



Hardware accelerated tape encryption with Amanda backup software

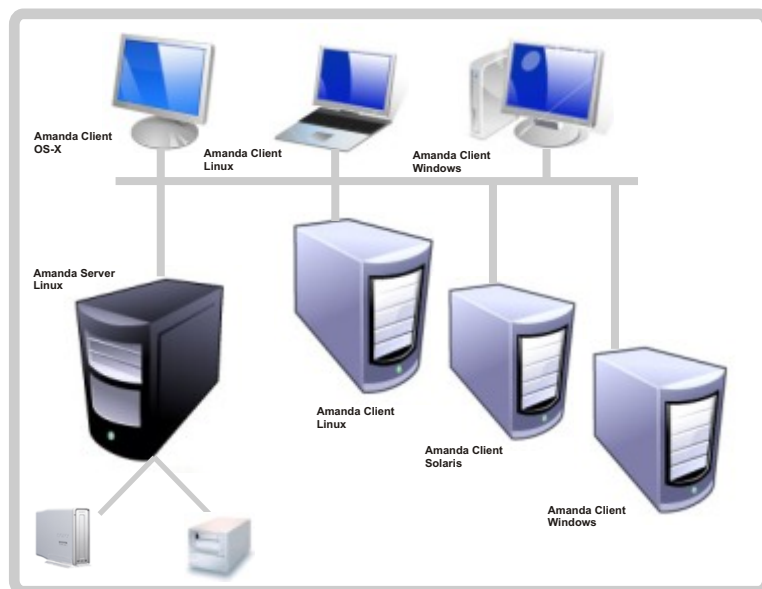
By Monish Shah, CEO, Indra Networks, Inc.

Introduction

Amanda (Advanced Maryland Automatic Network Disk Archiver) is an open source backup package that is popular with a variety of users in the commercial and educational domains. Amanda offers convenient, automated backups without the high cost of commercial backup packages.

One specific advantage of Amanda is that it allows users to plug in their own compression and encryption programs. Encrypting your backup data protects it from unauthorized disclosure in case backup tapes are lost or stolen. With increased emphasis on data security, encryption is fast becoming a requirement for many organizations.

The importance of data compression has long been recognized in the backup market. Virtually all tape drives made today include hardware compression. However, once data is encrypted, it looks highly randomized and such data is generally not compressible. This defeats the compression built into tape drives. Instead, you have to use Amanda's compression feature to perform compression before encryption.



*Fig. 1:
Network backup
using Amanda*

The Problem

A system administrator implementing compression and encryption with Amanda will run into two issues:

- Compression and encryption are very CPU intensive tasks. They can slow down the backup process. As a result, backup jobs may not complete within the allotted backup window. Further, if the same CPU is used for anything else, users will experience very sluggish performance.
- Key management: Encryption is done with a “key,” which is similar to a password. To retrieve the encrypted data, you must supply the original key to the decryptor. Without the key, it is nearly impossible to decrypt the data. Hence, it is imperative that you don't lose the key. At the same time, if someone steals the key, it is almost as if he/she has stolen the data, so key must be kept out of the reach of unauthorized people.

Amanda itself does not provide any key management features. It is left to the administrator to implement a suitable key management strategy. As we shall see later, securing the keys used in software encryption is essentially an unsolvable problem.

The Solution

As most readers are already aware, the solution to the problem is to use a dedicated tape encryption hardware device. Unfortunately, most available solutions are very expensive and are affordable to large enterprises only. Fortunately, Indra Networks offers a very cost effective solution that fits well within the budgets of SMBs (Small and Medium Businesses).

The solution consists of the StorSecure (SS) series of hardware accelerator boards and associated software to integrate with Amanda. An SS series board plugs into PCI-X or PCI-Express slot of a server and offloads compression and encryption tasks into dedicated hardware. Further, these boards provide secure storage for keys. Combined with Indra Networks' key management application, this enables a simple, easy to deploy key management solution that is inherently superior to software encryption.

The encryption algorithm is AES, which is the latest US government standard and is generally acknowledged to be the best encryption solution in commercial use. Key sizes of 128 bits and 256 bits are supported.

How It Works

In the typical case, an SS card is installed in the backup server, as shown in figure 2. The device driver for the card is also installed. At present, the device driver is supplied for Linux. Both 32 bit and 64 bit (x86_64) versions are supported. For other OSes, please check with Indra Networks.

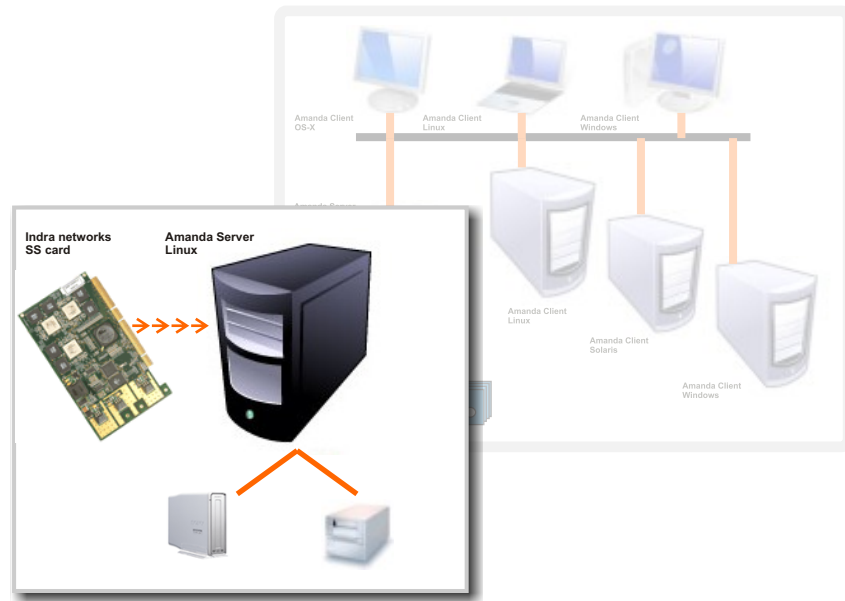


Fig. 2:
SS card inserted
in an Amanda server

The driver package for Amanda users also includes the following programs:

- "icompress" compresses data during amdump and decompresses the backup on amrecover
- "isecure" encrypts data during amdump and decrypts the backup on amrecover
- "icompress_secure" compresses and encrypts data during amdump and decrypts and decompresses the backup on amrecover
- "ikey_mgmt" manages keys stored in the SS card. This program can operate in either command line or GUI mode.

icompress, isecure, and icompress_secure are plugged into Amanda by putting appropriate entries in the amanda.conf file as shown in the example below:

```
define dumptype indra-compress-encrypt {
  global
  program "GNUTAR"
  comment "indra server compression and encryption dumped with tar"
  compress server custom
  server_custom_compress "/var/lib/amanda/icompress_secure"
}
```

Add following to the "disklist"

```
<client-ip-address> <backup-directory> <dumptype>
```

How It Works (contd.)

For example:

```
192.168.10.21 /backupdir indra-compress-encrypt
```

An important point to note is that it is not necessary to pass an encryption key to `isecure` or `icompress_secure`. The keys are stored in the SS card and used automatically, as explained below.

Key Management

The biggest challenge in implementing any encryption scheme is key management. The key must be accessible to the encryption program without being accessible to unauthorized users. Further, the keys themselves must be backed up, since any storage for storing keys is subject to failure.

With StorSecure, key management is done with a simple application called `ikey_mgmt`. To maintain security of keys, this application must be run as root. Also, for best security, the keys are stored in the StorSecure hardware itself. Figure 3 shows the movement of keys during various key management operations and during encryption / decryption operations. Note that under normal circumstances, keys remain within the hardware, device driver and root user's space. They are not made accessible to the backup user or any other unprivileged user.

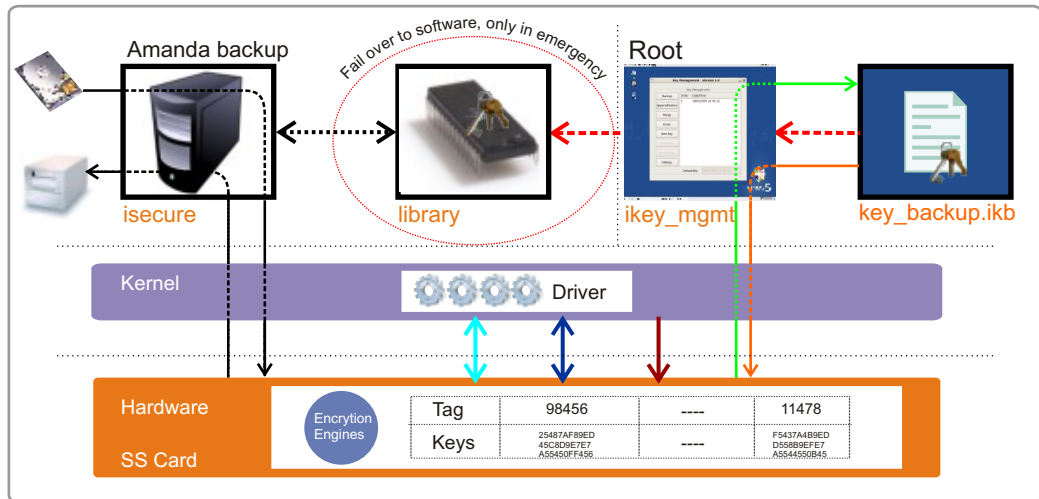


Fig. 3: Key Management

- Keys are created by issuing the appropriate command in `ikey_mgmt`. The command is given to the driver, which generates a random number as a key. The driver stores it in the StorSecure hardware, along with a unique tag. This path is shown in brown, in figure 3. Note that generating a random number is much more secure than generating the key from a password, as is done in some other schemes. The reason, in simple terms, is that a password based scheme can reduce the strength of a 128 bit or 256 bit encryption algorithm to a real strength of only 19 bits or so. [SCHNEIER]
- During an encryption operation, the default key is read by the device driver and then given to the encryption engine. The tag for the key is stored in the encrypted data stream. (This does not compromise the security of the encrypted data.) Although the data moves between the user level and the kernel driver, the keys never leave the kernel. The key movement for this case is shown in blue, in figure 3.

Key Management (contd.)

- During a decryption operation, the tag is read from the encrypted data and used to retrieve the key. The key moves only to the driver and then is used in the decryption operation. This key movement is shown in cyan.
- When keys are backed up, they move from the hardware, through the kernel driver to the root user and are stored in a file chosen by the root user. This path is shown in green. It is advisable to move that file into removable storage and put that device in a secure place.
- If a StorSecure card happens to fail and is replaced by a new card, the keys from the old card need to be restored to the new card. During this operation, the keys move from the file created during an earlier key backup, to `ikey_mgmt` and then to the driver and then the card. This path is shown in orange.

The operations listed above are the normal operations with StorSecure. There is an emergency recovery mode which can be employed to recover data if your StorSecure card happened to fail and you have not yet received a replacement. In this mode, the keys are retrieved from the key backup file and loaded into a user level library (shown in dashed red). This library can then be used by `isecure` to decrypt the data. Note that in this case, the keys are available to an unprivileged user. Hence, this mode should be used only in dire emergencies and precautions should be taken to keep untrusted users off the system when this type of recovery is in progress.

From the foregoing, it should be clear that as long as StorSecure is being used for encryption, the keys are kept within the hardware, device driver and the root user. They are not made accessible to other users. The only exception is software decryption in emergency mode.

In contrast, if software encryption / decryption were used, it would be necessary to make the keys available to the encryption / decryption software all the time. Since that software runs as backup user, the keys would have to be accessible to the backup user. While you may try to conceal where the keys are stored, a smart attacker can figure it out. This illustrates why hardware key management done by StorSecure is inherently more secure than any software encryption scheme.

Key Management best practices

`ikey_mgmt` program should be used periodically to create a new default key. The reason is that if the same key is used again and again and an attacker gets access to all the encrypted data, it becomes easier for the attacker to “break the code” and decrypt the data. Of course, this requires advance cryptographic knowledge on the part of the attacker.

When a new key is created, the old key will still be retained. This is so that the data encrypted with the old key can still be decrypted if required. As mentioned earlier, a tag stored in the data automatically selects the correct key. **The administrator is not required to remember which key was used for which backup, when using the Indra solution.**

An SS card can store upto 16 keys. If you run into the limit, you will have to eventually delete old keys. This should be done only when the old data has become obsolete. For example, if your tape rotation strategy is such that tapes are re-used after 3 months, keys older than 3 months are never required.

Key Management best practices (contd.)

Whenever a new key is created, a backup of all keys should be created and stored in a safe place. It is strongly recommended that at least one copy be kept offsite, in a secure location. The key backup should never be stored along with the encrypted data.

SS card in backup client

So far, we have assumed that the SS card is installed in the backup server. In some environments, there may be reasons to put an SS card in the backup client. For example if a particular server contains very sensitive data such that it should be encrypted even when traveling over the LAN, it may be sensible to add the SS card into that client. Another possibility is that a client may have a very large amount of data, in which case, compressing at the client may be required to avoid saturating LAN bandwidth. In such instances, an SS card can be added to the backup client and data can be compressed and encrypted right at the source.

Conclusion

Indra Networks' SS series cards are an ideal solution for SMBs who use Amanda and wish to encrypt their backups. The benefits of the SS series cards are:

- Hardware accelerated performance means that encryption can be implemented without loss of speed
- Simple yet secure key management
- AES encryption with 128 and 256 bit keys
- No need to memorize passwords (unlike some other encryption schemes)
- Low cost (pricing starts at less than \$1000)
- A readymade solution for Amanda users, saving time for the system administrator

References

1. [SCHNEIER] "Practical Cryptography" by Bruce Schneier, pg 348.