Naming Things

With bits of draft amsuess-t2trg-onion-coap, draft amsuess-core-coap-over-gatt and draft amsuess-t2trg-rdlink

Christian Amsüss

IETF118 Prague, T2TRG, 2023-11-03
The naming of things is a difficult matter

We have a uniform way of doing this: URIs.

URIs name resources on things. The authority component names (an aspect of) the thing. The scheme names how to reach it.\(^1\)

\(^1\)This might be controversial, and while I’m happy to have the discussion, I didn’t prepare anything for it.
What have scheme and authority ever done for us?
by example of web browser URIs

1. Tell us how to reach the service: https://example.com
2. Tell us where to reach the service: https://example.com being resolved through whatever the system’s resolver gives
3. Tell us how to verify whom to talk to: https://example.com through the browser PKI
4. Provides identity: https://example.com/page1 can be compared to an archived version

2OK it’s DNS, and maps to an IPv4 or IPv6 address)
But it’s never that simple

even in the browser

1. Tell us how to reach the service: ... but later we go h2/3
   ▶ ... but DNS resolution may provide hints for that (?)

2. Tell us where to reach the service: ... http://i2pwiki.i2p
   intentionally does not resolve

3. Tell us how to verify whom to talk to: ... https://[fc00:db8:1]
   needs extra knowledge

4. Provides identity: ... https://[fe80::1%eth0] better not be
   compared across hosts
In Constrained RESTful environments

- coap+uart://ttyUSB0 provides “how”, “where” and even some trust, but no identity. (draft bormann-t2trg-slipmux)
- coap+gatt://001122334455.ble.arpa (based on BLE MAC address) provides “how”, “where”, identity, but no trust (draft amsuess-core-coap-over-gatt)
- coap://nbswy3dpo5xxe3denbswy3dpo5xxe3de.ab.rdlink.arpa provides identity and trust, and relies on a protocol specific to .rdlink.arpa to provide a “where”, which can also provide an alternative “how”.
- The “how” is not absolutely critical – proxies don’t break trust.
Advancing the topic

Open discussion