CS 4783-Applied Cryptography
Project 2
Professor Di Crescenzo

Stephen Russo
Benson Kuang

Cloud Storage

- New popular thing
- Allows for access of files anywhere
- Issues
  - Integrity detection
    - Files can be altered or modified
  - Replay attacks
    - Someone can get access to data transmitted
  - May not be confidential
    - Access to files
  - Example – 2014 iCloud hack

Our Approach

- Emulation of cloud storage
- Client-server interaction
- Involves hashing, public key cryptography, passwords, and session tokens
- Combined for a smooth, efficient, secure storage system

Data Process Functionality

- Client-server model
- User logs in with password
- Can save file or retrieve contents of a saved file
  - Similar to cloud storage

Confidentiality

- File names are hashed by server
  - SHA-256 used
- When user enters in a file to save, server hashes name of the file and encrypts its contents
  - RSA used to send key for AES
  - 16 byte key and iv is used
  - Encrypted, then hex-encoded then stored into file
- When user enters in a file to retrieve, server hashes name of file, finds it, decrypts its contents and outputs it
  - Hex-decoded then decrypted

Integrity Detection

- When user saves a file, its contents are hashed and stored at the end of a file
- When user retrieves the file, server decrypts the file
  - Hashes the file's contents and compares it to the hash at the end of the file to check authenticity
  - Notifies user if message is unauthentic and has been tampered with

Replay Attack Detection

- Password used to login to cloud
  - Encrypted and stored in a file that the server accesses
  - Name of file is hashed in order to hide it
- When client starts, server sends it a unique, random session token
- User enters in password
- Session token is appended to password and hashed, then sent to server

- Server gets password from file
- Decrypts password, appends token and hashes it
- Compares what it hashed to what it received
- If it matches, user gets access to save/access files
- If a foreign attacker retrieves the hashed string, it cannot do anything with it
  - It receives a different token when it tries to login, making that hashed string irrelevant
  - Does not protect if actual password was stolen, however

```
ben@ben-K55N:~/Desktop/Crypto/project2/clientFiles$ ./client
Please enter the destination IP address: 127.0.0.1
Please enter the destination port number: 1111
Successfully connected to server
Press 1 to save a file, 2 to retrieve a file or 3 to quit. 1
Enter in the file you wish to access: test
3645313034304645353945313139394335314445354539424234383445454344454331374235323 7
353741453932353146353635343141363137394546303538
Press 1 to save a file, 2 to retrieve a file or 3 to quit.
```

```
ben@ben-K55N:~/Desktop/Crypto/project2/serverFiles$ ./server
Please enter the port number to be used: 1111
Waiting for connection
Successfully connected to client
Message authentic.
1
test
This is a test file


Waiting for connection
^C
ben@ben-K55N:~/Desktop/Crypto/project2/serverFiles$ ls
3574bb3aa5ab09e03642deb944e7e4efe7f04130ddaf17cd48e4db0ebca2   server
9f86d081884c7d659a2feaa0c55ad015a3bf4f1b2bb822cd15d6c15b0f0a8  server.cpp
ben@ben-K55N:~/Desktop/Crypto/project2/serverFiles$ cat 3574bb3aa5ab09e03642deb944e7e4efe7f04
130ddaf17cd48e4db0ebca2
C3A22B54CB1A475FB7280E203B1CC7B6
430B23BF5798924E6CF42052D895ED0Cben@ben-K55N:~/Desktop/Crypto/project2/serverFiles$ cat 9f86d
f1b2bb822cd15d6c15b0f0a8 15a3bf4f
3638434445423546313646433433423344364631414438353443423233314238ben@ben-K55N:~/Desktop/Crypto
/project2/serverFiles$
```

Efficiency

- Fluid interaction between client-server
- System is fast and secure
- Notifies user if message has been tampered with

Questions?