Authn Agililty with HTTP-SASL

Rick van Rein
rick+ietf@openfortress.nl

Most Protocols wrap SASL Authn

- Choice of Authentication Mechanisms
  - Choice of cryptographic strength
  - Choice of extra cryptographic properties
  - Flexible towards new cryptographic developments
  - Shared knowledge with most other protocols
- HTTP would also benefit from SASL

Present HTTP Authn is an Island

- Separate security standards causes a split
  - Credentials not usually shared among protocols
  - Low cryptographic agility (Kerberos, OPAQUE)
  - Channel Binding is barred for every use case
  - Remote trust is solved within the HTTP island

HTTP and the URI auth-part

- Authentication, Authorisation, Authority
  - URI’s define an *authority*-part
  - Basic Auth hacks treat it as a *authentication*-part
- URI user represents a resource
  - URI → server-side user ; Authn → client-side user
- Add a HTTP `User:` header, akin to Host:

Cryptographic Agility

- Cryptographers want to enable new algorithms
  - Quantum Relief, Channel Binding, Key Derivation
  - Designs need to pass tokens back and forth
  - HTTP imposes a large (and important) barrier
- HTTP-SASL adds cryptographic agility
  - IRTF OPAQUE; W3C sovereign identity

Example: HTTP-SASL

draft-vanrein-httpauth-sasl

```
WWW-Authenticate: SASL
   realm="members only"
   mech="GS2-KRB5-PLUS SCRAM-SHA-256-PLUS OPAQUE",
   s2s="[xxxxx]"
```

```
Authorization: SASL
   realm="members only"
   mech="SCRAM-SHA-256-PLUS",
   c2s="[n,,n=user,r=r0pr...q0]",
   s2s="[xxxxx]"
```

[base64]

Prior Attempt of HTTP-SASL

- Predates HTTP Authentication Framework
  - Raised issues with server-side state
  - HTTP specs must not *depend on* server-side state
- State travels server → client → server in `s2s=`
  - Protected with symkey encryption/signature
  - To avoid replay attacks, use TLS encryption

Realm Crossover for HTTP-SASL

```
   +---------+    SASL     +---------+    SASL    +----------+
   |  Web    |-----------> |  Web    | ---------> | Identity|
   | Browser|------------> | Server  | ---------> |  Domain  |
   +---------+    HTTP     +---------+  Diameter  +----------+
       ||                      ||                     ||
john@example.com         find SRV, TLSA           example.com
  & credential            & relay SASL         authentication


                Realm Crossover authentication:

          Client John authenticates to his own Domain
                while using a foreign Web Server.
```

Questions?

- HTTP-SASL built for Apache, Nginx, Firefox
- EU likes this direction (NGI Pointer)

- *Extra slides: blog, specs and code*

Blog, Documentation

- `http://internetwide.org/tag/identity.html`
- `http://common.arpa2.net/`
- `http://quick-sasl.arpa2.net/group__quickdiasasl.html`

Draft Specifications

- `draft-vanrein-httpauth-sasl`
- `draft-vanrein-internetwide-realm-crossover`
- `draft-vanrein-diameter-sasl`

Code for HTTP-SASL

- https://gitlab.com/arpa2/apachemod/-/tree/master/arpa2_sasl
- https://gitlab.com/arpa2/apachemod/-/tree/master/arpa2_diasasl
- https://github.com/stef/ngx_http_auth_sasl_module

Code for [[Quick-]Dia]SASL

- https://gitlab.com/arpa2/Quick-SASL
- https://gitlab.com/arpa2/Quick-SASL/-/blob/master/include/arpa2/quick-diasasl.h
- https://gitlab.com/arpa2/quick-der/-/blob/master/arpa2/Quick-DiaSASL.asn1
- https://gitlab.com/arpa2/freediameter-sasl