HTTP Access Service Description Objects

Benjamin Schwartz, MASQUE @ IETF 114
Problem: Discovering complex services

How do I learn about:

- A proxy service that supports CONNECT-UDP, CONNECT-IP, and DoH?
- A DoH resolver associated with an HTTP CONNECT TCP proxy?
- An OHTTP Gateway that I can use to access any URL on this origin?
- An Oblivious DoH URI template, along with its OHTTP Gateway and KeyConfig?
- A CONNECT-UDP proxy that can also act as an OHTTP Relay?
- An OHTTP Relay with its own Gateway for multihop OHTTP?
Realization: A unified solution is possible

- **Input**: An HTTP URL or origin
- **Output**: One or more of
  - DoH URI template
  - CONNECT-UDP proxy template
  - CONNECT-IP proxy template
  - OHTTP Relay URI template
  - OHTTP Gateway URL and KeyConfig
  - ... (whatever else we want to define)

Almost every subset of these outputs makes sense for some use case!
Proposal: A simple JSON format

```json
{
    "dns": {
        "template": "https://doh.example.com/dns-query{?dns}"
    },
    "udp": {
        "template": "https://proxy.example.org/masque{?target_host,target_port}"
    },
    "ip": {
        "template": "https://proxy.example.org/masque{?target,ip_proto}"
    },
    "ohttp": {
        "relay": {
            "template": "https://proxy.example.org/ohttp{?gateway_uri}"
        }
    }
}
```
Example: Oblivious DoH

```json
{
  "dns": {
    "template": "https://doh.example.com/dns-query{?dns}"
  },
  "ohttp": {
    "gateway": {
      "uri": "https://example.com/ohttp/",
      "key": "(KeyConfig in Base64)"
    }
  }
}
```
Origin vs. URL for service identification

- Access Services are identified by the URL of an Access Service Description
  - ... unless this is not possible for the use case.
- If the service is identified by a hostname or HTTP Origin, we fetch /well-known/access-services.
Conclusion

- “We can solve any problem by introducing an extra level of indirection.”

- Enables a bunch of useful stuff
  - Richer DNS interaction while using a proxy (without assuming a trusted third-party resolver)
    - HTTP/3 bootstrap with CONNECT-UDP
    - Encrypted ClientHello, even via “legacy” proxy configuration APIs
    - Client-side DNSSEC validation
  - Advertising new access service features
    - e.g. CONNECT-UDP Listener mode
  - Changing the capabilities of an existing service without reconfiguring the clients
    - e.g. adding CONNECT-IP or OHTTP Relay support to a CONNECT-UDP proxy
  - Key-Consistency DoubleCheck (related proposal in OHAI)
  - Many other possibilities!

- Seeking adoption in MASQUE