EPP over HTTP

M. Loffredo
IIT-CNR/Registro.it

Proposal details

- *draft-loffredo-regext-epp-over-http*
- Authors:
  - M. Loffredo, L. Luconi, M. Martinelli (IIT-CNR/Registro.it)
  - J. Romanowski, M. Machnio (NASK/.pl Registry)
- First submission: March 2022
- Current version: -02

Why EPP over HTTP? (1)

- HTTP is loosely coupled with the network
- HTTP provides client-server cross-platform technology communication
- HTTP simplicity reduces the development time
- HTTP offers standardized solutions to ensure security

Why EPP over HTTP? (2)

- The speed gap between HTTP and TCP is actually not so large as in the past
- Load balancing can be more easily implemented at L7 than at L4
- Migrating an HTTP server to cloud takes less effort than for a TCP server

Message Exchange

- EPP commands semantics are preserved
  - No EPP message is altered
  - Clients issue the EPP commands via HTTP POST
  - Servers return the EPP responses in the HTTP response body
  - No other part of HTTP request and response is used to deliver EPP data

Session Handling (1)

- An EPP session is mapped onto an HTTP session by using a Cookie (RFC 6265):

  - A server receiving a `<login>` command sends the session ID to the client through the `"Set-Cookie"` response header

```
== Server -> Client ==

Set-Cookie: SID=52ceb07c2a824f09a1c6f9c45574097d
```

  - The client includes the cookie in the subsequent requests of that EPP session

```
== Client -> Server ==

Cookie: SID=52ceb07c2a824f09a1c6f9c45574097d
```

- The name of the cookie attribute identifying the session ID is not relevant

Session Handling (2)

- An EPP session is ended by the client through the `<logout>` command
- A server receiving a `<logout>` command ends the EPP session by invalidating the HTTP session after having issued the response
- EPP sessions may be ended by the server due to timeout

<hello> Command

- The client may issue the <hello> command outside an EPP session
- The server returns the <greeting> response without starting a session by sending:
  - no cookie
  - an expired cookie
- Clients may also issue the <hello> command within an EPP session

Return Codes

- HTTP error codes are used for signaling HTTP requests failure
- EPP error codes are used for signaling EPP commands failure
  - The HTTP code `200` is used for both successful and unsuccessful EPP commands

Mapping Considerations

- RFC 5730 includes considerations to be addressed by mappings over transport (L4) protocols
- HTTP is a high level (L7) protocol largely used by REST APIs as a pseudotransport
  - Interpreted RFC 5730 in line with the common practice by HTTP-based applications of using sessions to store authentication information (see *draft-ietf-regext-rdap-openid*)
  - EPP sessions are one-to-one mapped onto HTTP sessions
    - An EPP session lives as long as the related HTTP session persists
    - An HTTP session starts with an EPP `<login>` request
    - An HTTP session ends with an EPP `<logout>` request or due to timeout

Security Considerations (1)

- HTTPS (RFC 8740) must be used to protect the transit of sensitive information
  - ➢ Support of TLS 1.2 (RFC 8446, RFC 9155) or higher is required
- Servers should:
  - implement additional measures to validate clients by:
    - IP whitelisting;
    - locking the session ID to the client's IP address;
    - requiring clients to present a valid X.509 certificate issued by a CA.

Security Considerations (2)

- Servers should:
  - generate at least 128 bit long session IDs to prevent them from being hijacked;
  - control the rate of both EPP sessions and HTTP connections to reduce the resource consumption.
- Servers may:
  - limit the lifetime of active sessions;
  - control cookies usage by setting other attributes (e.g. `"Path"`, `"Max-Age"`).

Thanks for the attention

Q&A at Panel Disussion