NETWORK WORKING GROUP
Internet-Draft
Expires: April 17, 2006

N. Williams
Sun

October 14, 2005

GSS-API Naming Extensions
draft-ietf-kitten-gssapi-naming-exts-01.txt

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on April 17, 2006.

Copyright Notice

Copyright (C) The Internet Society (2005).

Abstract

The Generic Security Services API (GSS-API) provides a simple naming architecture that supports name-based authorization. This document introduces new APIs that extend the GSS-API naming and authorization model.

Table of Contents

1. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Introduction

As described in [I-D.GSS-NAMING] the GSS-API's naming architecture suffers from certain limitations. This document proposes concrete GSS-API extensions as outlined in [I-D.GSS-NAMING].

A number of extensions to the GSS-API [RFC2743] and its C Bindings [RFC2744] are described herein with the goal of making authorization information, and other information that can be modelled as "name attributes" available as such to applications. For example, Kerberos V authorization data elements, both, in their raw forms as well as mapped to more useful value types, can be made available to GSS-API applications through these interfaces.

The model is that GSS names have attributes. The attributes of a name may be authenticated by the credential whence the name comes, or may have been set locally on a GSS name for the purpose of "asserting" the attribute during credential acquisition or security context exchange. Name attributes' values are network representations thereof (e.g., the actual value octets of the contents of an X.509 certificate extension, for example) and are intended to be useful for constructing portable access control facilities. Applications may often require language- or platform-specific data types, rather than network representations of name attributes, so a function is provided to obtain objects of such types associated with names and name attributes.

3. Name Attribute Sources and Criticality

A given GSS name object's name attributes may be authenticated or asserted by an associated credential, or it may be mapped or derived from another attribute of the same name.

That a given name's given attribute is 'mapped' means that it was obtained through some mapping mechanism applied to another attribute of the name that was not, itself, mapped. For example, such attributes as platform-specific internal identifiers may sometimes be mapped from other name attributes.

Name attributes may be "critical," meaning that applications that do

not understand them MUST reject security contexts where the peer has such unknown, critical attributes.

4. Name Attributes/Values as ACL Subjects

Some name attributes (e.g., numeric user or group identifiers) may be useful as subjects of access control list (ACL) entries, some may not (e.g., time of day login restrictions). The GSS_Inquire_name_attribute() function indicates this.

To facilitate the development of portable applications that make use of name attributes to construct and evaluate portable ACLs the GSS-API makes name attribute values available in canonical network encodings thereof.

To facilitate the development of platform- or language-specific applications that need access to native types of representations of name attributes an optional facility is provided, GSS_Map_name_to_any().

5. Mapping Mechanism Facilities to Name Attributes

[NOTE: This entire section should probably be split into one or more separate Internet-Drafts. It is here in the -00 of this I-D to help readers understand how to mechanism-specific name attributes would be accessed through these GSS-API extensions.]

Kerberos V [I-D.ietf-krb-wg-kerberos-clarifications] and the Simple Public-Key GSS-API Mechanism, SPKM [RFC2025], both support the concept and encoding of containers of "authorization-data" as described in [I-D.ietf-krb-wg-kerberos-clarifications].

PKIX [RFC3280] supports a number of authorization-data-like features, like Extended Key Usage values (EKUs) and certificate extensions.

The authorization data can be accessed through the GSS-API name attributes facility defined herein.

5.1. Kerberos V and SPKM Authorization-Data

Authorization-data non-container elements asserted in Kerberos V AP-REQ Authenticators MUST be mapped into *asserted* GSS-API name attributes; if not contained in AD-IF-RELEVANT then they MUST be mapped into *critical* GSS-API name attributes. AD-AND-OR authorization-data elements MUST be mapped into a single *critical* attribute, (TBD).

Authorization-data included in Kerberos V Tickets that is not contained in AD-KDCIssued (with valid signature) MUST be mapped into *asserted* GSS-API name attributes. Conversely, authorization-data elements in Kerberos V Tickets contained by AD-KDCIssued MUST be mapped into *authenticated* GSS-API name attributes

As with authorization-data elements in Authenticators, authorization-data elements in Tickets not contained in AD-IF-RELEVANT are to be mapped to *critical* name attributes, and similarly with AD-AND-OR (see above).

The OIDs for authorization-data elements are to be the authorization-data element's 'ad-type' integer ID, relative to the base OID <TBD> [NOTE: what about negative ad-type's? OID arcs are positive integers... ad-type is an Int32, so clearly something can be done.]

5.2. Kerberos V Cross-Realm Transit Paths

[Add text on how to represent/encode/interpret krb5 realm transit paths as name attribute values. And text on PKINIT too... Basically Ticket's 'transited' field should be exposed as an authenticated name attribute, with some uncompressed encoding, possibly encompassing certificate validation paths of client certs used for PKINIT, with criticality determined by the presence of the transit-policy-checked flag.]

5.3. PKIX Certificate Extensions

[NOTE: In the Kerberos V authorization-data case we can tell when AD elements are "authenticated" and when the are asserted, but what about x.509 certificate extensions? Clearly KU, EKUs and subjectAltNames are authenticated in that no CA should sign a cert with, say, arbitrary subjectAltNames not understood by the CA, but, does that also apply to all other x.509 certificate extensions? The answer may depend on actual CA operator practices... At worst a new extension may be needed, like Kerberos V's AD-KDCIssued AD container element; at best this text can just say "all cert extensions MUST be mapped to authenticated..." below.]

PKI certificate extensions MAY/SHOULD/MUST (see comment above) be mapped to *authenticated* GSS-API name attributes with the _same_ OIDs, and if they be marked critical in the certificate then they MUST be mapped as *critical* GSS-API name attributes. SubjectAltNames and EKUs, specifically, MUST be mapped to *authenticated* GSS-API name attributes; see below. Certificate extensions MUST be mapped to GSS-API name attributes whose OIDs are the same as the extensions'

5.3.1. PKIX EKUs

Extended Key Usage extensions, specifically, MUST be mapped as described above, except that GSS-API name attributes for EKUs MUST have NULL values (i.e., zero-length OCTET STRINGs).

PKI certificate key usages (KUs, but not EKUs), MUST NOT be mapped to GSS-API name attributes.

5.3.2. PKIX Certificate Alternative Names

PKI certificate subjectAltNames MUST be mapped as *authenticated*, *non-critical* GSS-API name attributes.

PKI certificate extensions MUST be mapped to *authenticated* GSS-API name attributes with the _same_ OIDs, and if they be marked critical in the certificate then they MUST be mapped as *critical* GSS-API name attributes.

Extended Key Usage extensions, specifically, MUST be mapped as described above, except that GSS-API name attributes for EKUs MUST have NULL values (i.e., zero-length OCTET STRINGs).

5.3.3. Other PKIX Certificate Extensions and Attributes

[Add text...]

5.4. PKIX Certificate CA Paths and Trust Anchors

[Add text on how to represent/encode/interpret PKI certificate validation CA paths as name attribute values, much as with Kerberos V transited paths.]

6. GSS_Inquire_name_attribute()

[NOTE: This function was somewhat controversial at IETF63; we should decide whether to remove it at IETF64. The controversy was, as I recall over whether reflection functionality might not be dangerous, leading to construction of inappropriate ACLs through dumb UIs. For now I am making some changes to it: adding a NAME object as an input parameter and some output parameters.]

Inputs:

- name NAME

- attr OBJECT IDENTIFIER

Outputs:

- major_status INTEGER,
- minor_status INTEGER,
- attr_name OCTET STRING, -- display name of the attribute
- attr_description OCTET STRING, -- description of the attribute
- attr_values_ordered BOOLEAN, -- whether the attribute's values are an ordered set
- attr_is_a_name BOOLEAN, -- whether the attribute's values can be used as subjects of access control list entries
- attr_is_trust_indicator BOOLEAN -- whether the attribute's values represent nodes in trust paths

Return major_status codes:

- GSS_S_COMPLETE indicates no error.
- GSS_S_UNAVAILABLE indicates that the given attribute OID is not known (even if present as a name's attribute).
- GSS_S_FAILURE indicates a general error.

This function outputs a name for the given name attribute, description for display to users, and indicates whether the attribute's values are ordered sets, whether the given name attribute's values are useful as the subject of an access control list entry and/or whether the given name attribute's values are useful as indicators of trust (for example, whether they name PKIX trust anchors).

6.1. C-Bindings

```
OM_uint32 gss_inquire_name_attribute(
  OM_uint32                    *minor_status,
  gss_name_t                   name,
  gss_OID                      attr,
  gss_buffer_t                 attr_name,
  gss_buffer_t                 attr_description,
  int                          attr_values_ordered,
```

```
  int                          *attr_is_a_name,
  int                          *attr_is_trust_indicator
);
```

7. GSS_Display_name_ext()

Inputs:

- name NAME,
- display_as_name_type OBJECT IDENTIFIER

Outputs:

- major_status INTEGER,
- minor_status INTEGER,
- display_name STRING

Return major_status codes:

- GSS_S_COMPLETE indicates no error.
- GSS_S_UNAVAILABLE indicates that the given name could not be displayed using the syntax of the given name type.
- GSS_S_FAILURE indicates a general error.

This function displays a given name using the given name syntax, if possible. This operation may require mapping MNs to generic name syntaxes or generic name syntaxes to mechanism-specific name syntaxes; such mappings may not always be feasible and MAY be inexact or lossy.

7.1. C-Bindings

```
OM_uint32 GSS_Display_name_ext(
  OM_uint32                    *minor_status,
  gss_name_t                   name,
  gss_OID                      display_as_name_type,
  gss_buffer_t                 display_name
);
```

8. GSS_Inquire_name()

Inputs:

- name NAME

Outputs:

- major_status INTEGER,
- minor_status INTEGER,
- name_is_MN BOOLEAN,
- mn_mech OBJECT IDENTIFIER,
- asserted_attrs SET OF OBJECT IDENTIFIER,
- authenticated_attrs SET OF OBJECT IDENTIFIER,
- critical_attrs SET OF OBJECT IDENTIFIER,
- all_attrs SET OF OBJECT IDENTIFIER,
- [NOTE: Perhaps this function should also output an indicator as to the provenance of the name, of which, in the GSS-API, there are three: imported, inquired from a credential, and a peer's name inquired from a security context.]

Return major_status codes:

- GSS_S_COMPLETE indicates no error.
- GSS_S_FAILURE indicates a general error.

This function outputs the sets of attributes of a name, that are authenticated, asserted or critical. It also indicates if a given NAME is an MN or not and, if it is, what mechanism it's an MN of.

8.1. C-Bindings

```
OM_uint32 gss_inquire_name(
  OM_uint32                    *minor_status,
  gss_name_t                   name,
  int                          name_is_MN,
  gss_OID                      *MN_mech,
```

```
    gss_OID_set                 *authenticated,
    gss_OID_set                 *asserted,
    gss_OID_set                 *critical,
    gss_OID_set                 *all_attrs
  );
```

9. GSS_Get_name_attribute()

Inputs:

- name NAME,
- attr OBJECT IDENTIFIER

Outputs:

- major_status INTEGER,
- minor_status INTEGER,
- authenticated BOOLEAN, -- FALSE if asserted but not authenticated by a trusted entity
- negative BOOLEAN,
- mapped BOOLEAN,
- critical BOOLEAN,
- values SET OF OCTET STRING,
- display_values SET OF STRING

Return major_status codes:

- GSS_S_COMPLETE indicates no error.
- GSS_S_UNAVAILABLE indicates that the given attribute OID is not known or set.
- GSS_S_FAILURE indicates a general error.

This function outputs the value(s) associated with a given GSS name object for a given name attribute.

NOTE: This function relies on the GSS-API notion of "SET OF" allowing for order preservation; this has been discussed on the KITTEN WG mailing list and the consensus seems to be that, indeed, that was always the intention.

9.1. C-Bindings

The C-bindings of GSS_Get_name_attribute() requires one function call per-attribute value, for multi-valued name attributes. This is done by using a single gss_buffer_t for each value and an input/output integer parameter to distinguish initial and subsequent calls and to indicate when all values have been obtained.

The 'more' input/output parameter should point to an integer variable whose value, on first call to gss_name_attribute_get() MUST be -1, and whose value upon function call return will be non-zero to indicate that additional values remain, or zero to indicate that no values remain. The caller should not modify this parameter after the initial call.

```
OM_uint32 gss_get_name_attribute(
  OM_uint32                    *minor_status,
  gss_name_t                   name,
  gss_OID                      attr,
  int                          *authenticated,
  int                          *negative,
  int                          *mapped,
  int                          *critical,
  gss_buffer_t                 value,
  gss_buffer_t                 display_value,
  int                          *more
);
```

10. GSS_Set_name_attribute()

Inputs:

- name NAME,
- critical BOOLEAN,
- negative BOOLEAN,
- attr OBJECT IDENTIFIER,
- values SET OF OCTET STRING

Outputs:

- major_status INTEGER,
- minor_status INTEGER

Return major_status codes:

- GSS_S_COMPLETE indicates no error.
- GSS_S_UNAVAILABLE indicates that the given attribute OID is not known or could not be set.
- GSS_S_FAILURE indicates a general error.

NOTE: This function relies on the GSS-API notion of "SET OF" allowing for order preservation; this has been discussed on the KITTEN WG mailing list and the consensus seems to be that, indeed, that was always the intention.

10.1. C-Bindings

The C-bindings of GSS_Set_name_attribute() requires one function call per-attribute value, for multi-valued name attributes -- each call adds one value. To replace an attribute's every value delete the attribute's values first with GSS_Delete_name_attribute().

```
OM_uint32 gss_set_name_attribute(
  OM_uint32                    *minor_status,
  gss_name_t                   name,
  int                          critical,
  int                          negative,
  gss_OID                      attr,
  gss_buffer_t                 value
);
```

11. GSS_Delete_name_attribute()

Inputs:

- name NAME,
- attr OBJECT IDENTIFIER,

Outputs:

- major_status INTEGER,

- minor_status INTEGER

Return major_status codes:

- GSS_S_COMPLETE indicates no error.

- GSS_S_UNAVAILABLE indicates that the given attribute OID is not known.

- GSS_S_FAILURE indicates a general error.

Deletion of negative authenticated attributes from NAME objects MUST NOT be allowed. [Do we need a new major status code for "permission denied"?]

11.1. C-Bindings

```
OM_uint32 gss_delete_name_attribute(
  OM_uint32                    *minor_status,
  gss_name_t                   name,
  gss_OID                      attr
);
```

12. GSS_Export_name_composite()

Inputs:

- name NAME

Outputs:

- major_status INTEGER,

- minor_status INTEGER,

- exp_composite_name OCTET STRING

Return major_status codes:

- GSS_S_COMPLETE indicates no error.

- GSS_S_FAILURE indicates a general error.

This function outputs a token which can be imported with GSS_Import_name(), using GSS_C_NT_COMPOSITE_EXPORT as the name type and which preserves any name attribute information associated with the input name (which GSS_Export_name() may well not). The token format is no specified here as this facility is intended for inter-process communication only; however, all such tokens MUST start with a two-octet token ID, hex 04 02, in network byte order.

The OID for GSS_C_NT_COMPOSITE_EXPORT is <TBD>.

12.1. C-Bindings

```
OM_uint32 gss_export_name_composite(
  OM_uint32                    *minor_status,
  gss_name_t                   name,
  gss_buffer_t                 exp_composite_name
);
```

13. GSS_Map_name_to_any()

Inputs:

- name NAME,
- authenticated BOOLEAN, -- if TRUE no data will be output unless it is authenticated
- type_id OBJECT IDENTIFIER

Outputs:

- major_status INTEGER,
- minor_status INTEGER,
- output ANY DEFINED BY type_id

Return major_status codes:

- GSS_S_COMPLETE indicates no error.
- GSS_S_UNAVAILABLE indicates that the mapping or conversion could not be done. The minor status code may provide additional information.

- GSS_S_FAILURE indicates a general error. The minor status code may provide additional information.

Whereas name attribute's values are encoded in some network representation applications often require native, language- and/or platform-specific data types. This function provides access to such types.

13.1. C-Bindings

```
typedef struct gss_any *gss_any_t;
OM_uint32 gss_map_name_to_any(
  OM_uint32                    *minor_status,
  gss_name_t                   name,
  int                          authenticated,
  gss_OID                      type_id,
  gss_any_t                    output
);
```

Note the new C bindings type, gss_any_t. We define it as a pointer to an incompletely declared struct.

14. GSS_Release_any_name_mapping()

Inputs:

- name NAME,
- type_id OBJECT IDENTIFIER,
- input ANY DEFINED BY type_id

Outputs:

- major_status INTEGER,
- minor_status INTEGER,

Return major_status codes:

- GSS_S_COMPLETE indicates no error.
- GSS_S_UNAVAILABLE indicates that the mapping or conversion could not be done. The minor status code may provide additional information.

- GSS_S_FAILURE indicates a general error. The minor status code may provide additional information.

This function releases, if possible, the objects of language- and/or platform-specific types output by GSS_Map_name_to_any(). If such types have native release functions applications MAY use either those or this function to release the given object.

14.1. C-Bindings

```
typedef struct gss_any *gss_any_t;
OM_uint32 gss_release_any_name_mapping(
  OM_uint32                    *minor_status,
  gss_name_t                   name,
  gss_OID                      type_id,
  gss_any_t                    *input
);
```

15. IANA Considerations

This document creates a namespace of GSS-API name attributes. Attributes are named by OID, so no single authority might be needed for allocation, however, in the interest of providing the community with an authority for name attribute OID allocation and a way to find the existing set of name attributes, the IANA should establish both, a single OID off of which name attributes could be allocated, and a registry of known GSS name attributes.

GSS-API name attribute registry entries should contain all the information that GSS_Inquire_name_attribute() may return about the given name attributes and their OIDs:

- a name attribute OID (this is a unique key)
- a name attribute symbolic name, starting with "GSS_C_NA_" (this is a unique key)
- a brief description, in English
- whether the attribute is useful as the subject of access control list entries
- whether the attribute is useful as an indicator of trust
- an optional normative reference to documentation for the given name attribute

The allocation and registration policy should be first come, first served. Registry entries' OIDs need not be based on the base OID given above.

16. Security Considerations

<TBA>

[In particular, the status of a name attribute as "authenticated" vs. "asserted" requires close review, particularly with respect to PKIX certificate extensions.]

[Also, we need to work out the security considerations of (and possibly remove) negative attributes.]

17. Normative References

[I-D.GSS-NAMING] Hartman, S., "Desired Enhancements to GSSAPI Naming", draft-ietf-kitten-gss-naming-01.txt (work in progress), February 2005.

[I-D.ietf-krb-wg-kerberos-clarifications] Neuman, C., "The Kerberos Network Authentication Service (V5)", draft-ietf-krb-wg-kerberos-clarifications-07 (work in progress), September 2004.

[RFC2025] Adams, C., "The Simple Public-Key GSS-API Mechanism (SPKM)", RFC 2025, October 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2743] Linn, J., "Generic Security Service Application Program Interface Version 2, Update 1", RFC 2743, January 2000.

[RFC2744] Wray, J., "Generic Security Service API Version 2 : C-bindings", RFC 2744, January 2000.

[RFC3280] Housley, R., Polk, W., Ford, W., and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002.

Author's Address

Nicolas Williams
Sun Microsystems
5300 Riata Trace Ct
Austin, TX 78727
US

Email: Nicolas.Williams@sun.com

Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Copyright Statement

Copyright (C) The Internet Society (2005). This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.