# HTTP Auth Wants/Needs

- My customers want/need:
  - Their choice of authentication mechanism
  - Privilege separation, particularly on the server side
  - Ease of/widespread implementation
  - HTTP router support: server that terminates HTTP(S) connection may forward request to another server
- TLS is here to stay, of course

# Implications

- Mechanisms with more than one pair of messages
  - Nor like DIGEST-MD5, with first msg being a challenge in a 404
- Can't have just one mechanism

# OK, how?

- Out-of-band: bridge authen. mechs:
  - Kerberos → whatever bearer token scheme
  - Kerberos → user certs (SACRED, browserid)
  - …
- Use mech in-band:
  - Like DIGEST-MD5 or Basic
  - Need to map stateful authen. exchanges onto stateless HTTP
- My proposal: in-band, “over” HTTP

# RESTauth

- A pattern for authentication
- Authen. msgs POSTed, first to login URI, then to session URI
- Don't specify use of SASL, GSS, EAP
- Binding requests to sessions:
  - Cookies, or cookie-like are a weak option
  - MAC with shared session key (associated with session URI) of TLS channel binding (i.e., the server cert.)

# RESTauth and the web

- JS XHR extensions / utilities
- HTML element
- Credential manager in browser chrome
    - Script/page element, or HTTP 401 trigger credential manager UI invocation (UI spoofing is a separate problem)
- Scripts/pages must not get access to raw credentials, or even list user IDs

# That's it.

- It may well be that bridging is the only thing that will see the light of day.  I'm OK with that, though it's not my first preference.