# Independent Data Unit Protection GSS API (IDUP)

## – 02.txt UPDATE –

Carlisle M. Adams
Bell-Northern Research
Ottawa, Canada

July 17-21, 1995
33rd Internet Engineering Task Force
Stockholm, Sweden



## Outline

- Background
  - Generic Security Service (GSS) [RFC 1508]
- IDUP Motivation and Goals
- Terminology / Concepts
- Function Calls
- Conclusions



## Background

- Generic Security Service (GSS)
  - an interface specification
  - defines a set of calls for services which applications developers can use to incorporate security
  - although "generic", GSS-API is targeted toward session-based (on-line) application peers
  - a security context is established between a pair of communicating applications
    - » both parties are required in order to estab. context
    - » message protection/unprotection has no meaning outside context



## Outline

- Background
  - Generic Security Service (GSS) [RFC 1508]
- ⇨ IDUP Motivation and Goals
- Terminology / Concepts
- Function Calls
- Conclusions



## IDUP-GSS

- Motivation:
  - many important applications are not session-based (e.g., e-mail)
  - need to protect a block of data independently of any other data
  - need to protect data independently of any on-line communication with target(s) of data
- Goals:
  - an extension / companion to the current GSS
  - want to re-use as much of GSS as possible
  - want to imitate the functionality of gss_sign / getMIC (but with confidentiality option), for "off-line" environments



## Outline

- Background
  - Generic Security Service (GSS) [RFC 1508]
- IDUP Motivation and Goals
- ⇨ Terminology / Concepts
- Function Calls
- Conclusions

## Terminology / Concepts

- "Environment"
  - different from GSS "context"
  - security environment is not shared on-line with peer
    - » rather, "environment" marks beginning to end of *local* IDU processing operations
    - » expiration time of environment is tied to credential



### Terminology / Concepts (cont'd)

- "Protection"
  - protection operations yield a prot_token
  - IDU is either unchanged or is encrypted, but is not encapsulated into a token in any way (*cf* gss_sign)
  - IDU + prot_token = "P-IDU"
- "Unprotection"
  - P-IDU is unprotected to decrypt and/or verify IDU
  - unprotection operations may yield a receipt_request_state flag (signals whether receipt generation call can be made)
- IDUP mechanisms may define multiple stages of protection and unprotection processing



### Terminology / Concepts (cont'd)

- Protection / Unprotection buffer sizes
  - IDU (or P-IDU) can be processed one buffer at a time
  - no synchronization is implied or expected between
    - » buffer size used for protection
    - » buffer size used for unprotection
    - » block sizes used for integrity or confidentiality



### Terminology / Concepts (cont'd)

- "Receipts"
  - entirely opaque to application
  - contain info. necessary to convince IDUP mechanism that P-IDU was received (i.e., contents are mechanism specific)
  - receiver can choose to generate / send receipt
- "Evidence"
  - IDUP allows *support* for non-repudiation
  - some mechanisms may actually wish to *do* non-repudiation
  - IDUP framework accommodates this, but actual mechanics of evidence generation/processing are mechanism specific



## Outline

- Background
  - Generic Security Service (GSS) [RFC 1508]
- IDUP Motivation and Goals
- Terminology / Concepts
- ⇨ Function Calls
- Conclusions



## Function Calls

- Environment management calls *(changed)*:
  - IDUP_Establish_environment()
  - IDUP_Inquire_environment()
    - » mech_type, ret_flags (conf, int, receipt, evidence)
  - IDUP_Abolish_environment()
- No "initiator" / "target" distinction at this level
- All subsequent calls (e.g., Protection and Unprotection) done in environment
- ★ Environment lifetime tied to credential
- Env. not tied to target information or IDU

## Function Calls (cont'd)

- **"Per-IDU" calls *(changed)*:**
  - **IDUP_Protect()**
  - **IDUP_Unprotect()**
  - **each call split into three parts**
    - » **Start_Operation() -- to identify target(s) / orig., QOP**
    - » **Operation() -- called once per buffer of input data**
    - » **End_Operation() -- called to return results**
  - **possible result tokens from End_Protect()**
    - » **prot_token -- contains info. req'd. for target to unprot.**
    - » **receipt_check -- contains info. to be matched against in-coming receipt**
  - **no receipt_token from End_Unprotect()**



## Function Calls (cont'd)

- **IDUP_Generate_Receipt()** ***(new)***
  - **Start(), Generate(), End() if required by mechanism**
- **IDUP_Process_receipt()** ***(changed)***
  - **validates receipt_token using receipt_check**
  - **returns COMPLETE or error, along with (rec.) originator info.**
- **IDUP_Generate_Evidence()** ***(new)***
  - **input gen_evidence_qop, prot_token**
  - **output evidence_token**
- **IDUP_Process_Evidence()** ***(new)***
  - **input proc_evidence_qop, evidence_token**
  - **output prot_token**



## Outline

- **Background**
  - **Generic Security Service (GSS) [RFC 1508]**
- **IDUP Motivation and Goals**
- **Terminology / Concepts**
- **Function Calls**
- **Conclusions**



## Conclusions

- **IDUP-GSS-API is meant to be a GSS API extension / companion for "off-line" environments:**
  - **incorporates many of the concepts and calls of GSS**
  - **environment management, per-IDU, support calls**
  - **can be supported with public-key-based or symmetric-key-based underlying mechanisms**
- **IDUP-GSS-API is being implemented by BNR**
  - **implementation conforms to "PEM-Based IDUP Mech. (PIM)"**
  - **we are encouraging any interested parties to do a public-domain implementation (e.g., MOSS, MSP)**