To: INCITS Technical Committee T10
From: Gideon Avida, Decru
Date: Oct. 13, 2006
Document: T10/06-389r1
Subject: Using Public-Key Cryptography for Key Wrapping

1 Revision History

Revision 0 (06-389r0): Posted to the T10 web site on August 25, 2006.
Proposal overview for discussion in the Nashua meeting.

Revision 1 (06-389r1): Posted to the T10 web site October 13, 2006.
Added threat model discussion (section 4.4).
Detailed proposal (added section 5).
Fixed typos and added more references.

2 Introduction

Public-Key Cryptography enables sending secrets securely without requiring a prior shared secret.
There are several advantages for this approach:

1. There is no need to generate and maintain a secure channel and its related security association.
2. Keys can be generated/stored and wrapped away from the application client. This reduces key exposure as there may be multiple client servers (usually running a general purpose OS) and only a few key management servers.
3. There is flexibility in complexity vs. security.

The proposed algorithms are PKCS #1 v2.1 RSAES-OAEP and ECIES.

3 References

1. The IEEE P1363 Standard Specifications For Public-Key Cryptography — Amendment 1: Additional Techniques
   http://grouper.ieee.org/groups/1363/tradPK/index.html
2. ANSI 9.63-2001
   http://webstore.ansi.org/ansidocstore/subscriptions/product.asp?sku=ANSI+X9.63-2001
3. ISO/IEC 18033-2 Encryption algorithms - Part 2: Asymmetric ciphers
   http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=37971&scopelist=PROGRAMME
4. PKCS #1 v2.1: RSA Cryptography Standard
   ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-1/pkcs-1v2-1.pdf
5. FIPS 186-2 Digital Signature Standard (DSS)
   http://csrc.nist.gov/publications/fips/fips186-2/fips186-2-change1.pdf

4 Overview

4.1 Complexity vs. security

Public-Key cryptography allows for a flexible implementation than can support multiple levels of security:

4.1.1 Confidentiality

Each device server has a secret private key and a public key. The public key is used for wrapping key material sent to the device server. The private key is used by the device server to unwrap the keys sent to it. The entity wrapping the encryption key is assured that only the device server that owns the private portion of this public key can unwrap the encryption key.
At this level, the device server cannot authenticate the entity wrapping the encryption key or detect replay attacks. However, this mode is simple to implement and is an improvement over the current scheme (keys in plain-text). The weaknesses of this approach can be mitigated by relying on the lower protocols (e.g. ipsec for iSCSI of FC-SP for Fibre-Channel) or on physical security of the connection.
Note that integrity checking is included.

4.1.2 Confidentiality and Sender Authentication

In addition to its private/public key pair, each device server stores the public keys of the entities from which it can receive encryption keys. While these public keys are not secret, the device server shall store them in a way that will prevent an attacker from modifying a public key or even injecting his own (such operations will grant the attacker the ability to send wrapped keys to the device server). In addition, there must be controls on how public keys are introduced. This can be done out of band in a vendor specific way.
The device server's public key is used for the encryption operation of the wrapping, and the wrapping entity's private key is used for the signing operation of the wrapping. By verifying the signature, the device server is assured of the sender's identity.
At this level it is trivial to add protection from replay attacks by requesting a random nonce from the device server prior to wrapping and including this value in the signing operation.
Note that the same message format can be used in both modes. In the "confidentiality assurance only" mode, the nonce and signature fields may not be populated by wrapper, and are not check by the device server.

4.2 No protocol negotiation

Similar to the approach taken in 06-103r2, there will be no protocol negotiation. The client application will discover the wrapping format supported by requesting the Supported Key Formats page (see ssc3r03 8.5.2.5). The wrapping entity is required to support all formats.

4.3 Management of Public Keys

While the public keys are not secret, installing a public key in an entity authorizes that entity to receive from (in the case of device server) or send to (in the case of the key

wrapping entity) the owner of that public key. Therefore this process should be controlled.

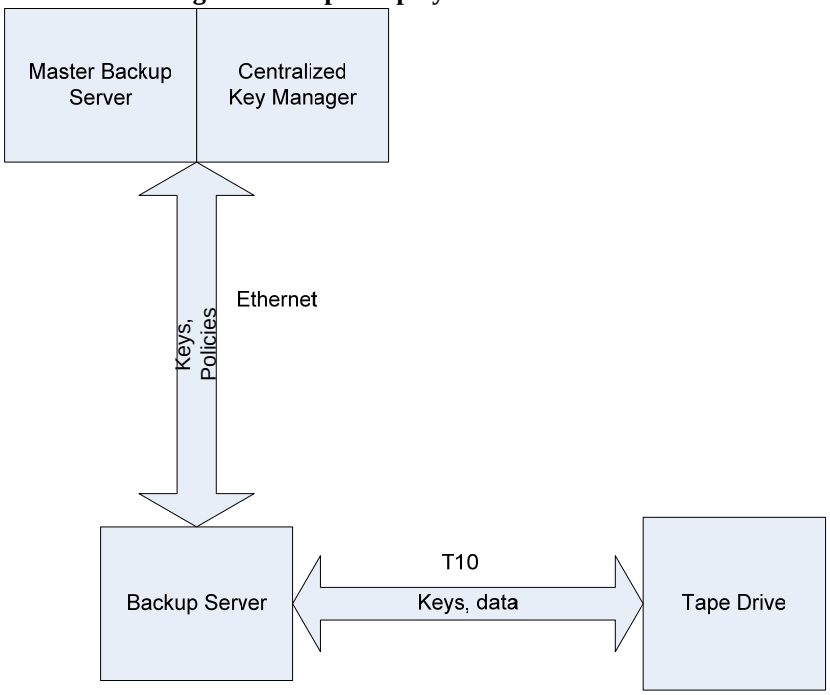
4.4 Threat Model Discussion

The goal of this section is to define the security environment, including the threats to the system, and to evaluate several proposals in light of this threat model.

4.4.1 Security Environment

The following diagram is meant to portray a typical deployment, in which a large number of tape devices are being accessed by a set of backup servers, each run by a master backup server. The addition of encryption in the tape drives introduces a Key Manager (which may or may not be part of the master backup server). Organizations will have a centralized key manager for the same reason that they use master backup servers, although there is no requirement that only one such is used, but rather that relatively few key managers are installed in an environment with a relatively large number of tape drivers and backup servers.

Figure 1: Simple Deployment Environment



Communication between the Key Manager/Backup Master and the Backup server is over the Ethernet network, while communication between the backup server and tape drive is over a network abstracted as “T10” (i.e. parallel-SCSI, FC, SCSI, iSCSI, etc.).

The following additional assumptions are in place:

1. The Ethernet network connection can easily be made as secure as the backup server (e.g. via SSL)

2. The Key Manager will be able to reliably associate keys with Backup Servers
3. Only the backup server is able to associate keys with data attributes (e.g. tape pool)
4. The tape drive has a minimal cryptographic module, able to encrypt and decrypt data, and to perform simple key load/key agreement operations.
5. Attacks against key disclosure or substitution for keys already stored within the tape drive, or Key Manager are out of scope.
6. The Backup server, in most cases, will have only a simple SSL module implemented in the underlying OS.

As a result of the above, we assume threats are against either the backup server or the T10 network.

4.4.2 Threats

In the following table, the first column lists threats, while the first row lists function being attacked. Entries in the table represent mechanisms to mitigate the threat.

Table 1: Threats overview

Threat	*Against Backup Server*	*Against t10 network*
Subverting key load protocol via CDB modification (e.g. turning off encrypted writes).	Harden the Backup Server OS of the backup manager	have a secure channel from the backup application to the tape drive.
Key Disclosure (passive)	Encrypt keys prior to reaching backup server. **or** Prevent information leakage by hardening the Backup Server OS to prevent caching or core dumps, etc.	Only send encrypted keys along T10 network, **or** have a secure channel from the backup application to the tape drive.
Injecting an attacker's key	Encrypt and sign keys at the key manager prior shipment to backup server. **or** Harden the Backup Server OS	Only send signed keys along the T10 network, **or** have a secure channel from the backup application to the tape drive.
Breaking association of keys to data via key replay	Harden the Backup Server OS	Rotate signing keys frequently, **or** have a secure channel from the backup application to the tape drive.

The most catastrophic threat is key disclosure (all other threats affect only a limited data window, and do not retroactively compromise encrypted tapes), and it can be mitigated by sending encrypted keys from the key manager to the Tape Drive, via the Backup Server.

The next class of threats is either turning off encryption or substitution of a legitimate key with an attacker's key. The latter is more difficult to detect, but may require some reverse engineering skills; and both attacks require active tampering with the T10 network. To defend against them, a secure T10 channel may be used (e.g. IPsec in iSCSI). Optionally, keys may be signed by the key manager.

The final class of threats assumes that an attacker has access to, or has compromised a valid key, and so wishes to read data by either substituting this (valid) key into a tape drive that processes someone else's data. Securing the T10 channel will protect against the substitution happening in the channel. The attack can also take place in the backup server, but this is problematic, since attackers with this ability will already have access to the policies governing key to data association.

4.4.3 Proposed Solution

We propose encryption and signing of the keys at the key management server as this will protect against the most serious threat of key exposure. This approach provides protection of the keys without requiring a major overhaul of backup server technology (i.e. OS and application hardening…)
Protection against link based attacks (e.g. key replay and disabling of encryption) can be done today with existing link protection technology such as IPsec and FC-SP.

5 Proposed changes to SSC-3 (based on ssc3r03a)

5.1 8.5.2 SECURITY PROTOCOL IN command specifying Tape Data Encryption security protocol

5.1.1 Modify table 94

Code	Description	Reference
0000h	Tape Data Encryption In Support page	8.5.2.2
0001h	Tape Data Encryption Out Support page	8.5.2.3
0002h-000Fh	Reserved	
0010h	Data Encryption Capabilities page	8.5.2.4
0011h	Supported Key Formats page	8.5.2.5
0012h	Data Encryption Management Capabilities page	8.5.2.6
0013h-001Fh	Reserved	
0020h	Data Encryption Status page	8.5.2.7
0021h	Next Block Encryption Status page	8.5.2.8
0022h-002Fh	Reserved	
0030h	Device Server Public Key page	8.5.2.9

0031h-FFFFh	Reserved	

5.1.2 Add 8.5.2.9

8.5.2.9 Device Server Public Key page

Note: The operator should be given a way to confirm that the public key received through the service delivery subsystem is indeed the device server's public key.

Table P – Device Server Public Key page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			PAGE CODE (0030h)				
1								(LSB)
2	(MSB)			PAGE LENGTH (n-3)				
3								(LSB)
4	(MSB)			PUBLIC KEY TYPE				
5								(LSB)
6	(MSB)			PUBLIC KEY FORMAT				
7								(LSB)
8	(MSB)			PUBLIC KEY LENTGH (n-9)				
9								(LSB)
10				PUBLIC KEY				
n								

The PUBLIC KEY TYPE field indicates the type of public key in the PUBLIC KEY field. Values for the PUBLIC KEY TYPE field are specified in table T.

Table T – PUBLIC KEY TYPE field values

Code	Description	Reference
0000h	RSA 2048	8.5.2.9.1
0001h-000Fh	Reserved	
0010h	ECC 521	8.5.2.9.2
0011h – BFFFh	Reserved	
C000h – FFFFh	Vendor Specific	

8.5.2.9.1 RSA 2048 Public Keys

The PUBLIC KEY FORMAT field shall be set to 0000h. The PUBLIC KEY LENTGH field shall be set to 512. Bytes 10 through 265 shall contain the modulus n. Bytes 266 though 521 shall contain the public exponent e.

8.5.2.9.1 ECC 521 Public Keys

The PUBLIC KEY FORMAT field shall be set to 0000h. The PUBLIC KEY LENTGH field shall be set to 67. Bytes 10 through 76 shall contain the ECC 521 public key as converted by the algorithm specified in ANSI X9.63 section 4.3.6 using the compressed form.

5.2 Changes in 8.5.2.8 Set Data Encryption page

5.2.1 Modify table 113:

Table 113 — KEY FORMAT field values

Code	Description
00h	The KEY field contains the key to be used to encrypt or decrypt data.
01h	The KEY field contains a vendor-specific key reference.
02h	The KEY field contains thy key wrapped by RSAES-OAEP
03h	The KEY field contains the key wrapped by ECIES
04h-BFh	Reserved
C0h-FFh	Vendor specific

5.2.2 Insert after paragraph beginning with “If the KEY FORMAT field is set to 01h…”

If the KEY FORMAT field is set to 02h or 03h, the KEY field will consist of three sub fields:
L – Public information associated with the key (e.g. Key ID).
C – The encrypted key
S – The sender’s signature over C.

If the device server encounters an error while unwrapping a key or while verifying the signature over the wrapped key, the device server shall terminate the command with CHECK CONDITION status and set the sense key to ILLEGAL REQUEST and the additional sense code to INVALID FIELD IN PARAMETER DATA.

L shall consist of a version byte and a format byte; both set to 0000h, followed by Wrapped key descriptors (see 8.5.4.6).

If the KEY FORMAT field is set to 02h, C shall contain the key wrapped using RSAES-OAEP. The RSA modulus length shall be 2048 bits. The hash function used shall be SHA-256. The mask generation function (MGF) used shall be MGF1 with SHA-256 as the hash function.

C shall be the key encrypted with the device server's public as specified by PKCS #1 v2.1 in 7.1.1 RSAES-OAEP-ENCRYPT ((n, e), M, L), where (n, e) is the public key, M is the raw key and L as described above.

S shall be the RSASSA-PSS signature over C as specified by PKCS #1 v2.1 in 8.1.1 using the wrapping entity's private key. Note that the RSAES-OAEP-ENCRYPT operation adds an integrity check to both L and M (the key). The signature is for providing proof of the sender's ID.

If the KEY FORMAT field is set to 03h, C shall contain the key wrapped using ECIES-HC as specified in ISO/IEC 18033-2. The following parameters shall be used for ECIES-HC:

- ECIES-KEM
  - The elliptic curve used shall be Curve P-521 as specified in appendix 6 of FIPS 186-2 Change 1.
  - KDF shall be the key derivation function defined by SP800-56A, section 5.8.1 (Note: This deviation from the ISO standard is to allow conforming implementation to be FIPS 140-2 compliant). The KDF parameters are defined below:
    - H shall be SHA-512
    - AlgorithmID shall be set to 00001h
    - PartyUInfo shall be the 32 byte Recipient ID (see 8.5.4.7)
    - PartyVInfo shall be the 32 byte Sender ID (see 8.5.4.8)
    - SuppPubInfo and SuppPrivInfo shall not be included.
  - CofactorMode = 0
  - OldCofactorMode = 0
  - CheckMode = 1
  - SingleHashMode = 0
  - KeyLen = 96
- DEM1
  - SC is SC1 using AES-256 as the block cipher
  - MA is HMAC with SHA-512

S shall be the ECDSA signature over C as specified by FIPS 186-2 using the wrapping entity's private key.

5.3 New sections inserted after 8.5.4.5

8.5.4.6 Wrapped Key Descriptors

Each Wrapped key descriptor shall be of the format as specified in table A.

Table A — Wrapped Key Descriptor Format

Bit / Byte	7	6	5	4	3	2	1	0
0	Wrapped Key Descriptor Type							
1	Reserved							
2	(MSB)	Wrapped Key Descriptor Length (n-3)						
3								(LSB)
4	(MSB)	Wrapped Key Descriptor						
n								(LSB)

If more than one wrapped key descriptor is specified, they shall be in increasing numeric order of the value in the Wrapped Key Descriptor Type field.

The Wrapped key descriptor type field contains a value from table B that defines the contents of the Wrapped key descriptor.

Table B – Wrapped key Descriptor field values

Code	Description	Reference	Required
00h	Recipient ID	8.5.4.7	Yes
01h	Sender ID	8.5.4.8	Yes
02h	Key Label	8.5.4.9	
03h	Key ID	8.5.4.10	Yes
04h	Key Length	8.5.4.11	Yes
05-BFh	Reserved		
C0-FFh	Vendor Specific		

8.5.4.7 Wrapped Key Recipient ID descriptor

The Wrapped Key Descriptor Length field shall be set to 32.

The Wrapped Key Descriptor field shall contain the SHA-256 hash value of the recipient's public key. This value may be used by the device server to validate that key was wrapped with the correct public key. It may also be used for this by application client in the case were the key was wrapped be a third system (e.g. a key management server).

8.5.4.8 Wrapped Key Sender ID descriptor

The Wrapped Key Descriptor Length field shall be set to 32.

The Wrapped Key Descriptor field shall contain the SHA-256 hash value of the sender's public key. This value may be used by the device server to locate the sender's public key for signature verification.

8.5.4.9 Wrapped Key Label descriptor

The Wrapped Key Descriptor field shall contain information that may be used to identify the key. This information is not required to be unique (e.g. a label entered by the system operator). This field may be treated as an A-KAD.

8.5.4.10 Wrapped Key ID descriptor

The Wrapped Key Descriptor field shall contain the unique key identifier. This field may be treated as an A-KAD.

8.5.4.11 Wrapped Key Length descriptor

The Wrapped Key Descriptor field shall contain the length of the wrapped key in bytes.