Progetto Foundations of Cybersecutiry

Punti Fondamentali:

- Autenticazione
- Protocollo di comunicazione
- Gestione dei file

Autenticazione

Per l'autenticazione l'idea è quella di usare il protocollo STS (Station To Station), utilizzando poi Diffie Helmann per lo scambio di chiavi. In questo caso potremmo non avere bisogno della firma digitale.

Scambio dei pacchetti:

Ispirato principalmente al protocollo di Giovanni: **CLIENT** A -> chiave per DFH per ottenre la session key KS C -> chiave per DFH per ottenere la chiave per l'hash KH **SERVER** B -> chiave per DFH per ottenre la session key KS D -> chiave per DFH per ottenere la chiave per l'hash KH

SIG -> firma digitale del server a partire dalle 4 chiavi A,B,C,D IV -> Initialization Vector

- M1: <opcode , username, username length, length(A), length(C), A , C >
- M2: <length(A) , length(B) , length(C) , length(D) , length(SIG) , IV,A , B , C , D, {SIG}KS >
- M3: <length(A) , length(B) , length(C) , length(D) , length(SIG) , IV,A , B , C , D, {SIG}KS >

Protocollo di comunicazione

Come gestiamo i pacchetti di comunicazione? Requisiti:

- Tutto il contenuto verrà cifrato con la chiave simmetrica
- Bisogna introdurre qualcosa per la freschezza, ad esempio un counter, questo potrebbe non bastare perchè potrebbe essere replicabile, quindi potremmo aggiungere una nonce random
- Il messaggio inviato dal client deve contenere:
- operation Code -> Codice di operazione che vogliamo svolgere
- counter
- timestamp
- dest-amount -> "None-0" di default

Questo contenuto viene poi hashato quindi il pacchetto avrà i campi visti sia in chiaro sia hashati così che chi riceve può controllare

- Il messaggio inviato dal server invece deve contenere:
- response Code
- timestamp del client
- response content
- response length

Anche queste info vengono poi hashate e inserite nel pacchetto

Quindi i pacchetti avranno questa struttura generica:

- IV
- lunghezza del contenuto
- contenuto
- Hash del contenuto

Gestione dei file

- File di balance (userBalance.enc.txt): +---------+---------+ | user_id | balance | +---------+---------+
- File di transfer (userHistory.enc.txt): +---------+---------+---------+ |tranc_id | dest | amount | +---------+---------+---------+

Accortezze

- gestione tentativi password in caso si utilizzi

Struttura file users:

+--------+------------+---------+ |username| hashed pswd | user_id | +--------+------------+---------+

comandi utili:

- cifrare il file

```
openssl rsautl -encrypt -in users.txt -out users.txt.enc -inkey
keys/server_privK.pem
```

- decifrare

```
openssl rsautl -decrypt -inkey server_privK.pem -in users.txt.enc -out
users.txt
```

- generare chiavi private e publice

```
openssl genrsa -aes128 -f4 -out user_privK.pem
openssl rsa -in user_privK.pem -outform PEM -pubout -out user_pubK.pem
```

- hash psw

```
openssl passwd -1 -salt $(openssl rand -base64 6) <password>
```