Compression Dictionary Transport

Overview

- Use previous responses as compression dictionaries for future requests
- Client-driven, negotiated

Examples

- Previous version of JS lib as dictionary for updated version
- Custom dictionary for HTML pages with common template content
Advertise Dictionary - Use-As-Dictionary

- **Use-As-Dictionary** response header (sf-dictionary)
- **Params:**
  - **match**: URL path (same-origin) e.g. `/app/*/main.js`
  - **ttl**: Time To Live (seconds - optional)
  - **hashes**: List of supported hash algorithms (optional)
Use Dictionary - **Sec-Available-Dictionary**

**Client**

- Selects “best” available dictionary for request
- Adds “Sec-Available-Dictionary: <hash>”
- Adds supported dictionary content-encodings to “Accept-Encoding:”
  - “br-d” - Brotli with Dictionary
  - “zstd-d” - Zstandard with Dictionary

**Server** (if hash is known dictionary and content-encoding supported)

- Serves dictionary-compressed response
- Sets “Content-Encoding” to “br-d” or “zstd-d” to match algorithm used
- Adds “Vary: accept-encoding, sec-available-dictionary” (always)
Privacy

- Client-managed
- Partitioned with Cache and Cookies
- Cleared with Cache and Cookies
Security - Oracle attacks

- Opaque dictionary and payload expected to be revealable
- Only use dictionary compression for non-opaque requests
- *Mostly* achievable on-client outside of protocol:
  - If request is known to be opaque, omit `sec-available-dictionary`
Server MUST NOT use dictionary compression when:

- **Sec-Fetch-Mode: cors** AND
- **Origin != Access-Control-Allow-Origin** AND
- **NOT Access-Control-Allow-Origin: ***

**Red:** Request header  
**Blue:** Response header
Thank You

pmeenan@google.com

yoavweiss@google.com