Authentication Concerns for Tape Drive Encryption Key Wrapping

To: INCITS T10 Committee
From: Greg Wheeless, Symantec

Background:

There are currently proposals in development to provide a secure method for an application client to send an encryption key to a tape drive device server, known as “encryption key wrapping”.

Any such proposal should address the problem of authentication, specifically attacks involving impersonation of either endpoint to gain inappropriate access to encrypted information.

Revision History:

Revision 0 – Initial Document.

Impersonation Threats to Consider

Terminology:

Secure key exchange will likely involve additional keys used to secure the key that is in turn used to encrypt data on tape – for the sake of clarity within this document I'll used the term "Tape Key" to mean a key used to encrypt data on tape, and "Wrapping Key" to mean any key used to secure the Tape Key exchange.

Depending on the secure key exchange methodology, a "Persistent Wrapping Key" may be used to exchange a "Session Wrapping Key", which is used to secure a single Tape Key exchange.

Overview:

Encryption key wrapping should authenticate the endpoints of the key exchange as necessary to minimize vulnerability to an attacker impersonating either the application client or the device server. "Man-in-the-middle" attacks are considered here only as a mechanism for impersonating an endpoint.

Note that the application client, not merely the initiator port, is important to authentication as an attacker may have access to the same initiator port as a legitimate application client, especially where server virtualization is used.

Three scenarios seem especially important:

a) backup of data to a drive providing encryption;
b) restore of data from a drive providing encryption; and
c) management of Persistent Wrapping Keys.

Additionally, authentication of a device upon its initial introduction to a storage network may warrant special consideration.

Backup Scenario:

The backup scenario involves the application client sending a Tape Key to the device server, followed by writing some amount of data to tape encrypted by that device key. (This may be repeated many times for what the user of a backup application would consider a "backup job", but that repetition would not seem to add additional authentication concerns.)

An attacker may impersonate a device server, gaining access to the Tape Key (as well as the plaintext backup data). If done as a man-in-the-middle attack, the backup may succeed, leaving the attacker with the Tape Key for a known tape, or the attacker might choose a Tape Key of his own for use in the real tape drive, without direct indication of a problem to the application client.

The application client needs to authenticate the device server in the backup scenario, or at least insure that the key wrapping protocol makes the wrapped Tape Key unavailable to an impersonator or man-in-the-middle. However, some vulnerability to a man-in-the-middle attack will remain despite authentication.

An attacker may impersonate an application client, and interfere with a backup performed by a different application client on a storage network. The attacker might change the Tape Key after it was established by the legitimate application so that some or all of the data written to tape would be encrypted with a key of the attacker's choosing.

A mechanism to alert the legitimate application client to a key change (even one made by the same initiator port) would seem to address this issue. Unless there are additional threats, the device server does not seem to need to authenticate the application client in the backup scenario.

Restore Scenario:

The restore scenario involves the application client sending a Tape Key to the device server, followed by reading some amount of data from tape decrypted by that device key. (This may be repeated many times for what the user of a backup application would consider a "restore job", but that repetition would not seem to add additional authentication concerns.)

An attacker may impersonate a device server, gaining access to the Tape Key. If done as a man-in-the-middle attack, the restore may succeed, leaving the attacker with the Tape Key for a known tape without direct indication of a problem to the application client.

The application client needs to authenticate the device server in the restore scenario, or at least insure that the key wrapping protocol makes the wrapped Tape Key unavailable to an impersonator or man-in-the-middle.

An attacker may impersonate an application client, but there does not seem to be any way to gain the Tape Key by this action, as it is never revealed by the device server. While an attacker could impersonate the legitimate application client mid-restore, to gain access to the plaintext restore data, the attacker could achieve the same result by the simpler action of "sniffing" the storage network passively, so this does not seem to be a threat that authentication can address.

The device server does not need to authenticate the application client in the restore scenario, as the possession of the Tape Key serves that purpose.

Key Management Scenario:

Security policies may require a limited lifetime for Persistent Wrapping Keys. An application client would need to be involved in this process if this key management is done in-band. Key management is usually the hardest part of a secure system to get right, and the value of robust in-band key management cannot be ignored.

If the device server is capable of generating secure Persistent Wrapping Keys on demand, the role of the application client is limited to triggering this event, and the threat from impersonation is similarly limited.

However, if due to device server limitations an application client will need to generate a new Persistent Wrapping Key (or public/private key pair) and send that key to the device server in-band, the threat from impersonation is high.

An attacker impersonating an application client could cause a denial of service by changing the device server's Persistent Wrapping Key arbitrarily. An attacker impersonating a device server could, though a man-in-the-middle attack, gain access to the new Persistent Wrapping Key being sent to the device server.

If an application client will perform in-band key management services for a device server, the application client and device server need to be able to authenticate one another.

Illustrative Examples (Not a Proposal)

Simple Symmetric Key:

A simple approach to authentication and key wrapping can be illustrated with a symmetric key as a shared secret. This key would be used both for authentication and as the Persistent Wrapping Key.

Before a tape drive was first used in normal operation, a symmetric key would need to be shared between the tape drive device server and the application client. Many methods are available, for example:

a) a symmetric key is set in the tape drive at the factory, and communicated out-of-band to the application client;
b) a symmetric key is generated by the application client and communicated out-of-band to the device server; or
c) a symmetric key is sent in-band in the clear to the device server in a controlled environment before production deployment.

In the backup and restore scenarios, the use of the symmetric key as the Persistent Wrapping Key would implicitly authenticate (or alleviate the need for authentication, depending on one's interpretation) the device server to the application client, as only an endpoint with knowledge of this secret would have access to the Tape Key.

In the key management scenario, the symmetric key would allow the device server to explicitly authenticate the application client. For example, the device server could send a challenge to the application client, which would respond by encrypting both the challenge and the new symmetric key using the old symmetric key.

The obvious disadvantage of this approach is scalability. If multiple application clients wish to use a given tape drive, the application clients must share that tape drives Persistent Wrapping Key by some out-of-band method, which may be challenging to do securely. This key sharing would need to be repeated whenever the key was changed.

The advantage of this approach is its simplicity: very few resources are needed on the device server. Authentication would add little complexity over that required for key wrapping. If there was a need to provide key wrapping for a tape drive without multi-initiator support (perhaps for some future regulatory compliance, or in a virtual server environment) some approach similar to this would seem appropriate.

Public Key Infrastructure:

As an example of authentication within a public key infrastructure, consider the case where a data center administrator wishes to use an existing system of certificate creation and certificate brokers, but the tape drives lack the resources to generate certificates or to store certificates for many initiators in a storage network. (If the tape drive can generate key pairs and store certificates, authentication is a well-solved problem.)

A specific application client or clients may be designated by the data center administrator as key management clients (perhaps these application clients would be hosted on third-party key management appliances). A key management client would initially generate a private/private key pair and certificate on behalf of the tape drive. Depending on the protocol chosen, the private key could be used for both authentication and as the private Persistent Wrapping Key, or two key pairs might be generated: for the sake of simplicity of discussion we'll assume the one key pair serves both purposes.

The key management client would send the private key to the tape drive. Initially the key would need to be sent either out-of-band, or in-band in the clear in a controlled environment. Once the first key was established, a new key could be sent securely in-band, by using the current private key symmetrically to allow the device server to authenticate the application client with a challenge, or by using a more sophisticated protocol.

The key management client would share the certificate and public key with all other application clients using the certificate brokers, and expire the tape drives private Wrapping Key on the schedule called for by the security policy, and in general make normal use of the key management infrastructure. If multiple key management clients were desired for redundancy, they would still need to share the tape drive private keys by some secure out-of-band method.