Part 1: HTTP Mutual auth

Yutaka OIWA

7 November 2012

HTTPAUTH, IETF 85

What's this?

- Highly-secure HTTP authentication:
  - Based on "weak shared secret" (i.e. passwords or temporary shared tokens)
  - Mutually-authenticating (server-to-client / client-to-server simultaneously)
  - All exchanges cryptographically encrypted
  - Encrypted server-side stored secret

Why we need weak secrets (e.g. passwords/passphrases) auth?

1. passwords are the simplest, the easiest for novice users
2. SSO/federation needs an initial trust point
3. strong secret-based authentications (including public and shared keys) need facilities for boot-strapping, transferring and/or recovering secrets

How it works?

- Designed on top of HTTPbis-P7
- Fully HTTP/1.1 compliant
  - Per-message semantics
    + internal explicit session ID
  - Not relying on “implicit” sessions (connections)
  - Will work on HTTP/2.0, too (hopefully)
- Efficient
  - Initial authentication: 2 RTs (+ initial plain 401 RT)
  - 1 RT for re-authentication
    - (Strong) keys cached for re-authentication
    - path hints provided for removing plain 401 RTs

Crypto (non-)choices

- Not fixed to specific crypto
  - Any ZKPPs/augumented PAKEs should work
  - Generic framework for mutual, multi-hop authentication on HTTP
- Open for any proposals in WG/BOF/IDs
  - HTTPAUTH WG or some other should decide
- One “sample” on separate draft
    - For trial implementation and inter-op tests.

Additional features

- Host-based SSO
  - Hosts on any same domain can use same passwords securely and privide SSO
    - The concept provided partially in RFC2617 Digest but never implemented
      - Prone for offline brute-force password discoveries
    - Now secure because passwords strongly protected
- Optional Auth / auth control
  - Now became a separate, generic features
    - Continues to next part

Ways to go

- More features?
  - Or, current simplicity better?
- Share the framework for *multi-hop message-based authentication* with other proposals?

Part 2: HTTP auth extensions

Yutaka OIWA

7 November 2012

HTTPAUTH, IETF 85

What's this?

- Generic extension of httpbis-p7 for interactive clients (i.e. browsers)
  - Applicable for any http-p7-based authentications
  - Provides ways for precisely controlling user experience of authentications

Why needed? (1)

- Current HTTP auth lacks required features for many Web applications, providing a crappy UI experiences
  - Authentication is always mandatory
    - If server sends 401, clients have to always ask users to authenticate themselves
    - No easy support for guest users on portal sites
  - Logout is non-standard
  - (More reasons...)
- Services tends to use Form authentications for detailed control of UI experiences

Why needed? (2)

- However, HTML-layer authentication have severe security implications
  - Fake UIs, script injections, …
- Currently, HTTP auth is hard to deploy
  - Service providers says, “we cannot do it because we cannot implement bla-bla-bla…”
- So, we want to say, at least, “we can do anything you want to do with Form auth!”
  - Isn’t it cool? ☺

Relation to existing specs/software

- Does not change natures of httpbis-p7
  - Still fully conforming to httpbis-p*
- Backward compatible
  - Will not confuse existing clients
  - Safely ignored by existing clients
  - Fallback to Form authentication possible
    - Provides a transferring paths to new HTTP auths

Features (1)

- Non-mandatory (optional) authentication
  - = “Guest user” support
  - Servers to serve a “200”-status document, while allowing clients to start authentication
    - If clients knows a credential (re-authentication situation), clients should retry with it
  - “`Optional-WWW-Authenticate:`” header
    - Equivalent to httpbis “200 + WWW-Authenticate” implication, but explicit and legacy-friendly

Features (2)

- Additional control for auth behaviors
  - "`Authentication-Control:`" header
  - When/how to start authentication
    - Redirect URI for *non-authenticated* clients
      - Compatibility to current "log-in" pages
    - No new authentication on this page
  - When/how to terminate authentication sessions
    - Timed log-out
    - Immediate, server-initiated log-out
    - Redirect URI for user-initiated log-out

Ways to go

- More features?
  - Required?
- Be a foundation for next-gen HTTP auths?
- Whether to use httpbis “new implications”?
  - About WWW-authenticate on non-401 msgs
  - My opinion: clearer, but compatibility problem will arise