Encrypted Data Storage.

John White (for Patrick Guio and Joni Hahkala)
John.White@cern.ch
Helsinki Institute of Physics @ CERN
Brno, Tuesday June 21st, 2005

- Requirement.
- The plan.
- Present Status.
- The way ahead.

Encrypted Data Storage Requirement.

- There is a request/requirement from NA4/Biomed to provide an encrypted data storage (EDS)scheme.
- From `https://savannah.cern.ch/projects/egeeptf/`

PTF #	Name	Client
100542	On-disk encryption of data	Biomed
100597	Auto data encryption	NA4
100661	Long term data storage	NA4
100706	Store data in encrypted form	NA4

- From the above requirements we must provide the means to:
  - Encrypt data on disk to prevent data leaks at the storage site level;
  - Automatically encrypt data when it is written to a storage element.
  - Not use a user's Grid credentials for long term data storage;
  - It shall be possible for the user to store data in an encrypted form.

EDS General Plan.

- Services involved:
  - Data storage system
    - Consists of many services.
    - Stores the encrypted file and creates/returns GUID.
  - Metadata catalog
    - Stores the key(-parts in the second proto).
    - Storing and retrieval based on GUID.
  - OpenSSL
    - Used to create the key.
    - Used to encrypt/decrypt the file (file stream).

EDS General Plan. (0)

- Prototype 0.
- Provide documented openssl command-line programs to generate key.
- Provide documented openssl command-line programs to do encrypt/decrypt.
- Use the DM client programs to do storage/retrieval of key(s) and files.
- Essentially create some csh/sh scripts.

EDS General Plan. (A)

- Provide a (C/C++) API to encryption/decryption methods.
- **NO key splitting.**
- Methods (packages) needed:
  - byte createKey(int len) **(openssl)**
  - encryptFile(file in_file, file out_file, String algorithm, byte[] key) **(openssl)**
  - int storeKey(byte[][] key_parts, int numparts, String[] MC_endpoints) **(gLite DM)**
- Reverse methods:
  - byte[][] getKey(String[] MC_endpoint) **(gLite DM)**
  - int decryptFile(file in_file, file out_file, String algorithm, byte[] key) **(openssl)**
- The key storage and retrieval will use the gLite DM API.

EDS General Plan. (B)

- Provide a (C/C++) API to encryption/decryption methods.
- **Key split into n parts.**
- Methods (packages) needed:
  - byte createKey(int len) **(openssl)**
  - encryptFile(file in_file, file out_file, String algorithm, byte[] key) **(openssl)**
  - byte[][] splitKey(int num_parts) **(generic)**
  - int storeKeys(byte[][] key_parts, int numparts, String[] MC_endpoints) **(gLite DM)**
- Reverse methods:
  - byte[][] getKeyParts(String[] MC_endpoints) **(gLite DM)**
  - byte[] combineKeyParts(byte[][] parts, int numparts) **(generic)**
  - int decryptFile(file in_file, file out_file, String algorithm, byte[] key) **(openssl)**
- The key storage and retrieval will use the gLite DM API.

EDS: Current Status.

- Received an example encrypt/decrypt program from J. Montagnat.
- Unfortunately, the mechanics are incorrect. Useful as a guide.
- So we start from scratch...
- Generate a key the openssl way:

```
#include <openssl/bn.h>

// Generate the 256 bit key.  Symmteric encryption.
BIGNUM *largekey;

largekey = BN_new();
BN_rand(largekey, 256, 0, 0);
BN_print_fp(stdout, largekey);
```

- Currently, we are working on the openssl encrypt/decrypt.
- Afterwards, with the DM gLite APIs.
- **Lots of work to do.**

Conclusions.

- Work underway by JRA3 "CERN" cluster.
- We have a openssl contact at KTH, Richard LeVitte.