SASL GSS-API Bridge: GS2

- http://josefsson.org/sasl-gs2-*/
- Open Issues
  - IAB concern about excessive layering (20%)
  - Channel bindings relationship (60%)
  - Defining channel bindings for GS2 under TLS (20%)
- If you are implementing this now, please contact simon@josefsson.org if you are interested in interop testing.

draft-iab-auth-mech 1/2

- http://www.ietf.org/internet-drafts/draft-iab-auth-mech-05.txt
- Section 11.3:

> Many of the legacy authentication mechanisms that users and administrators wish to support are themselves generic frameworks of one kind or another. For instance, SASL allows the use of GSSAPI, which itself is a generic framework for a number of mechanisms. This sort of layering dramatically increases both implementation and deployment complexity. For instance, GSSAPI contains mechanisms that are essentially equivalent to Kerberos, but SASL also supports Kerberos directly. Under what conditions should clients use Kerberos directly and under which should they use it through GSSAPI?
>
> In accordance with the principle of having as few mechanisms as possible, frameworks should avoid mechanisms that are themselves frameworks, in favor of using the second framework's mechanisms directly.
>
> "We'll build ours on top of theirs" is a bad policy.

draft-iab-auth-mech 2/2

- Q: Does anyone have concrete plans to use non-krb5 mechanisms?
  - Worried about absent review from non-KerberosV5 communities.
- Solutions:
  1. Go ahead. If there is a problem with non-KerberosV5 mechanisms, solve them later on. May lead to GS3-* which imply even worse interop problems than between GS2-* and GSSAPI. Potential disaster unless someone has a good idea on how to avoid it.
  2. Drop support for non-KerberosV5 mechanisms, i.e., rename this to GS2-KRB5. Non-KerberosV5 mechanism can use this protocol as a foundation to define a new mechanism.
     - If we need to support e.g. SPKM too, include it in this spec too, as GS2-SPKM.

Channel binding relationship 1/2

- Background:
  - “Sam points out that in SASL what we want out of channel binindings is to affect the negotiation of security layers. If a channel being bound to provides integrity and confidentiality protection then the application won't want to use a SASL mechanism's own security layers -- it's a waste.”
- draft-ietf-sasl-gs2-00 can transfer non-GSS-API “channel binding” data in a separate field outside of the GSS Context/Wrap tokens.
- Problem: how do GSS-API cb interact with GS2 cb?
- Please chose between choices in next slide... (or come up with more options)

1. **There is just one channel binding concept. The application will have to put exactly the same data into "channel_binding_data" and "chan_binding".**
   - This will lead to a security consideration that the GS2 mechanism may expose the channel binding in the clear.
2. **There are two channel bindings concepts, and they are orthogonal. Applications may set "chan_binding" (GSS-API cb) on a per-application need, and the "channel_binding_data" (GS2 cb) will have to be defined for, e.g., TLS.**
   - Having "chan_binding" be application-specified seem problematic to me. There is little interoperability in that. However, it may solve some problems.
3. **Same as 2 but the "chan_binding" field MUST be NULL. In other words, for GS2 interop we disallow the GSS-API cb completely, and only support a GS2 cb.**
4. **There are two channel bindings concepts, but they are related. The GS2 cb data field is included within the GSS-API cb field.**
   - This would permit the most flexibility while continuing to use the GSS-API cb fields. It has the same interop problems as 2.

- **4) There are two channel bindings concepts, but they are related. The GSS-API cb is included within the GS2 cb.**
  - This is similar to 3, but has some different security properties. It seems this may end up including the initiator/acceptor addresses in the data protected by the Wrap token.

Channel bindings for GS2 under TLS

- Offline discussions with Nico to use the (expired) draft-ietf-kitten-gssapi-channel-bindings-01.txt
- Technical change: Don't use finished messages, use the TLS PRF to derive specific data used in GS2.
  - Reason: GnuTLS applications cannot easily access the unencrypted finished messages (they just see the encrypted message).
  - This approach seem generally safer in the future given various TLS extensions (TLS/IA) that may modify semantic of a finished message.