README.md

Java encrypted-file-server

uses jni with c implementation of TEA encryption, with java classes KeyPairGenerator and KeyAgreement with diffie-hellman protocol to negoiate shared key for encryption

Design notes:

I used the the latest encrypt.c and decrypt.c files from the supplied moodle page as is with no modification (you can see these file for yourself). But I did choose how I applied these routine in practice: seeing as how "TEA operates on two 32-bit unsigned integers"(from Wikipedia where I found similar code). But this is insufficient as I was tranmitting messages that could be much larger than two 32-bit ints, I used the TEA encryption routine to encode much larger int arrays. So I would run the encryption routine on element 1 and element 2, then on element 2, and element 3 and so on. Given an int array with n ints in them :

encrypt(element1,element2),encrypt(element2,element3), encrypt(element3,element4)...encrypt(element(n-1),element(n))

This was accomplished with a simple for loop. This also made mandatory that the smallest message to be encrypted had to be the size of at least 2 or more ints. Also, since the encryption ran on ints, to encrypt messages in other datatypes (mainly bytes and strings), I had to convert the message to a byte array, and pad it zero bytes to ensure the byte array was an a size that was a multiple of 4 (or more if I had to pad to ensure the message was at least 2 ints, or 8 bytes long). Once the bytes are padded appropriatly, they can be encrypted as described above.

Decryption worked the exact opposite way, it ran the decrypt function on the second last byte and last byte, the decrypted the third last byte and second last byte as so: This essentially used the reverse of above for loop, accounting for the fact that the first value pointer would be the second last byte (n-1) and not (n) and decriment down to the first element.Given an int array with n ints in them : decrypt(element(n-1),element(n)),decrypt(element(n-2),element(n-1))...decrypt(element(1),element(2)) Note the for loops are actually 0 indexed in the code so indeces were from 0 to (n-1) and (n-2) to 0.

This protocol was not made to send int arrays directly as this could lose data. Current only bytes or strings are supported, but this could be extended. These messages are wrapped in EncryptedMessage objects, which only store an encrypted int array corresponding to data to transmitted.

For hashing and salting the shadow password file: I did not reinvent the wheel or try to get the TEA to work on inputed strings and a generated salt, which would involve more data conversions. Instead I looked around online (linked the resource below) and found another method. This involved creating random byte array as a salt, and concating this with the bytes of the combined string of a username with a password delimited by a comma. I take the concatenated salt,username and password combination and then use the existing library class MessageDigest with MD5 hashing protocol to create a hex string hash out it. For info on the MD5 algorithim, see wikipedia page in sources.

For the design documents(UMLs) I used the ObjectAid(class diagram), an external tool for eclipse. creatly.com for sequence diagram (please note I did not involve static classes in this diagram as they cannot really have life lines)

How to run

I did not use Eclipse for actual coding or testing functionality, so please use the make file. (it has all target specified)

```
make
```

Please note the output of the makefile to check that load path is correct

To run, first setup the server by typing:

`java Server`

follow the instructions to load up valid username and password hashes with a new salt. This is just simulating the server already existing and running with users already stored (this is in assignment spec: "...presumably already stored...").

`exampleusername,examplepassword`

`.done`

Optionally, you can add files to retrieve in the servers DATA folder at this stage, although a few already exist

on separate terminal (project said "for simplicity, the server should run on the same machine") run:

`java Client`

follow the instructions to login

`exampleusername`

`examplepassword`

enter valid file names that exist in the servers DATA folder.

`RickRollMarioPaint.mp3`

`.finished`

files should be saved in a new folder with the username and a time stamp

Test files:

`empty.txt`

`input.txt`

`RickRollMarioPaint.mp3`

`testphoto.jpg`

Final thoughts

I did not make the user interface very nice, as this was not a requirement. For example, to quit the server, one must send sigint with Ctrl+C. Also due to time constraints, I did not refactor as much as I would like. I am aware that the sequence diagram is not great, but it is hard to model this sort of program. (concurrency and sockets).

Sources

Background https://en.wikipedia.org/wiki/MD5 https://en.wikipedia.org/wiki/Feistel_cipher https://en.wikipedia.org/wiki/Tiny_Encryption_Algorithm

For idea of using MessageDigest class and md5 hashing, implemented slighlty differently (no point in making salt a string, or hardcoding it anywhere) http://viralpatel.net/blogs/java-md5-hashing-salting-password/

Refresher on diffie-hellman https://github.com/firatkucuk/diffie-hellman-helloworld/blob/master/src/main/java/com/codvio/examples/diffie_hellman_helloworld/Person.java

Mario Paint RickRoll by ihasmario http://www.albinoblacksheep.com/audio/mariopaintroll tolkien logo in omnivir's 'Collection of awesome LOTR posts' http://imgur.com/gallery/DKlrA

Code sources. Nothing copied verbatim, but modified for my own use:

http://stackoverflow.com/questions/17003164/byte-array-with-padding-of-null-bytes-at-the-end-how-to-efficiently-copy-to-sma
http://stackoverflow.com/questions/2165006/simple-java-client-server-program
http://stackoverflow.com/questions/17003164/byte-array-with-padding-of-null-bytes-at-the-end-how-to-efficiently-copy-to-sma
http://stackoverflow.com/questions/2833853/create-whole-path-automatically-when-writing-to-a-new-file
http://stackoverflow.com/questions/5343689/java-reading-a-file-into-an-arraylist
http://stackoverflow.com/questions/18268502/how-to-generate-salt-value-in-java
http://stackoverflow.com/questions/4746671/how-to-check-if-a-given-path-is-possible-child-of-another-path
http://stackoverflow.com/questions/4350084/byte-to-file-in-java http://stackoverflow.com/questions/8138411/masking-password-input-from-the-console-java