

Network Working Group
INTERNET-DRAFT
IETF Common Authentication Technology WG
<[draft-ietf-cat-fipsjjjgss-00.txt](#)>

S. Murphy
D. Balenson
J. Galvin
TIS
July 1995

The FIPS PUB JJJ Entity Authentication GSS-API Mechanism

Status of this Memo

This document is an Internet Draft. 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 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''.

To learn the current status of any Internet Draft, please check the `1id-abstracts.txt` listing contained in one of the Internet-Drafts Shadow Directories on `ftp.is.co.za` (Africa), `nic.nordu.net` (Europe), `munni.oz.au` (Pacific Rim), `ds.internic.net` (US East Coast), or `ftp.isi.edu` (US West Coast).

Abstract

This specification defines protocols, procedures, and conventions to be employed by peers implementing the Generic Security Service Application Program Interface, as specified in RFCs 1508 and 1509 [2] [3], (and as amended by the Internet Draft defining GSS-API Version 2 [4]) when using the FIPS PUB JJJ class of authentication mechanisms (as defined in FIPS PUB JJJ [1]).

1. Acknowledgements

Some of the material in this document, particularly Sections 2 and 5, is based on the Simple Public-Key GSS-API Mechanism (SPKM) specification [5]. The authors of this document thank Warwick Ford, Paul Van Oorschot, and Carlisle Adams of Bell-Northern Research, John Linn and Marc Horowitz of OpenVision Technologies, and Joan Marshall for many fruitful

Internet Draft

FIPS PUB JJJ GSS-API Mechanism

July 1995

discussions.

2. Introduction

FIPS PUB JJJ, recently proposed as a Federal Information Processing Standard (FIPS) [8], defines a class of mechanisms employing digital signatures to provide either unilateral and mutual authentication using public key cryptography. It is based primarily on a subset of ISO Standard 9798 [6].

The goal of the Generic Security Service Application Program Interface (GSS-API) is stated in the abstract of [RFC 1508](#) [2] as follows:

"This Generic Security Service Application Program Interface (GSS-API) definition provides security services to callers in a generic fashion, supportable with a range of underlying mechanisms and technologies and hence allowing source-level portability of applications to different environments. This specification defines GSS-API services and primitives at a level independent of underlying mechanism and programming language environment, and is to be complemented by other, related specifications:

- documents defining specific parameter bindings for particular language environments;
- documents defining token formats, protocols, and procedures to be implemented in order to realize GSS-API services atop particular security mechanisms."

Each mechanism in the FIPS PUB JJJ class of authentication mechanisms is an instance of the latter type of document and is therefore termed a "GSS-API Mechanism". If a FIPS PUB JJJ class mechanism conforms to the interface defined by [RFC 1508](#) [2], it can be used as a drop-in replacement by any application which makes use of security services through GSS-API calls (for example, any application which already uses the Kerberos GSS-API for security).

The tokens defined in this document are intended to be used by application programs according to the GSS API "operational paradigm" (see [RFC 1508](#) [2] for further details):

The operational paradigm in which GSS-API operates is as follows. A typical GSS-API caller is itself a communications protocol [or is an application program which uses a communications protocol], calling on GSS-API in order to protect its communications with authentication, integrity, and/or confidentiality security services. A GSS-API caller accepts tokens provided to it by its

local GSS-API implementation [i.e., its GSS-API mechanism] and transfers the tokens to a peer on a remote system; that peer

passes the received tokens to its local GSS-API implementation for processing.

[3.](#) Related Work

FIPS PUB JJJ defines unilateral and mutual authentication mechanisms based on ISO Standard 9798. There is an IETF effort presently ongoing called Simple Public Key Mechanism [\[5\]](#) that defines unilateral and mutual authentication mechanisms based on CCITT Standard X.509 [\[7\]](#). The two standards are extremely close in intent and technique. SPKM includes more than just authentication; it also includes provisions for confidentiality and integrity protection of the messages transmitted through the established context. FIPS PUB JJJ provides authentication, but no protection of the resultant context. The possibility that FIPS PUB JJJ and the SPKM work could be combined should be pursued.

[4.](#) GSS-API Interface Calls

As FIPS PUB JJJ deals only with authentication, we define the implementation of only the GSS-API context establishment calls for a FIPS PUB JJJ class mechanism. We presume that some other negotiation, either out of band, by the calling protocol, or by some as yet undefined GSS-API negotiation mechanism, has determined that a FIPS PUB JJJ mechanism will be used, or that a FIPS PUB JJJ mechanism is the local default for the initiator and target.

[4.1.](#) GSS_Init_sec_context call

FIPS PUB JJJ defines authentication mechanisms for both unilateral and mutual authentication. Unilateral authentication authenticates the initiator to the target. There may be mechanisms that can support only unilateral authentication or support only mutual authentication or that can support both. As there is no reason to believe that any one mechanism in the FIPS PUB JJJ class can do both types of authentication, one duty of the GSS-API interface calls will be to negotiate the use of mutual or unilateral authentication.

As FIPS PUB JJJ defines a class of authentication mechanisms, it is possible that the initiator's and target's individual mechanisms may have different sets of digital signature algorithms available. A negotiation for algorithm type will proceed concurrently with the negotiation of use of unilateral or mutual authentication.

[4.1.1.](#) First call to GSS_Init_sec_context

On the initial call to GSS_Init_sec_context, the initiator creates a FIPS_SETUP token containing the set of representations for the digital signature algorithms available locally, as well as flags indicating whether mutual and/or unilateral authentication is available and whether mutual or unilateral authentication is requested in the GSS_Init_sec_context call. The services, i.e., delegation, sequencing, etc., requested in the GSS_Init_sec_context call may optionally be included in the token. (Different FIPS PUB JJJ class mechanisms may have some or all of these services available; the negotiation for those services is not specified here.) GSS_S_CONTINUE_NEEDED is returned as the major_status code and FIPS_SETUP is returned as output_token. The mech_type on entry must be FIPS_PUB_JJJ or null, and the returned value must be FIPS_PUB_JJJ.

[4.1.2.](#) Second call to GSS_Init_sec_context

A second call to GSS_Init_sec_context must pass an input_token that is either FIPS_ERROR, indicating an error, or FIPS_CHALN, containing the final negotiated type of authentication, a subset of the initially offered digital signature algorithm set indicating the algorithms acceptable to the target and the FIPS PUB JJJ challenge from the target.

If the input_token is FIPS_ERROR, the major_status return code must be GSS_S_FAILURE, with a minor_status code chosen to represent the error indicated in the FIPS_ERROR token. No context is established.

If the input_token is FIPS_CHALN, then the initiator must pick one of the set of digital signature algorithms indicated in the alg_subset field to be used for the remainder of the authentication. The algorithm

choice is included in the output_token.

If unilateral authentication is indicated in the FIPS_CHALN, the initiator must form a response to the challenge according to FIPS PUB JJJ using the chosen digital signature algorithm and include the response in the output_token. The output_token will be FIPS_REP_IT, the major_status returned will be GSS_S_COMPLETE and a context should be established. The values returned for deleg_state, mutual_state, replay_det_state, sequence_state, and anon_state must take their values from the FIPS_CHALN options field, if present.

If mutual authentication is indicated in the FIPS_CHALN, the initiator must form a response to the challenge (using the chosen digital signature algorithm) and a new challenge both according to FIPS PUB JJJ

and include the response and new challenge in the output_token. The output_token will be FIPS_REPCHALN_IT and the major_status returned will be GSS_S_CONTINUE_NEEDED.

[4.1.3.](#) Third call to GSS_Init_sec_context

A third call to GSS_Init_sec_context, necessary only if mutual authentication is agreed upon, will pass an input_token that is either FIPS_ERROR, indicating an error, or FIPS_REP_TI, containing the target's response to the challenge from the initiator found in FIPS_REPCHALN_IT.

If the input_token is FIPS_ERROR, the major_status return code must be GSS_S_FAILURE, as discussed above.

If the input_token is FIPS_REP_TI and the response to the challenge fails verification, then the major_status returned should be either GSS_S_BAD_SIG or GSS_S_FAILURE with an appropriate minor_status code, at the implementor's option. Note that it is the decision and responsibility of the caller to inform the other end of the connection that authentication has failed.

If the input_token is FIPS_REP_TI and the response to the challenge is verified, then the major_status returned must be GSS_S_COMPLETE and the context should be established. The values returned for deleg_state, mutual_state, replay_det_state, sequence_state, and anon_state must take

their values from the FIPS_REP_TI options field, if present.

The second and third calls to GSS_Init_sec_context can be distinguished from the first by the fact that the input_token and input_context_handle must be null in the first call and not null in the subsequent calls. The second and third calls to GSS_Init_sec_context can be distinguished by the token_ID's in the input_token, except in the case that the input_token is FIPS_ERROR. FIPS_ERROR is a permissible parameter for both the second and the third GSS_Init_sec_context calls. As the processing in both cases is the same, this should not be a problem.

[4.2.](#) GSS_Accept_sec_context call

The GSS_Accept_sec_context call will always be called twice, whether unilateral or mutual authentication is used in the context establishment.

[4.2.1.](#) First call to GSS_Accept_sec_context

The call to GSS_Accept_sec_context will pass an input_token. If this is the first call to GSS_Accept_sec_context, the token must be FIPS_SETUP. The token contains the availability of unilateral and mutual authentication at the initiator, as well as the caller's request for unilateral or mutual authentication, and an indication of the set of digital signature algorithms the initiator has available.

The target constructs the subset of the offered set of digital signature algorithms that it has available. If this subset is empty, then the initiator and target have no common digital signature algorithms. The target must return a GSS_S_FAILURE major_status code and a FIPS_ERROR token containing the error code 0001. No context will be established. If this subset is not empty, then the subset must be included in the alg_subset field of the FIPS_CHALN token.

If the initiator can do only unilateral authentication and the target can do only mutual authentication or vice versa, then no context can be

established. The token returned will be FIPS_ERROR, with an error code of 0002 and a major_status of GSS_S_FAILURE. If the initiator and target can both do unilateral or both do mutual authentication, but not both do both, then the common type must be specified in the returned FIPS_CHALN token by setting the value of the mutual_tobeused field to true (for mutual) or false (for unilateral). If both initiator and target can both do both types of authentication, then the value of the mutual_tobeused field in the returned FIPS_CHALN token must be set to the value of the mutual_req field in the FIPS_SETUP token. In other words, if the initiator and target can both do both types of authentication, then the initiator's caller's request determines the type.

If no error was determined, then a challenge must be formed (using the chosen digital signature algorithm) according to FIPS PUB JJJ and included in the output_token. The first call to GSS_Accept_sec_context must return with a GSS_S_CONTINUE_NEEDED major_status code and FIPS_CHALN as the output_token.

4.2.2. Second call to GSS_Accept_sec_context

The second call to GSS_Accept_sec_context can be distinguished by a non-null input_context_handle and by the token_ID in the input_token. The second call should pass an input_token containing either FIPS_ERROR or a response token. If the authentication type was determined on the first call to be unilateral authentication, then the response token must

be FIPS_REP_IT; if it was mutual, then the response token must be FIPS_REPCHALN_IT.

The response token contains the response to the target's challenge. The response must be verified according to FIPS PUB JJJ. If the verification fails and mutual authentication was chosen, then a FIPS_ERROR token must be formed with an error code of 0003. The major_status returned should be either GSS_S_BAD_SIG or GSS_S_FAILURE with an appropriate minor_status code, at the implementor's option. Note that if unilateral authentication is chosen, receipt of a FIPS_REP_IT token means that the other side of the connection has completed context establishment. Therefore, it is the decision and responsibility of the caller to inform the other side of the connection

that authentication has failed.

If the response to the challenge is verified, then the `major_status` returned must be `GSS_S_COMPLETE` and the context must be established. If the type of authentication is mutual, then the `input_token` was `FIPS_REPCHALN_IT` and contains a challenge. A response to this challenge must be formed according to FIPS PUB JJJ and included in a `FIPS_REP_TI` `output_token`. In either case, the values returned for `deleg_state`, `mutual_state`, `replay_det_state`, `sequence_state`, and `anon_state` must take their values from the `input_token`, if present.

5. Tokens

Three classes of tokens are defined in this section: "Initiator" tokens, emitted by calls to `GSS_Init_sec_context()` and consumed by calls to `GSS_Accept_sec_context()`; "Target" tokens, emitted by calls to `GSS_Accept_sec_context()` and consumed by calls to `GSS_Init_sec_context()`; and "Error" tokens, potentially emitted by calls to `GSS_Init_sec_context()` and `GSS_Accept_sec_context()`, and potentially consumed by calls to `GSS_Init_sec_context()` and `GSS_Accept_sec_context()`. The "Initiator" tokens are `FIPS_SETUP`, `FIPS_REP_IT`, and `FIPS_REPCHALN_IT`. The "Target" tokens are `FIPS_CHALN` and `FIPS_REP_TI`. The "Error" token is `FIPS_ERROR`.

Per [RFC-1508, Appendix B](#), the initial context establishment token will be enclosed within framing as follows:

```
InitialContextToken ::= [APPLICATION 0] IMPLICIT SEQUENCE {
    thisMech          MechType
        -- MechType is OBJECT IDENTIFIER
        -- representing "FIPS PUB JJJ"
    innerContextToken ANY DEFINED BY thisMech
}
```

```
        -- contents mechanism-specific
    }
```

The above GSS-API framing shall be applied to all tokens emitted by the FIPS PUB JJJ GSS-API mechanism, `FIPS_CHALN` (the challenge from the Target to the Initiator) `FIPS_REP_IT` (the response from the Initiator to the Target for unilateral authentication), `FIPS_REPCHALN_IT` (the

response from the Initiator to the Target for mutual authentication), FIPS_REP_TI (the response from the Target to the Initiator for mutual authentication), and FIPS_ERROR, not just to the initial token (FIPS_SETUP) in a context establishment sequence. While not required by [RFC-1508](#), this enables implementations to perform enhanced error-checking. Furthermore, every innerContextToken will contain a token-id followed by a context-id.

The innerContextToken field of context establishment tokens for the FIPS PUB JJJ GSS-API mechanism will consist of a token (FIPS_SETUP, FIPS_CHALN, FIPS_REP_IT, FIPS_REPCHALN_IT, FIPS_REP_TI or FIPS_ERROR) containing a token_ID field having the value 0100 (hex) for FIPS_SETUP tokens, 0200 (hex) for FIPS_CHALN tokens, 0300 (hex) for FIPS_REP_IT, [0400](#) (hex) for FIPS_REPCHALN_IT tokens, 0500 (hex) for FIPS_REP_TI tokens, and 0600 (hex) for FIPS_ERROR tokens. All innerContextToken's are encoded using Abstract Syntax Notation One Basic Encoding Rules (ASN.1 BER).

Each of the following tokens contains a token ID field. The FIPS tokens themselves have an optional token ID field. But because this token is an optional field, there is no guarantee that any particular mechanism in the FIPS PUB JJJ class will have implemented that field. It is therefore necessary to include such a field in the tokens here.

The FIPS tokens are defined in FIPS PUB JJJ, [Appendix A](#).

FIPS_SETUP ::= SEQUENCE {

token_ID	INTEGER,
context_ID	RandomNumber,
alg_choices	SEQUENCE OF AlgorithmIdentifier,
mutual_req	BOOLEAN,
mutual_avail	BOOLEAN,
unilateral_avail	BOOLEAN,
context	ContextData OPTIONAL

}

RandomNumber ::= INTEGER

ContextData ::= SEQUENCE {
 chan_bindings OCTET STRING OPTIONAL,
 options Options OPTIONAL
 }

Options ::= BIT STRING {
 deleg_req (0),
 replay_req (1),
 sequence_req (2),
 anon_req (3),
 lifetime_req (4)
 }

FIPS_CHALN ::= SEQUENCE {	
token_ID	INTEGER,
context_ID	RandomNumber,
alg_subset	SEQUENCE OF AlgorithmIdentifier,
mutual_tobeused	BOOLEAN,
context	ContextData OPTIONAL
fips_token	MessageBA
}	

```
FIPS_REP_IT ::= SEQUENCE {  
    token_ID          INTEGER,  
    context_ID        RandomNumber,  
    alg_choice        AlgorithmIdentifier,  
    context            ContextData OPTIONAL  
    fips_token        MessageABU  
}
```

```
FIPS_REPCHALN_IT ::= SEQUENCE {  
    token_ID          INTEGER,  
    context_ID        RandomNumber,  
    alg_choice        AlgorithmIdentifier,  
    context            ContextData OPTIONAL  
    fips_token        MessageABM  
}
```

```
FIPS_REP_TI ::= SEQUENCE {  
    token_ID          INTEGER,  
    context_ID        RandomNumber,  
    alg_choice        AlgorithmIdentifier,  
    context            ContextData OPTIONAL  
    fips_token        MessageBA2  
}
```

```
FIPS_ERROR ::= SEQUENCE {  
    token_ID          INTEGER,  
    context_ID        RandomNumber,  
    error_type        INTEGER  
}
```

Internet Draft

FIPS PUB JJJ GSS-API Mechanism

July 1995

Where the value of error_type will be

0001	No compatible authentication algorithm found
0002	No compatible authentication type (mutual/unilateral) found
0003	Verification of challenge response has failed.
0004	Invalid token format
0005	Invalid format for fips_token field inside token

The following definition is taken from X.509 - The Directory: Authentication Framework [7]

```
Algorithm Identifier ::= SEQUENCE {  
    algorithm          OBJECT IDENTIFIER,  
    parameter          ANY DEFINED BY algorithm OPTIONAL  
}
```

6. Security Considerations

This document deals with the security issue of public-key based entity authentication.

7. References

- [1] Standard for Public Key Cryptographic Entity Authentication Mechanisms (Draft), FIPS PUB JJJ, National Institute of Standards and Technology, March 13, 1995. (Available online at <http://csrc.ncsl.nist.gov/fips> as pkauth.txt and pkauth.ps.)
- [2] Linn, J., Generic Security Service Application Program Interface, [RFC 1508](#), September 10, 1993.
- [3] Wray, J., Generic Security Service API : C-bindings, RFC 1509, September 10, 1993.
- [4] Wray, J., Generic Security Service Application Program Interface, Version 2, Internet Draft (work in progress), <[draft-ietf-cat-gssv2-01.txt](#)>, March 1995.

- [5] Adams, C., The Simple Public-Key GSS-API Mechanism (SPKM), Internet Draft (work in progress), <[draft-ietf-cat-spkmgss-02.txt](#)>

Murphy/Balenson/Galvin Expires: January 1996

[Page 11]

Internet Draft

FIPS PUB JJJ GSS-API Mechanism

July 1995

- [6] ISO/IEC 9798-1, Information Technology - Security Technologies - Entity Authentication Mechanisms - Part 1: General model, 1991-09-01.
- [7] ITU-T Rec. X.509 | ISO/IEC 9594-8, Information Technology - Open Systems Interconnection - The Directory: Authentication Framework, Editor's DRAFT, February 14, 1993.
- [8] Public Key Cryptography Entity Authentication Mechanisms, Federal Register, Volume 60, Number 108, June 6, 1995, pp. 29749-29958.

[8.](#) Authors' Address

Sandra Murphy <murphy@tis.com>
Dave Balenson <balenson@tis.com>
Jim Galvin <galvin@tis.com>

Trusted Information Systems
[3060](#) Washington Road
Glenwood, MD 21738

Internet Draft

FIPS PUB JJJ GSS-API Mechanism

July 1995

Table of Contents

Status of this Memo	<u>1</u>
Abstract	<u>1</u>
<u>1</u> Acknowledgements	<u>1</u>
<u>2</u> Introduction	<u>2</u>
<u>3</u> Related Work	<u>3</u>
<u>4</u> GSS-API Interface Calls	<u>3</u>
<u>4.1</u> GSS_Init_sec_context call	<u>3</u>
<u>4.1.1</u> First call to GSS_Init_sec_context	<u>4</u>
<u>4.1.2</u> Second call to GSS_Init_sec_context	<u>4</u>
<u>4.1.3</u> Third call to GSS_Init_sec_context	<u>5</u>
<u>4.2</u> GSS_Accept_sec_context call	<u>5</u>
<u>4.2.1</u> First call to GSS_Accept_sec_context	<u>6</u>
<u>4.2.2</u> Second call to GSS_Accept_sec_context	<u>6</u>
<u>5</u> Tokens	<u>7</u>
<u>6</u> Security Considerations	<u>11</u>
<u>7</u> References	<u>11</u>
<u>8</u> Authors' Address	<u>12</u>

Murphy/Balenson/Galvin Expires: January 1996

[Page 13]