XRootD + Token Tests

Some notes from the last week of testing…

XRootD + HTTP protocol update

- 5.5.0-rc1 contains an essential patch to handle storage.create authorizations correctly.
  - Particularly, you can upload or rename files as long as the *destination* doesn't exist – e.g., data is not deleted.
- Unfortunately, there is still one "incorrect" status code. When you try to "overwrite" a file with storage.create, XRootD (correctly) fails the operation.
  - RC1 returned a 500 (Internal error). That's obviously wrong!
  - Currently, the branch returns a 409 (Conflict) which is akin to an EEXIST, the POSIX error code you'd get in this situation.
  - PR #1763 changes this to a 403 (Forbidden), matching other implementations.
- All fixed by 5.5.0 final!

XRootD + xrootd protocol + tokens

- The xrootd protocol has a **strong** concept of a session which is lacking in HTTP.
  - Historically, the xrootd session was authenticated which resulted in certain authorizations on paths.
  - Contrast this to HTTP where each request is separately authorized via headers.
    - The ~outlier being how WLCG uses HTTPS + client X.509 where the session is used.
- In XRootD 5.0.0, the implementation made it easier to authorize individual requests for the xrootd protocol.
  - This was available earlier but internal data structures made this awkward.

XRootD + xrootd protocol + tokens

- <u>However</u>, even if you authorize each request, do you need to authenticate the session?
  - HTTP protocol: **NO!** No privileges just from establishing a session (a TCP socket or TLS connection).
  - xrootd protocol: **YES!** At the protocol level, some reasonably-expensive requests are available to anyone who establishes a session.
    - N.b.: Unclear to Brian if this is really true at the protocol level or deeply embedded into the available implementations.
- Hence, the "ZTN" authentication protocol was born!
  - To establish a session, (a) a TLS connection must be established and (b) the **client must send a bearer token**.

XRootD + xrootd protocol + tokens

- With the built-in ZTN authentication method,
  - XRootD client libraries automatically Does The Right Thing. No changes to the CLI or API calls.
  - URLs don’t need to be changed to include the token.
  - No worry about accidental logging of URLs resulting in tokens leaking to logfiles.
  - Handles redirection correctly.

XRootD + xrootd protocol + tokens

- If you established a valid session with token X, what does this authorize?
  - After lengthy discussion, we decided to treat this as a **session token**:
    - All requests are evaluated AS IF the token X was presented with the request.
    - If token Y is presented with the request, then it takes precedence over X.
- Nitty-gritty – to enable the ZTN protocol in your XRootD configuration, set:

```
sec.protocol ztn
```

- (Defaults to using existing SciTokens configuration)

Existing Problems

- What happens when the session token expires?
  - Current XRootD implementation: token is re-evaluated on each access, all requests on that session are not authorized.
  - **PROPOSAL**: Authorization for the session is evaluated when the session starts, lasts for the entire session.
- What happens when token expires and XRootD client reconnects?
  - For each new connection, the WLCG bearer token discovery protocol is run.
    - The BEARER_TOKEN env var is static throughout the process's lifetime. No chance for updates.
    - The BEARER_TOKEN_FILE is re-read for each new connection. Updated token is used if the file is overwritten.
  - Conclusion: Always use **BEARER_TOKEN_FILE**.