          Extended Generic Security Service Mechanism Inquiry APIs

Abstract

   This document introduces new application programming interfaces
   (APIs) to the Generic Security Services API (GSS-API) for extended
   mechanism attribute inquiry. These interfaces are primarily intended
   to reduce instances of hardcoding of mechanism identifiers in GSS
   applications.

   These interfaces include mechanism attributes and attribute sets, a
   function for inquiring the attributes of a mechanism, a function for
   indicating mechanisms that possess given attributes, and a function
   for displaying mechanism attributes.

Status of This Memo

   This document specifies an Internet standards track protocol for the
   Internet community, and requests discussion and suggestions for
   improvements. Please refer to the current edition of the "Internet
   Official Protocol Standards" (STD 1) for the standardization state
   and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Table of Contents

1. Introduction

   GSS-API [RFC2743] mechanisms have a number of properties that may be
   of interest to applications. The lack of APIs for inquiring about
   available mechanisms' properties has meant that many GSS-API
   applications must hardcode mechanism Object Identifiers (OIDs).
   Ongoing work may result in a variety of new GSS-API mechanisms.
   Applications should not have to hardcode their OIDs.

   For example, the Secure Shell version 2 (SSHv2) protocol [RFC4251]
   supports the use of GSS-API mechanisms for authentication [RFC4462]
   but explicitly prohibits the use of Simple and Protected GSS-API
   Negotiation (SPNEGO) [RFC4178]. Future mechanisms that negotiate
   mechanisms would have to be forbidden as well, but there is no way to
   implement applications that inquire what mechanisms are available and
   then programmatically exclude mechanisms "like SPNEGO".

2. 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].

3. New GSS-API Interfaces

   We introduce a new concept -- that of mechanism attributes. By
   allowing applications to query the set of attributes associated with
   individual mechanisms and to find out which mechanisms support a
   given set of attributes, we allow applications to select mechanisms
   based on their attributes without having to hardcode mechanism OIDs.

   Section 3.1 describes the mechanism attributes concept. Sections
   3.4.2, 3.4.3, and 3.4.4 describe three new interfaces that deal in
   mechanisms and attribute sets:

   o GSS_Indicate_mechs_by_attrs()

   o GSS_Inquire_attrs_for_mech()

   o GSS_Display_mech_attr()

3.1. Mechanism Attributes and Attribute Sets

   An abstraction for the features provided by mechanisms and pseudo-
   mechanisms is needed in order to facilitate the programmatic
   selection of mechanisms. Pseudo-mechanisms are mechanisms that make
   reference to other mechanisms in order to provide their services.
   For example, SPNEGO is a pseudo-mechanism, for without other
   mechanisms SPNEGO is useless.

   Two data types are needed: one for individual mechanism attributes
   and one for mechanism attribute sets. To simplify the mechanism
   attribute interfaces, we reuse the 'OID' and 'OID set' data types and
   model individual mechanism attribute types as OIDs.

   To this end, we define an open namespace of mechanism attributes and
   assign them arcs off of this OID:

   <1.3.6.1.5.5.13>

   Each mechanism has a set of mechanism attributes that it supports as
   described in its specification.

3.2. List of Known Mechanism Attributes

```
+-------------------------+---------+-------------------------+
| Mech Attr Name          | OID Arc | Arc Name                |
+-------------------------+---------+-------------------------+
| GSS_C_MA_MECH_CONCRETE   |     (1) | concrete-mech           |
| GSS_C_MA_MECH_PSEUDO     |     (2) | pseudo-mech             |
| GSS_C_MA_MECH_COMPOSITE  |     (3) | composite-mech          |
| GSS_C_MA_MECH_NEGO       |     (4) | mech-negotiation-mech   |
| GSS_C_MA_MECH_GLUE       |     (5) | mech-glue               |
| GSS_C_MA_NOT_MECH        |     (6) | not-mech                |
| GSS_C_MA_DEPRECATED      |     (7) | mech-deprecated         |
| GSS_C_MA_NOT_DFLT_MECH   |     (8) | mech-not-default        |
| GSS_C_MA_ITOK_FRAMED     |     (9) | initial-is-framed       |
| GSS_C_MA_AUTH_INIT       |    (10) | auth-init-princ         |
| GSS_C_MA_AUTH_TARG       |    (11) | auth-targ-princ         |
| GSS_C_MA_AUTH_INIT_INIT  |    (12) | auth-init-princ-initial |
| GSS_C_MA_AUTH_TARG_INIT  |    (13) | auth-targ-princ-initial |
| GSS_C_MA_AUTH_INIT_ANON  |    (14) | auth-init-princ-anon    |
| GSS_C_MA_AUTH_TARG_ANON  |    (15) | auth-targ-princ-anon    |
| GSS_C_MA_DELEG_CRED      |    (16) | deleg-cred              |
| GSS_C_MA_INTEG_PROT      |    (17) | integ-prot              |
| GSS_C_MA_CONF_PROT       |    (18) | conf-prot               |
| GSS_C_MA_MIC             |    (19) | mic                     |
| GSS_C_MA_WRAP            |    (20) | wrap                    |
| GSS_C_MA_PROT_READY      |    (21) | prot-ready              |
| GSS_C_MA_REPLAY_DET      |    (22) | replay-detection        |
| GSS_C_MA_OOS_DET         |    (23) | oos-detection           |
| GSS_C_MA_CBINDINGS       |    (24) | channel-bindings        |
| GSS_C_MA_PFS             |    (25) | pfs                     |
| GSS_C_MA_COMPRESS        |    (26) | compress                |
| GSS_C_MA_CTX_TRANS       |    (27) | context-transfer        |
| <reserved>              | (28...) |                         |
+-------------------------+---------+-------------------------+
```

                            Table 1

```
+------------------------+----------------------------------------+
| Mech Attr Name         | Purpose                                |
+------------------------+----------------------------------------+
| GSS_C_MA_MECH_CONCRETE | Indicates that a mech is neither a     |
|                        | pseudo-mechanism nor a composite       |
|                        | mechanism.                             |
| GSS_C_MA_MECH_PSEUDO   | Indicates that a mech is a             |
|                        | pseudo-mechanism.                      |
| GSS_C_MA_MECH_COMPOSITE| Indicates that a mech is a composite of|
|                        | other mechanisms.  This is reserved for|
|                        | a specification of "stackable"         |
|                        | pseudo-mechanisms.                     |
| GSS_C_MA_MECH_NEGO     | Indicates that a mech negotiates other |
|                        | mechs (e.g., SPNEGO has this           |
|                        | attribute).                            |
| GSS_C_MA_MECH_GLUE     | Indicates that the OID is not for a    |
|                        | mechanism but for the GSS-API itself.  |
| GSS_C_MA_NOT_MECH      | Indicates that the OID is known, yet it|
|                        | is also known not to be the OID of any |
|                        | GSS-API mechanism (or of the GSS-API   |
|                        | itself).                               |
| GSS_C_MA_DEPRECATED    | Indicates that a mech (or its OID) is  |
|                        | deprecated and MUST NOT be used as a   |
|                        | default mechanism.                     |
| GSS_C_MA_NOT_DFLT_MECH | Indicates that a mech (or its OID) MUST|
|                        | NOT be used as a default mechanism.    |
| GSS_C_MA_ITOK_FRAMED   | Indicates that the given mechanism's   |
|                        | initial context tokens are properly    |
|                        | framed as per Section 3.1 of [RFC2743].|
| GSS_C_MA_AUTH_INIT     | Indicates support for authentication of|
|                        | initiator to acceptor.                 |
| GSS_C_MA_AUTH_TARG     | Indicates support for authentication of|
|                        | acceptor to initiator.                 |
| GSS_C_MA_AUTH_INIT_INIT| Indicates support for "initial"        |
|                        | authentication of initiator to         |
|                        | acceptor.  "Initial authentication"    |
|                        | refers to the use of passwords, or keys|
|                        | stored on tokens, for authentication.  |
|                        | Whether a mechanism supports initial   |
|                        | authentication may depend on IETF      |
|                        | consensus (see Security                |
|                        | Considerations).                       |
| GSS_C_MA_AUTH_TARG_INIT| Indicates support for initial          |
|                        | authentication of acceptor to          |
|                        | initiator.                             |
| GSS_C_MA_AUTH_INIT_ANON| Indicates support for                  |
|                        | GSS_C_NT_ANONYMOUS as an initiator     |
|                        | principal name.                        |
```

```
| GSS_C_MA_AUTH_TARG_ANON | Indicates support for              |
|                         | GSS_C_NT_ANONYMOUS as a target     |
|                         | principal name.                    |
| GSS_C_MA_DELEG_CRED     | Indicates support for credential   |
|                         | delegation.                        |
| GSS_C_MA_INTEG_PROT     | Indicates support for per-message  |
|                         | integrity protection.              |
| GSS_C_MA_CONF_PROT      | Indicates support for per-message  |
|                         | confidentiality protection.        |
| GSS_C_MA_MIC            | Indicates support for Message Integrity |
|                         | Code (MIC) tokens.                 |
| GSS_C_MA_WRAP           | Indicates support for WRAP tokens. |
| GSS_C_MA_PROT_READY     | Indicates support for per-message  |
|                         | protection prior to full context   |
|                         | establishment.                     |
| GSS_C_MA_REPLAY_DET     | Indicates support for replay detection. |
| GSS_C_MA_OOS_DET        | Indicates support for out-of-sequence |
|                         | detection.                         |
| GSS_C_MA_CBINDINGS      | Indicates support for channel bindings. |
| GSS_C_MA_PFS            | Indicates support for Perfect Forward |
|                         | Security.                          |
| GSS_C_MA_COMPRESS       | Indicates support for compression of |
|                         | data inputs to GSS_Wrap().         |
| GSS_C_MA_CTX_TRANS      | Indicates support for security context |
|                         | export/import.                     |
+-------------------------+------------------------------------------+
```

                                Table 2

3.3. Mechanism Attribute Sets of Existing Mechs

   The Kerberos V mechanism [RFC1964] provides the following mechanism
   attributes:

   o   GSS_C_MA_MECH_CONCRETE

   o   GSS_C_MA_ITOK_FRAMED

   o   GSS_C_MA_AUTH_INIT

   o   GSS_C_MA_AUTH_TARG

   o   GSS_C_MA_DELEG_CRED

   o   GSS_C_MA_INTEG_PROT

   o   GSS_C_MA_CONF_PROT

   o GSS_C_MA_MIC

   o GSS_C_MA_WRAP

   o GSS_C_MA_PROT_READY

   o GSS_C_MA_REPLAY_DET

   o GSS_C_MA_OOS_DET

   o GSS_C_MA_CBINDINGS

   o GSS_C_MA_CTX_TRANS (some implementations, using implementation-
      specific exported context token formats)

   The Kerberos V mechanism also has a deprecated OID that has the same
   mechanism attributes as above as well as GSS_C_MA_DEPRECATED.

   The mechanism attributes of the Simple Public-Key GSS-API Mechanism
   (SPKM) [RFC2025] family of mechanisms will be provided in a separate
   document, as SPKM is currently being reviewed for possibly
   significant changes due to problems in its specifications.

   The Low Infrastructure Public Key (LIPKEY) mechanism [RFC2847] offers
   the following attributes:

   o GSS_C_MA_MECH_CONCRETE

   o GSS_C_MA_ITOK_FRAMED

   o GSS_C_MA_AUTH_INIT_INIT

   o GSS_C_MA_AUTH_TARG (from SPKM-3)

   o GSS_C_MA_AUTH_TARG_ANON (from SPKM-3)

   o GSS_C_MA_INTEG_PROT

   o GSS_C_MA_CONF_PROT

   o GSS_C_MA_REPLAY_DET

   o GSS_C_MA_OOS_DET

   o GSS_C_MA_CTX_TRANS (some implementations, using implementation-
      specific exported context token formats)

(LIPKEY should also provide GSS_C_MA_CBINDINGS, but SPKM-3
requires clarifications on this point.)

The SPNEGO mechanism [RFC4178] provides the following attributes:

o GSS_C_MA_MECH_NEGO

o GSS_C_MA_ITOK_FRAMED

All other mechanisms' attributes will be described elsewhere.

3.4. New GSS-API Function Interfaces

Several new interfaces are given by which, for example, GSS-API
applications may determine what features are provided by a given
mechanism and what mechanisms provide what features.

These new interfaces are all OPTIONAL.

Applications should use GSS_Indicate_mechs_by_attrs() instead of
GSS_Indicate_mechs() wherever possible.

Applications can use GSS_Indicate_mechs_by_attrs() to determine what,
if any, mechanisms provide a given set of features.

GSS_Indicate_mechs_by_attrs() can also be used to indicate (as in
GSS_Indicate_mechs()) the set of available mechanisms of each type
(concrete, mechanism negotiation pseudo-mechanism, etc.).

3.4.1. Mechanism Attribute Criticality

Mechanism attributes may be added at any time. Not only may
attributes be added to the list of known mechanism attributes at any
time, but the set of mechanism attributes supported by a mechanism
can be changed at any time.

For example, new attributes might be added to reflect whether a
mechanism's initiator must contact an online infrastructure and/or
whether the acceptor must do so. In this example, the Kerberos V
mechanism would gain a new attribute even though the mechanism itself
is not modified.

Applications making use of attributes not defined herein would then
have no way of knowing whether a GSS-API implementation and its
mechanisms know about new mechanism attributes. To address this
problem, GSS_Indicate_mechs_by_attrs() and
GSS_Inquire_attrs_for_mech() support a notion of critical mechanism
attributes. Applications can search for mechanisms that understand

   mechanism attributes that are critical to the application, and the
   application may ask what mechanism attributes are understood by a
   given mechanism.

3.4.2. GSS_Indicate_mechs_by_attrs()

   Inputs:

   o desired_mech_attrs SET OF OBJECT IDENTIFIER -- set of GSS_C_MA_*
      OIDs that the mechanisms indicated in the mechs output parameter
      MUST offer.

   o except_mech_attrs SET OF OBJECT IDENTIFIER -- set of GSS_C_MA_*
      OIDs that the mechanisms indicated in the mechs output parameter
      MUST NOT offer.

   o critical_mech_attrs SET OF OBJECT IDENTIFIER -- set of GSS_C_MA_*
      OIDs that the mechanisms indicated in the mechs output parameter
      MUST understand (i.e., mechs must know whether critical attributes
      are or are not supported).

   Outputs:

   o major_status INTEGER

   o minor_status INTEGER

   o mechs SET OF OBJECT IDENTIFIER -- set of mechanisms that support
      the given desired_mech_attrs but not the except_mech_attrs, and
      all of which understand the given critical_mech_attrs (the caller
      must release this output with GSS_Release_oid_set()).

   Return major_status codes:

   o GSS_S_COMPLETE indicates success; the output mechs parameter MAY
      be the empty set (GSS_C_NO_OID_SET).

   o GSS_S_FAILURE indicates that the request failed for some other
      reason.

   GSS_Indicate_mechs_by_attrs() returns the set of OIDs corresponding
   to mechanisms that offer at least the desired_mech_attrs but none of
   the except_mech_attrs, and that understand all of the attributes
   listed in critical_mech_attrs.

   When all three sets of OID input parameters are the empty set, this
   function acts as a version of GSS_indicate_mechs() that outputs the
   set of all supported mechanisms.

3.4.3. GSS_Inquire_attrs_for_mech()

   Inputs:

   o mech OBJECT IDENTIFIER -- mechanism OID

   Outputs:

   o major_status INTEGER

   o minor_status INTEGER

   o mech_attrs SET OF OBJECT IDENTIFIER -- set of mech_attrs OIDs
      (GSS_C_MA_*) supported by the mechanism (the caller must release
      this output with GSS_Release_oid_set()).

   o known_mech_attrs SET OF OBJECT IDENTIFIER -- set of mech_attrs
      OIDs known to the mechanism implementation (the caller must
      release this output with GSS_Release_oid_set()).

   Return major_status codes:

   o GSS_S_COMPLETE indicates success; the output mech_attrs parameter
      MAY be the empty set (GSS_C_NO_OID_SET).

   o GSS_S_BAD_MECH indicates that the mechanism named by the mech
      parameter does not exist or that the mech is GSS_C_NO_OID and no
      default mechanism could be determined.

   o GSS_S_FAILURE indicates that the request failed for some other
      reason.

   GSS_Inquire_attrs_for_mech() indicates the set of mechanism
   attributes supported by a given mechanism.

3.4.4. GSS_Display_mech_attr()

   Inputs:

   o mech_attr OBJECT IDENTIFIER -- mechanism attribute OID

   Outputs:

   o major_status INTEGER

   o minor_status INTEGER

   o name OCTET STRING, -- name of mechanism attribute (e.g.,
      GSS_C_MA_*).

   o short_desc OCTET STRING, -- a short description of the mechanism
      attribute (the caller must release this output with
      GSS_Release_buffer()).

   o long_desc OCTET STRING -- a longer description of the mechanism
      attribute (the caller must release this output with
      GSS_Release_buffer()).

   Return major_status codes:

   o GSS_S_COMPLETE indicates success.

   o GSS_S_BAD_MECH_ATTR indicates that the mechanism attribute
      referenced by the mech_attr parameter is unknown to the
      implementation.

   o GSS_S_FAILURE indicates that the request failed for some other
      reason.

   This function can be used to obtain human-readable descriptions of
   GSS-API mechanism attributes.

3.4.5. New Major Status Values

   A single, new, major status code is added for
   GSS_Display_mech_attr():

   o GSS_S_BAD_MECH_ATTR,

   roughly corresponding to GSS_S_BAD_MECH but applicable to mechanism
   attribute OIDs rather than to mechanism OIDs.

   For the C-bindings of the GSS-API [RFC2744], GSS_S_BAD_MECH_ATTR
   shall have a routine error number of 19 (this is shifted to the left
   by GSS_C_ROUTINE_ERROR_OFFSET).

3.4.6. C-Bindings

   Note that there is a bug in the C bindings of the GSS-APIv2u1
   [RFC2744] in that the C 'const' attribute is applied to types that
   are pointer typedefs. This is a bug because it declares that the
   pointer argument is 'const' rather than that the object pointed by it
   is const. To avoid this error, we hereby define new typedefs, which
   include const properly:

```
typedef const gss_buffer_desc * gss_const_buffer_t;
typedef const struct gss_channel_bindings_struct *
   gss_const_channel_bindings_t;
typedef const <platform-specific> gss_const_ctx_id_t;
typedef const <platform-specific> gss_const_cred_id_t;
typedef const <platform-specific> gss_const_name_t;
typedef const gss_OID_desc * gss_const_OID;
typedef const gss_OID_set_desc * gss_const_OID_set;
```

                        Figure 1: const typedefs

   Note that only gss_const_OID and gss_const_OID_set are used below.
   We include the other const typedefs for convenience since the C
   bindings of the GSS-API do use const with pointer typedefs when it
   should often instead use the above typedefs instead.

```
#define GSS_S_BAD_MECH_ATTR (19ul << GSS_C_ROUTINE_ERROR_OFFSET)

OM_uint32 gss_indicate_mechs_by_attrs(
    OM_uint32         *minor_status,
    gss_const_OID_set  desired_mech_attrs,
    gss_const_OID_set  except_mech_attrs,
    gss_const_OID_set  critical_mech_attrs,
    gss_OID_set        *mechs);

OM_uint32 gss_inquire_attrs_for_mech(
    OM_uint32         *minor_status,
    gss_const_OID      mech,
    gss_OID_set        *mech_attrs,
    gss_OID_set        *known_mech_attrs);

OM_uint32 gss_display_mech_attr(
    OM_uint32         *minor_status,
    gss_const_OID      mech_attr,
    gss_buffer_t       name,
    gss_buffer_t       short_desc,
    gss_buffer_t       long_desc);
```

                        Figure 2: C bindings

   Note that output buffers must be released via gss_release_buffer().
   Output OID sets must be released via gss_release_oid_set().

   Please see Appendix A for a full set of typedef fragments defined in
   this document and the necessary code license.

4. Requirements for Mechanism Designers

    All future GSS-API mechanism specifications MUST:

    o   list the set of GSS-API mechanism attributes associated with them.

5. IANA Considerations

    The namespace of programming-language symbols with names beginning
    with GSS_C_MA_* is reserved for allocation by IETF Consensus. IANA
    allocated a base OID, as an arc of 1.3.6.1.5.5, for the set of
    GSS_C_MA_* described herein, and registered all of the GSS_C_MA_*
    values described in Section 3.2.

6. Security Considerations

    This document specifies extensions to a security-related API. It
    imposes new requirements on future GSS-API mechanisms, and the
    specifications of future protocols that use the GSS-API should make
    reference to this document where applicable. The ability to inquire
    about specific properties of mechanisms should improve security.

    The semantics of each mechanism attribute may include a security
    component.

    Application developers must understand that mechanism attributes may
    be added at any time -- both to the set of known mechanism attributes
    as well as to existing mechanisms' sets of supported mechanism
    attributes. Therefore, application developers using the APIs
    described herein must understand what mechanism attributes their
    applications depend critically on, and must use the mechanism
    attribute criticality features of these APIs.

7. References

7.1. Normative References

    [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.

7.2. Informative References

   [RFC1964] Linn, J., "The Kerberos Version 5 GSS-API Mechanism",
              RFC 1964, June 1996.

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

   [RFC2847] Eisler, M., "LIPKEY - A Low Infrastructure Public Key
              Mechanism Using SPKM", RFC 2847, June 2000.

   [RFC4178] Zhu, L., Leach, P., Jaganathan, K., and W. Ingersoll, "The
              Simple and Protected Generic Security Service Application
              Program Interface (GSS-API) Negotiation Mechanism",
              RFC 4178, October 2005.

   [RFC4251] Ylonen, T. and C. Lonvick, "The Secure Shell (SSH)
              Protocol Architecture", RFC 4251, January 2006.

   [RFC4462] Hutzelman, J., Salowey, J., Galbraith, J., and V. Welch,
              "Generic Security Service Application Program Interface
              (GSS-API) Authentication and Key Exchange for the Secure
              Shell (SSH) Protocol", RFC 4462, May 2006.

Appendix A. Typedefs and C Bindings

   This appendix contains the full set of code fragments defined in this
   document.

```
   typedef const gss_buffer_desc * gss_const_buffer_t;
   typedef const struct gss_channel_bindings_struct *
      gss_const_channel_bindings_t;
   typedef const <platform-specific> gss_const_ctx_id_t;
   typedef const <platform-specific> gss_const_cred_id_t;
   typedef const <platform-specific> gss_const_name_t;
   typedef const gss_OID_desc * gss_const_OID;
   typedef const gss_OID_set_desc * gss_const_OID_set;
```

```
   #define GSS_S_BAD_MECH_ATTR (19ul << GSS_C_ROUTINE_ERROR_OFFSET)

   OM_uint32 gss_indicate_mechs_by_attrs(
      OM_uint32          *minor_status,
      gss_const_OID_set  desired_mech_attrs,
      gss_const_OID_set  except_mech_attrs,
      gss_const_OID_set  critical_mech_attrs,
      gss_OID_set        *mechs);

   OM_uint32 gss_inquire_attrs_for_mech(
      OM_uint32          *minor_status,
      gss_const_OID       mech,
      gss_OID_set        *mech_attrs,
      gss_OID_set        *known_mech_attrs);

   OM_uint32 gss_display_mech_attr(
      OM_uint32          *minor_status,
      gss_const_OID       mech_attr,
      gss_buffer_t        name,
      gss_buffer_t        short_desc,
      gss_buffer_t        long_desc);
```

Author's Address

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

   EMail: Nicolas.Williams@sun.com