PEM-Based IDUP Mechanism (PIM)

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 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."

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

Comments on this document should be sent to "cat-ietf@mit.edu", the IETF Common Authentication Technology WG discussion list.

ABSTRACT

The Independent Data Unit Protection Generic Security Service Application Program Interface (IDUP-GSS-API) extends the GSS-API [RFC-1508] for applications requiring protection of a generic data unit (such as a file or message) in a way which is independent of the protection of any other data unit and independent of any concurrent contact with designated "receivers" of the data unit. Thus, it is suitable for applications such as secure electronic mail where data needs to be protected without any on-line connection with the intended recipient(s) of that data. Subsequent to being protected, the independent data unit can be transferred to the recipient(s) - or to an archive - perhaps to be processed only days or years later.

This document is a companion document to IDUP-GSS-API [IDUP] and IDUP: C-bindings [IDUP-C]. It provides a PEM-based [RFC-1421] mechanism for IDUP (analogous to [Kerb5] or [SPKM] which provide underlying mechanisms for [GSS]). This mechanism specifies both encapsulation and non-encapsulation procedures for creating PEM messages. If encapsulation is chosen by the calling application, then the PIM implementation creates true PEM messages. On the other hand, if encapsulation is not chosen, then the PIM implementation performs the canonicalization, security, and encoding (see [RFC-1421]) aspects of PEM, returning to the caller a true PEM header and transportable data; applications wishing to build PEM messages must then concatenate the header and the encoded IDU with the necessary delimiters and CRLFs to complete PEM message construction.
1. INTRODUCTION

The Independent Data Unit Protection Generic Security Service Application Program Interface (IDUP-GSS-API) [IDUP] provides security services to calling applications in a local environment. This PEM-Based IDUP Mechanism (PIM) allows an application to "protect" an independent data unit (IDU) for future use and to "unprotect" a protected IDU, applying security services such as confidentiality, integrity and data origin authentication on a per-data-unit basis.

There are four stages to using the IDUP-GSS-API:

(a) The application acquires a set of credentials with which it may bind its identity to a data unit. The application's credentials vouch for its global identity, which may or may not be related to the local username under which it is running.

(b) The application establishes a security environment using its credentials. The security environment contains whatever information is necessary in order to provide per-IDU security services.

(c) Per-IDU calls are invoked to provide one or both of the following for PEM-based IDU protection:
   - data origin authentication with data integrity;
   - data confidentiality.

The application wishing to "protect" an IDU will call the protection set of IDUP-GSS-API routines, specifying the appropriate security environment. The recipient (wishing to "unprotect" the data) will pass the protected IDU (P-IDU) to the unprotection set of routines to remove the protection and/or to validate the data.

(d) At the completion of a security environment (which may extend across several protection and unprotection operations), the application calls an IDUP-GSS-API routine to abolish the security environment.

2. INDEPENDENT DATA UNIT PROTECTION MECHANISM

2.1. Protection Token

A P-IDU is a caller-opaque data structure that PIM uses to store protection information regarding an IDU. It is an OCTET STRING generated during the protection set of calls for use by PIM or by another mechanism during the unprotection set of calls. If encapsulation is requested by the calling application, then the P-IDU consists entirely of the contents of the pidu_buffer, output_buffer, and final_pidu_buffer parameters used in the IDUP protection set of calls. If encapsulation is not used, the P-IDU consists of the logical concatenation of the unencapsulated_token parameter in each Prot_Service parameter bundle, the contents of the midu_buffer, output_buffer, and final_midu_buffer parameters, and other PEM-specific delimiter values (see Section 4.1.1 below).
2.2. Security Services

PIM makes use of the security services given in Section 1(c) above. The security services "proof of origin" and "service solicitation" (such as a request for "proof of delivery"), as defined in [IDUP], are not included in this PEM-based IDUP mechanism. Receipt generation and processing are beyond the scope of [RFC-1421] and other types of non-repudiable evidence generation and processing are not addressed in this version of PIM but may be included in future versions of this specification.

2.3. IDUP Parameter Bundle Uses and Defaults

The following parameter bundle uses and defaults are specified.

Mech_Specific_Info
  - NOT USED (the only acceptable input, therefore, is NULL)

Idu_Sensitivity
  - NOT USED (the only acceptable input, therefore, is NULL)

Service_Creation_Info
  - NOT USED (the only acceptable input, therefore, is NULL)

Service_Verification_Info
  - NOT USED (the only acceptable input, therefore, is NULL)

Quality
  - the qop_algs parameter is supported. For PIM implementation and interoperability purposes, the following qop_algs values are defined.
    * For the Confidentiality MA field: 0001 (1) = DES-CBC
      (this is also defined to be the TS "DEFAULT" conf. alg.).
    * For the Integrity MA field: 0001 (1) = RSA-MD5
      (this is also defined to be the TS "DEFAULT" int. alg.).
  - the parameters validity, policy_id, and allow_policy_mapping are NOT USED (NULLs are therefore the only acceptable input)

Idu_Information
  - the idu_type parameter must have a value representing "RFC822"
or any other valid PEM "Content-Domain" (the DEFAULT value is specified to be "RFC822");
  - the idu_title parameter is NOT USED (the only acceptable input, therefore, is NULL)
Prot_Information
- the originator_name and idu_type (in Idu_Information) parameters are read from the PEM header information and are output by IDUP_Start_Unprotect. The originator_name parameter shall be formatted as follows (specified in the descriptive grammar BNF; see [RFC-822] and Section 4.1 below):
  <originator-name> ::= [<entity-id>] CRLF [<issuing-auth>] CRLF [<version-expiration>] CRLF
  <entity-id> ::= 1* (<ia-char> / ",")
  <issuing-auth> ::= 1* (<ia-char> / ",")
  <version-expiration> ::= 1* (<ia-char> / ",")

The actual values for <originator-name> may be read directly from the P-IDU (if <origid-symm> or <origid-asymm> are present) or may be extracted from the originator’s certificate (if <cert> is present).
- all other parameters are NOT USED (and therefore NULL)

Special_Conditions
- NOT USED (the only acceptable input, therefore, is NULL)

Target_Info
- this bundle is used as described in IDUP; no DEFAULT values are specified

General_Service_Data
- the unencapsulated_token parameter is used if encapsulation_request is FALSE;
- the minor_status parameter is used to return minor status values as specified in Section 5 below.

Prot_Service
- the prot_service_type parameter may have a value of "1" ("perform unsolicited service") or NULL (which specifies the DEFAULT value of "1");
- the service_id parameter must have a value representing "PER_CONF" or "PER_DOA";
- the parameters Service_Creation_Info, service_to, Service_Verification_Info, and service_verification_info_id are NOT USED (and therefore NULL)

Unprot_Service
- the unprot_service_type parameter will always have a value of "1" ("receive unsolicited service");
- the service_id parameter will have a value representing "REC_CONF" or "REC_DOA";
- the parameters service_verification_info_id, Service_Verification_Info, service_to, and Service_Creation_Info, are NOT USED (and therefore NULL)
3. SUMMARY OF TRANSFORMATIONS

If confidentiality is applied during the IDUP protection set of calls, then the following composition of transformations is applied to the IDU for full PEM compliance; see [RFC-1421] for definitions of "encode" and "canonicalize":

\[
\text{Transmit\_Form} = \text{Encode}(\text{Encrypt}(\text{Canonicalize}(\text{Local\_Form})))
\]

The inverse transformations are performed, in reverse order, to unprotect the IDU:

\[
\text{Local\_Form} = \text{DeCanonicalize}(\text{Decipher}(\text{Decode}(\text{Transmit\_Form}))).
\]

If confidentiality is not applied (i.e., data origin authentication with data integrity alone is applied), then the IDU goes through the following operation:

\[
\text{Transmit\_Form} = \text{Canonicalize}(\text{Local\_Form})
\]

Again, this needs to be inverted by the receiver:

\[
\text{Local\_Form} = \text{DeCanonicalize}(\text{Transmit\_Form}).
\]

It is the responsibility of the underlying PIM implementation to perform all transformations given in this section (regardless of whether or not encapsulation is requested by the calling application) unless the transformation is unnecessary (e.g., the input data is already in canonical form).

Note that the Local\_Form and the functions to transform messages to and from Canonical\_Form may vary between the protector and unprotector systems provided there is no loss of information.

4. TOKEN FORMAT

This section discusses protocol-visible characteristics of PIM; it defines elements of protocol for interoperability and is independent of any IDUP language bindings.

The PIM IDUP-GSS-API mechanism will be identified by an Object Identifier representing "PIM", having the value:

\{
iso(1) org(3) dod(5) internet(1) security(5) PIM(xx) \}

The token transferred between IDUP-GSS-API peers (for IDU protection and unprotection purposes) is defined.
4.1. The Protected-IDU (P-IDU) Token

The OCTET STRING "protpart" conforms to the Privacy Enhanced Mail header format described in [RFC-1421]. The Protected-IDU (P-IDU) is the concatenation of protpart and the "idupart" (which may be the canonicalized, encoded version of the original IDU or, if confidentiality was applied, the encrypted IDU). If encapsulation is requested by the calling application, this concatenation is handled entirely by the PIM implementation and the application needs to deal only with the resulting pidu_buffer output parameters from the protection set of calls.

If encapsulation is not requested, the calling application (not the PIM implementation) is responsible for the creation of the P-IDU from the protpart and the idupart. In this case, the protpart is supplied by PIM in the unencapsulated_token parameter of one of the Prot_Service parameter bundles (either bundle can be chosen if both PER_CONF and PER_DOA are requested) and idupart is supplied in the midu_buffer parameters. For example, to create a true PEM message, the application would concatenate the pre-Encapsulation-Boundary delimiter ("-----BEGIN PRIVACY-ENHANCED MESSAGE-----"), CRLF, protpart, CRLF, idupart, the post-Encapsulation-Boundary delimiter ("-----END PRIVACY-ENHANCED MESSAGE-----"), and a final CRLF. (Note that the PIM implementation must perform canonicalization and message encoding (as per [RFC-1421], Section 4.3.2.4), as required, as part of the protection process.)

In this section both idupart and protpart are specified in the descriptive grammar BNF.

; PIM BNF representation, using RFC-822 notation [RFC-822].
; Imports field meta-syntax (field, field-name, field-body, field-body-contents) from RFC-822.
; Imports DIGIT, ALPHA, text, CRLF from RFC-822.

<idupart> ::= 1*<binchar>       ; for ENCRYPTED msgs (enc. IDU) / 1*(<text> CRLF) ; for MIC-CLEAR msgs (orig. IDU)
<binchar> = <any binary character> ; (0-377 (octal), 0-255 (decimal))

<protpart> ::= <normalhdr>
<normalhdr> ::= <proctype> <contentdomain> [dekinfo] ; needed if ENCRYPTED

(1*<origflds> *<recipflds>)) ; symmetric case
; recipflds included for all proc types / ((1*<origflds>) *(<recipflds>)) ; asymmetric case
; recipflds included for ENCRYPTED proc type

<origflds> ::= <asymmorig> [keyinfo] *(issuercert) <micinfo> ; asymmetric
/ <origid-symm> [keyinfo] ; symmetric
<recipflds> ::= <recipid> <keyinfo>
<asymmorig> ::= <origid-asymm> / <cert>
; definitions for PEM header fields
<proctype> ::= "Proc-Type" ":" "4" "," <pimtypes> CRLF
<pimtypes> ::= "ENCRIPTED" / "MIC-CLEAR"
<contentdomain> ::= "Content-Domain" ":" <contentdescrip> CRLF
<contentdescrip> ::= "RFC822"
; This specification defines one value ("RFC822") for
; <contentdescrip>: other values may be defined in future in
; separate or successor documents
<dekinfo> ::= "DEK-Info" ":" <dekalgid> ["," <dekparameters>] CRLF
<origid-symm> ::= "Originator-ID-Symmetric" ":" <symmid> CRLF
<origid-asymm> ::= "Originator-ID-Asymmetric" ":" <asymmid> CRLF
<cert> ::= "Originator-Certificate" ":" <encbin> CRLF
<symmid> ::= <IKsubfld> ["," <IKsubfld>] ["", [<IKsubfld>]]
<asymmid> ::= <IKsubfld> "," <IKsubfld>
<br> ::= 1*<ia-char>
; Note: "," removed from <ia-char> set so that Orig-ID and Recip-ID
; fields can be delimited with commas (not colons) like all other
; fields
<ia-char> ::= DIGIT / ALPHA / "/" / "+" / "(" / ")" / 
"-" / "/" / ":" / "@" / ";" / ":" / "=" / ":" / "<" / ">
<micinfo> ::= "MIC-Info" ":" <micalgid> "," <ikalgid> "," <asymsignmic> CRLF
<issuercert> ::= "Issuer-Certificate" ":" <encbin> CRLF
<encbin> ::= 1*<encbinrp>
<encbinrp> ::= 4*4<encbinchar>
<encbinchar> ::= ALPHA / DIGIT / "+" / "/" / ":="
<br> ::= <recipid-symm> / <recipid-asymm>
<recipid-symm> ::= "Recipient-ID-Symmetric" ":" <symmid> CRLF
<recipid-asymm> ::= "Recipient-ID-Asymmetric" ":" <asymmid> CRLF
<br> ::= "Key-Info" ":" <ikalgid> "," <micalgid> "," <symencdek> "," <symencmic> CRLF
; symmetric case
/ "Key-Info" ":" <ikalgid> "," <asymencdek> CRLF ; asymmetric case
<encbinbody> ::= *(16*16<encbinrp> CRLF) [1*16<encbinrp> CRLF]

; The following items are defined in [RFC-1423] and are meant to
; serve as RECOMMENDED algorithms for PIM implementation
; interoperability purposes. It is recognized that this list may
; be altered in future versions of this specification and may grow
; over time.
4.2. Examples of Protection Tokens

4.2.1 Integrity Only (Example)

protpart = {
  Proc-Type: 4,MIC-CLEAR
  Content-Domain: RFC822
  Originator-Certificate:
    MIIBITCCAscCAwUwQJJoZIhcNQECBQAwUfTELMKAg1UEBoMCVVMxIDAeBgNV
    BAoTF1JTSBEYHRIFSNY3vyr5LCBjmMuMQ8wDQYDVQQLEw2CZXRhIDExDAN
    BgNVsAsTbKBk5PEFSTaewFw05MTA5MDQxODM4TdaFw05mzaMDmXMODMTaZAMEUX
    CzAJBqNVBYA1VTMSAwqHDVQQKExdSU0EgRGF0YSBTZWN1cm10eSwgSw5ljEU
    MBIGAEsAXMLVGVzdCBVc2VyIDEwWTABgRVCAEBAgICANLADBIAkEwZHl2I+y
    JcqcTjdJcOwnqDBjrdAIN5C+CntOjOELyuQbgGrglhhj3j8/x0FM+YrsyFlu3F
    LzPvztzlndhYFQIDAQABMA0GCSqGSIb3DQEEBGAIAkACKr0PghjYw7jYpTcIq
    c1glWFPU5njg9Kh7g7ASFxskvYEMRJNZV/H2D2QeHtVaU7JxfsSwf5x6yMpf23U/
    5UXUGx7qusDGHQgs7j9kW8Cn1fu5SwUGn4w==
    Issuer-Certificate:
      MIIB3DCAUuCQowQJJoZIhcNQECBQAwUfTELMKAg1UEBoMCVVMxIDAeBgNV
      BAoTF1JTSBEYHRIFSNY3vyr5LCBjmMuMQ8wDQYDVQQLEw2CZXRhIDExDAN
      BgNVsAsTbKBk5PEFSTaewFw05MTA5MDQxODM4TdaFw05mzaMDmXMODMTaZAMEUX
      CzAJBqNVBYA1VTMSAwqHDVQQKExdSU0EgRGF0YSBTZWN1cm10eSwgSw5ljEU
      MBIGAEsAXMLVGVzdCBVc2VyIDEwWTABgRVCAEBAgICANLADBIAkEwZHl2I+y
      JcqcTjdJcOwnqDBjrdAIN5C+CntOjOELyuQbgGrglhhj3j8/x0FM+YrsyFlu3F
      LzPvztzlndhYFQIDAQABMA0GCSqGSIb3DQEEBGAIAkACKr0PghjYw7jYpTcIq
      c1glWFPU5njg9Kh7g7ASFxskvYEMRJNZV/H2D2QeHtVaU7JxfsSwf5x6yMpf23U/
      5UXUGx7qusDGHQgs7j9kW8Cn1fu5SwUGn4w==
    MIC-Info: RSA-MD5,RSA,
    jV2OfH+nnXHU8bnL8KPaad/mS1TDZ1bbVuxvZAOVRZ5q5+Ej15bQvqNeqOUNqjr6
    EtE7K2DcemgCycxj8dJ8a==
}

idupart = {
  This is the (canonicalized) plaintext message part.
}

4.2.2 Confidentiality-Only (Example)

protpart = {
    Proc-Type: 4, ENCRYPTED
    Content-Domain: RFC822
    DEK-Info: DES-CBC,F8143EDE5960C597
    Originator-ID-Symmetric: john.doe@abc.com,
    Recipient-ID-Symmetric: john.smith@xyz.com,ptf-kmc,3
    Key-Info: DES-ECB, RSA-MD2, 9FD3AAD2F2691B9A,
              B70665BB9BF7CBCDA60195DB94F727D3
    Recipient-ID-Symmetric: rpm@ghi.com, ptf-kmc,4
    Key-Info: DES-ECB, RSA-MD2, 161A3F75DC82EF26,
              E2EF532C65CBCFF79F83A2658132DB47
}

idupart = {
    [This would contain canonicalized, encrypted, PEM-encoded data.]
}

4.2.3. Integrity And Confidentiality (Example)

protpart = {
    Proc-Type: 4, ENCRYPTED
    Content-Domain: RFC822
    DEK-Info: DES-CBC,93652FD82C1A4202
    Originator-Certificate:
        MIIBXDCCAQwCBAUSVXQwBwYFkw4DAgMWGzEALMAKGA1UDEhMCQ0ExDDAKBgNVBAoT
        A0JOUjFqIF5NTx4MjAxMm22MTk2M0w1MDwMNjEzKzA0MDEwMigcDML
        MAkGA1UEBhMCQ0ExDDAKBgNVBAoT
        Originator-ID-Symmetric: john.doe@abc.com,
        Recipient-ID-Symmetric: john.smith@xyz.com, rpm@ghi.com, ptf-kmc,3
        Key-Info: DES-ECB, RSA-MD2, 9FD3AAD2F2691B9A,
                  B70665BB9BF7CBCDA60195DB94F727D3
        Recipient-ID-Symmetric: rpm@ghi.com, ptf-kmc,4
        Key-Info: DES-ECB, RSA-MD2, 161A3F75DC82EF26,
                  E2EF532C65CBCFF79F83A2658132DB47
}

idupart = {
    [This would contain canonicalized, encrypted, PEM-encoded data.]
}
5. MINOR STATUS CODES

No minor status codes have yet been defined for PIM.

6. SECURITY CONSIDERATIONS

Security issues are discussed throughout this memo.

7. REFERENCES


8. AUTHOR'S ADDRESS

Carlisle Adams
Bell-Northern Research, Ltd.
P.O.Box 3511, Station C
Ottawa, Ontario, CANADA K1Y 4H7
Phone: +1 (613) 763-9008
E-mail: cadams@bnr.ca