

Network Working Group
Williams
Internet-Draft
Cryptonector
Updates: [RFC2743](#) RFC2744
2013
(if approved)
Intended status: Standards Track
Expires: August 16, 2013

N.

February 12,

Channel Binding Signalling for the Generic Security Services
Application

Programming Interface
[draft-williams-kitten-channel-bound-flag-01](#)

Abstract

This Internet-Draft proposes the addition of a "channel bound" return flag for the `GSS_Init_sec_context()` and `GSS_Accept_sec_context()` functions. Two behaviors are specified: a default, safe behavior, and a behavior that is only safe when the application specifically tells the Generic Security Services Application Programming Interface (GSS-API) that it (the applicaiton) supports the new behavior.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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."

This Internet-Draft will expire on August 16, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

Williams
1]

Expires August 16, 2013

[Page

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction
3
1.1. Error in [RFC2743](#)
3
1.2. Alternative Designs
3
1.3. Conventions used in this document
4
2. Channel Binding State Extension
5
2.1. GSS_Set_cred_option()
5
2.1.1. C-Bindings
5
2.2. GSS_Set_cred_option_critical()
6
2.2.1. C-Bindings
7
2.3. Return Flag for Channel Binding State Signalling
7
2.3.1. C-Bindings
7
2.4. Credential Option For Channel Binding Negotiation
Semantics Selection
7
2.4.1. C-Bindings
8
3. References
9
3.1. Normative References
9
3.2. Informative References
9
10 Author's Address

Williams
2]

Expires August 16, 2013

[Page

1. Introduction

The GSS-API [[RFC2743](#)] supports "channel binding" [[RFC5056](#)], a technique for detection of man-in-the-middle (MITM) attacks in secure

channels at lower network layers. This facility is meant to be all-or-nothing: either both the initiator and acceptor use it and it succeeds, or both must not use it. This has created a negotiation problem when retrofitting the use of channel binding into existing application protocols.

Many implementations of the Kerberos V5 GSS-API mechanism [[RFC4121](#)] cause the acceptor to succeed when the initiator used channel binding

but the acceptor application did not. This has helped deployment of channel binding in existing applications: first fix all the initiators, then fix all the acceptors. But even this is insufficient when there are many clients to fix, such that fixing them all will take a long time.

This document proposes a new method for deployment of channel binding

that allows the feature to be enabled on the acceptor side before fixing all initiators. If the GSS-API had always had a return flag by which to indicate channel binding state then we could have had a simpler method of deploying channel binding: applications check that return flag and act accordingly (e.g., fail when channel binding is required). We cannot safely introduce this behavior now without an indication of support by the application.

It is worth noting that at least one implementor of GSS-API mechanisms (but not of the GSS-API itself) has similar semantics in its API to those proposed herein. [XXX add references to the relevant SSPI docs? -Nico]

1.1. Error in [RFC2743](#)

The GSS-APIv2u1 [[RFC2743](#)] seems to indicate that mechanisms must ignore channel bindings when one party provided none. In practice some mechanisms ignore channel bindings when the acceptor provides none, but not when the initiator provides none. Note that it would be useless to allow security context establishment to succeed when the initiator does not provide channel bindings but the acceptor does, at least as long as there's no outward indication of whether channel binding was used! And indeed, the GSS-APIv2u1 does not provide any such indication. We correct this flaw in this document.

1.2. Alternative Designs

Other ways to design the selection of channel binding signalling semantics are feasible. One option would be to add a new version of

Williams
3]

Expires August 16, 2013

[Page

GSS_Accept_sec_context() that takes an input flags parameter.
Another option would be to add new versions of
GSS_Init_sec_context()
and GSS_Accept_sec_context() that take additional input parameters
for specifying arbitrary options in an extensible way. However, we
have a number of extensions that could be delivered in this fashion

-

it would take a long time to negotiate a suitable design to minimize
the number of additional variants of GSS_Init_sec_context() and/or
GSS_Accept_sec_context(). Using CREDENTIAL HANDLES to communicate
these options seems like a very light-weight extension in comparison
to introducing new versions of security context establishment
functions. Additionally, the notion of credential handle options is
already present in several implementations, thus using an existing
design has the benefit of resulting in less code/feature bloat.

1.3. 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](#)].

Williams
4]

Expires August 16, 2013

[Page

2. Channel Binding State Extension

We propose a new return flag for `GSS_Init_sec_context()` and `GSS_Accept_sec_context()`, as well as a pair of functions for setting options on credential handles, along with an option for signalling understanding of the new flag in the acceptor applications.

C bindings of these extensions are provided along the lines of [\[RFC2744\]](#) and [\[RFC5587\]](#).

2.1. GSS_Set_cred_option()

Inputs:

`cred_handle` CREDENTIAL HANDLE If no credential handle is given then the option MAY be applied globally to the default credential handle, but the implementation MAY return an error instead.

`desired_object` OBJECT IDENTIFIER Desired option; MUST NOT be `GSS_C_NO_OID`.

`value` OCTET STRING Value for the option.

Outputs:

o `major_status` INTEGER

o `minor_status` INTEGER

Return major status codes:

o `GSS_S_COMPLETE` indicates success.

o `GSS_S_UNAVAILABLE` indicates that the the given option is not supported by any mechanism.

o `GSS_S_FAILURE` indicates a general failure.

This function sets the given value to a credential option named by `desired_object` on the given `cred_handle`.

2.1.1. C-Bindings

Williams
5]

Expires August 16, 2013

[Page

```
OM_uint32
gss_set_cred_option(OM_uint32 *minor_status,
                    gss_cred_id_t *cred_handle,
                    gss_const_OID desired_object,
                    gss_const_buffer_t value);
```

Figure 1: C-Bindings of GSS_Set_cred_option()

NOTE: the cred_handle input argument to gss_set_cred_option() is a pointer to gss_cred_id_t for historical reasons. This is in conflict with the regular GSS-API pattern, but it cannot be changed at this stage. [XXX We could rename this function and not document gss_set_cred_option(). Should we? -Nico]

2.2. GSS_Set_cred_option_critical()

Inputs:

input_cred_handle CREDENTIAL HANDLE If no credential handle is given

then the option MAY be applied globally to the default credential handle, but the implementation MAY return an error instead.

desired_object OBJECT IDENTIFIER Desired option; MUST NOT be GSS_C_NO_OID.

value OCTET STRING Value for the option.

Outputs:

- o output_cred_handle CREDENTIAL_HANDLE
- o major_status INTEGER
- o minor_status INTEGER

Return major status codes:

- o GSS_S_COMPLETE indicates success.
- o GSS_S_UNAVAILABLE indicates that the the given option is not supported by any mechanism.
- o GSS_S_FAILURE indicates a general failure.

This function sets the given value to a credential option named by desired_object on the given input_cred_handle or on a duplicate handle output in the output_cred_handle parameter if desired. If any mechanisms -for which the credential has elements- fails to set the

Williams
6]

Expires August 16, 2013

[Page

option then that element will be removed from the credential.

2.2.1. C-Bindings

```
OM_uint32
gss_set_cred_option_critical(OM_uint32 *minor_status,
                             gss_const_cred_id_t input_cred_handle,
                             gss_cred_id_t *output_cred_handle,
                             gss_const_OID desired_object,
                             gss_const_buffer_t value);
```

Figure 2: C-Bindings of GSS_Set_cred_option_critical()

2.3. Return Flag for Channel Binding State Signalling

Whenever both the initiator and the acceptor provide matching channel bindings to GSS_Init_sec_context() and GSS_Accept_sec_context(), respectively, then the mechanism SHALL indicate that the context is channel bound via an output flag for the established context.

See [Section 2.4](#) for details on when security context establishment may succeed in spite of only one of the two peers providing channel bindings.

2.3.1. C-Bindings

```
#define GSS_C_CHANNEL_BOUND 2048 /* 0x00000800 */
```

Figure 3: C-Bindings of channel_bound_flag

2.4. Credential Option For Channel Binding Negotiation Semantics Selection

Whenever the acceptor shall not have set GSS_C_CHANNEL_BOUND_CRED_OPT_OID and the acceptor has provided channel bindings, then the mechanism MUST NOT allow security context establishment to succeed when the initiator has not itself provided channel bindings. Whenever the acceptor shall not have set GSS_C_CHANNEL_BOUND_CRED_OPT_OID the mechanism SHOULD allow security context establishment to succeed when the acceptor has not itself provided channel bindings even if the initiator did.

Whenever the caller shall have set the GSS_C_CHANNEL_BOUND_CRED_OPT_OID (see below) then the mechanism SHOULD allow security context establishment to succeed even if one of the initiator or acceptor failed to provide channel bindings while the other did.

Williams
7]

Expires August 16, 2013

[Page

Note that mechanisms that cannot distinguish whether the peer did or did not provide channel bindings are likely to fail security context establishment when one party uses channel binding and the other does not, and that such mechanisms may have no way to implement the GSS_C_CHANNEL_BOUND_CRED_OPT_OID credential option. Such mechanisms SHOULD return GSS_S_UNAVAILABLE when this option is set on their credentials, but MAY pretend that they support it as a local configuration option. However, it is strongly RECOMMENDED that mechanisms have a way to distinguish (in their security context tokens) the use/non-use of channel bindings.

2.4.1. C-Bindings

```
gss_const_OID GSS_C_CHANNEL_BOUND_CRED_OPT_OID; /* OID TBD */
```

Figure 4: C-Bindings of GSS_C_CHANNEL_BOUND_CRED_OPT_OID

Williams
8]

Expires August 16, 2013

[Page

3. References

3.1. 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.
- [RFC5056] Williams, N., "On the Use of Channel Bindings to Secure Channels", [RFC 5056](#), November 2007.
- [RFC5587] Williams, N., "Extended Generic Security Service Mechanism Inquiry APIs", [RFC 5587](#), July 2009.

3.2. Informative References

- [RFC4121] Zhu, L., Jaganathan, K., and S. Hartman, "The Kerberos Version 5 Generic Security Service Application Program Interface (GSS-API) Mechanism: Version 2", [RFC 4121](#), July 2005.

Williams
9]

Expires August 16, 2013

[Page

Internet-Draft
2013

GSS Channel Bound Flag

February

Author's Address

Nicolas Williams
Cryptonector, LLC

Email: nico@cryptonector.com

