD-sponsored? Some WG? Something else?

- (Currently no plan to update RFC 7595, or requirements for permanent registration)