draft-ietf-oncrpc-rpcsec_gss-01.txt

Summary of discussion and issues

Mike Eisler, Document Editor

mre@Eng.Sun.Com

OVERVIEW OF DISCUSSION AND ISSUES

- Negotiation of mechanism - unresolved
- Specification of QOP/Service values - unresolved
- Denial of service of attacks using sequence numbers - unresolved
- Clarify integrity check wording - resolved
- RPCSEC_GSS vs. GSS errors - resolved
- Generation of session handles - resolved
- Version negotiation - unresolved
- gss_get_mic() vs. gss_wrap() - unresolved

- section 6.2 - flooding attacks - resolved
- section 5.2.1 - context creation messages - unresolved
- data type of gss_proc - resolved
- GSS-API V1 references vs. GSS-API V2 - unresolved

NEGOTIATION OF MECHANISM

- John Linn: “no facility at the rpcsec_gss layer to transport or negotiate GSS-API mechanism identifiers, mechanism selection must either (a) be static, (b) be performed out-of-band, or (c) be negotiated within the GSS layer”
- Anonymous: “... ONC RPC model doesn't stretch very well to include models such as negotiated QOS or authentication flavor.... if you don't add such improvements to ONC RPC, there's little attracting me to ONC over the alternatives.”
- *Are the above issues serious enough to hold back the ID?*

SPECIFICATION OF QOP/SERVICE VALUES

- John: “is it expected that any such QOP values will be sourced within the RPC layer, or instead that the relevant GSS mechanism ID will be reflected up to the RPC caller so that it's equipped to undertake the responsibility of selecting QOPs suitable to the prevailing mechanism?”
- *The API is expected to provide mechanism selection input/output to the client and server. The ID can be updated to reflect this.*
- John: “Use of “default” would be simplifying here, and is probably to be preferred where/if possible”

- *Proposal: When the client doesn’t know what QOP to use, “default” should be specified in the ID*
- Marc Horowitz: Method of imposing QOP/service has problems:
  - “client has to “guess” in rpc_gss_init_arg”
  - “field is not protected in any way”
  - proposes: “including the required qop and service values in the final (GSS_S_COMPLETE) rpc_gss_init_res, and using gssapi to protect this information.”
- *The intent of feature in the ID was that clients would pick the QOP from an out-of-band name service. The server (e.g. NFS) may not know what the right QOP is until the client accesses a specific resource.*

- ***Is including the required QOP in the response redundant? If GSS-API is used to protect the data, then the QOP is encoded in the results of the GSS-API operation.***
- **Marc: "why have the service/QOP" in the protocol?**
- **Marc: "it seems more and more to me like dropping the QOP/service stuff from the init phase is the right answer."**
- ***It may be possible to remove this from the protocol and still achieve the desired effect. However, the next discussion point from Barry Jaspan raises an issue, leaving this still a point of contention.***

DENIAL OF SERVICE OF ATTACKS USING SEQUENCE NUMBERS

- **Barry Jaspan: suggests "the seq_window also be protected when communicated to the client"**
- ***This is reasonable. However, if we delete QOP/ integrity negotiation from the protocol, what QOP and what service does the server use to protect seq_window?***
- ***Proposal: leave QOP/service in the protocol, and protect the service/seq_window when communicated back to the client with GSS-API***

CLARIFY INTEGRITY CHECK WORDING

- **John: description of Context Management (sec. 5.3.3.1) should emphasize: "that the integrity check on an incoming message is to be validated before adjusting the receive window in response to the incoming message's sequence number"**
- ***Agreed, though if the sequence number is below the window, the request can be dropped without the integrity check.***

RPCSEC_GSS VS. GSS ERRORS

- **John: RPCSEC_GSS_NOCRED and RPCSEC_GSS_FAILED are similar in name to GSS-GSS-API major statuses GSS_NO_CRED and GSS_FAILURE, but apparently different in meaning. Should the RPCSEC_GSS_* codes be renamed?**
- ***They will be renamed since they definitely mean different things.***
- **John: "is it possible and useful to define the mapping between specific GSS-level major status codes and the corresponding RPC layer error?"**
- ***An attempt will be made to define these mappings.***

GENERATION OF SESSION HANDLES

- **Marc: Regarding sec. 5.2.2.1, it states: The server must generate handles such that they will be generated again for the same pair of client and server principals.**
  - "What if there are two simultaneous connections from the same client to the same server?"
  - "GSSAPI requires that mechanisms protect context setup against replay attacks"
- ***There should have been a "not" between "will" and "be".***
- ***Proposal: for simplicity, delete the offending sentences from section 5.2.2.1.***

VERSION NEGOTIATION

- **Marc: "The version negotiation procedure seems unnecessarily complex."**
- **Marc: "In the case when the client and server both support the same protocol version (which will be most of the time), it would be useful to be able to do an aggressive setup, where instead of asking for a protocol version, the first message for that version is sent."**
- ***Agreed. Since the server-side of version negotiation must be stateless, the server doesn't care.***
- ***Proposal: Since version negotiation is contentious, and since we at only verison1, delete version***

negotiation from the specification, reserve version 0, and if versions are added in the future, re-visit version negotiation then.

GSS_GET_MIC() VS. GSS_WRAP()

- **Marc: The protocol in the ID uses “gss_get_mic() for integrity, and gss_wrap() for encryption. it would be simpler... to use gss_wrap in both cases, with the conf_req flag set accordingly”**
- ***This will introduce an unnecessary byte copy that gss_wrap will incur when conf_req is zero.***

- **Marc: Perhaps both modes should be allowed, as there is talk about extending GSS-API to use an application buffer.**
- ***This extension won’t eliminate the byte copy. Have two ways to do integrity is protocol bloat and by consequence, ONCRPC-API bloat. Why would an ONC-RPC programmer pick a “wrapped-integrity” service if the documentation warned that this consumed a byte copy?***

SECTION 6.2 - FLOODING ATTACKS

- **Marc: “Section 6.2 goes through some effort to demonstrate that there are no flooding attacks possible. There is a trivial attack where the attacker sends fake requests above the window. These will not be rejected due to the sequence number (since it must increase), forcing the server to validate the header checksum and fail.”**
- ***The ID does, albeit not clearly, acknowledge this.***
- ***Proposal: Clarify section 6.2.***

SECTION 5.2.1 - CONTEXT CREATION MESSAGES

- **Marc: Section 5.2.1 says: The first RPC request from the client to the server initiates context creation for those mechanisms that require context creation messages. “All mechanisms will generate at least one token requiring a context creation message”**
- ***The GSS-API v2 specification (draft-ietf-cat-gssv2-08.txt) seems to imply that an initial call to GSS_Init_sec_context can return GSS_S_COMPLETE. Is there a problem with calling GSS_Accept_sec_context() on a token created with a GSS_Init_sec_context call returning GSS_S_COMPLETE?***

DATA TYPE OF GSS_PROC

- **Marc: “gss_proc is an unsigned int instead of an enum. Why?”**
- ***Proposal: redefine gss_proc as an enumerated type.***

GSS-API V1 REFERENCES VS. GSS-API V2

- **Marc: "the draft should use [GSS-API] v2 function names, not v1 function names"**
- ***It wasn't clear if referencing another ID was appropriate.***
- ***Proposal: change the names to use GSS-API V2 function names. Change them back to V1 if the GSS-API V2 is not an RFC by the time the rpcsec_gss ID is published as its own RFC.***