Ασφάλεια Πληροφοριακών Συστημάτων (ΠΛΗ591 Χειμερινό Εξάμηνο - 2016)

Project Report

- Λυγεράκης Φώτιος (2012030101)

Part 1

Στο πρώτο κομμάτι της εργασιας, που βρίσκεται στον φάκελο Menu του src χρησιμοποίησα τα αρχεία με τις συναρτήσεις κρυπτογράφησης που είχα δημιουργήσει στην άσκηση 2. Δημιούργησα, επίσης το αρχείο helper.py στο οποίο έχω συντάξει μερικές συναρτήσεις που με βοηθούν στην αποθήκευση των αποτελεσμάτων των υποερωτημάτων 1-4.

Για την δημιουργία του πιστοποιητικού χρησιμοποίησα την βιβλιοθήκη openssl της python:

```
openssl genrsa -des3 -out server.key 1024
openssl req -new -key server.key -out server.csr
cp server.key server.key.org
openssl rsa -in server.key.org -out server.key
```

Επίσης το πιστοποιοιτικό το κάνω self-signed:

```
openssl x509 -req -days 365 -in server.csr -
signkey server.key -out server.crt
```

Η υλοπίηση ήταν αρκετά απλή:

- Εμφανίζεται ένα μενού στην κονσόλα
- Ο χρήστης επιλέγει τι θέλει να κάνει πατώντας τον κατάλληλο αριθμό της υπηρεσίας
- Τυπώνονται τα κατάλληλα μηνύματα περιγραφής της διαδικασίας
- Για διευκόλυνση της εξέτασης του κώδικα έχω τοποθετήσει σε σχόλεια τις μεθόδους raw_input και έχω βάλει default ονοματα για τα files.
- Για την επιβεβαίωση του πιστοποιητικού χρησιμοποίησα την M2Crypto

Part 2

Στο δεύτερο κομμάτι της άσκησης,που βρίσκεται στον φάκελο TCP του src, ακολουθόντας τις οδηγίες που βρίσκονται στην ιστοσελίδα https://docs.python.org/2/library/ssl.html έστησα την επικοινωνία του Client.py και του Server.py.

Για να τρέξουμε την επικοινωνία ανοίγουμε σε διαφορετικά τερματικά τον client και τον server. Αφού γίνει η επιβεβαίωση του πιστοποιητικού του σερβερ μέσω της wrap_socket που χρησιμοποιώ, ο πελάτης στέλνει ένα μήνυμα στον σερβερ και ευτός το επιστρέφει πίσω.

Για την υλοποίηση χρησιμποίησα self-signed certificate για τον server.

Part 3

Τέλος,στο φάκελο TCP_SECURE του src χρησιμοποιώ σαν βάση το προηγούμενο μέρος και αρχικά επιβεβαιώνω και από τις δύο μεριές το πιστοποιητικό του καθενός, συμπληρώνοντας τα κατάλληλα ορίσματα στην wrap_socket.

Στη συνέχεια, επειδή οι δικές μου υλοποιήσεις των κρυπτογραφικών συνερτήσεων δεν παράγουν τα κλειδιά και τα cipher σε μορφή που διαχειρίζεται εύκολα και για να εξασκηθώ λίγο και στην χρήση των συναρτήσεων της pycrypto, χρησιμποποίησα αυτήν για την κρυπτογράφηση του των μηνυμάτων.

Η διαδικασία που ακολουθήται(και μιάζει σαν να κάνεις self-chat) έιναι η εξής(μετά την πιστοποίηση):

1. Ο client παράγει RSA κλειδιά και στέλνει το δημόσιο κλειδή του στον server
2. Ο server παράγει ένα τυχαίο κλειδί και ένα τυχαίο IV και στέλνει το IV unencrypted.
3. Έπειτα ο server στέλνει το AES κλειδί κρυπτογραφημένο με το δημόσιο κλειδί που του έστειλε ο client.
4. Ο client παίρνει το κρυπτογραφημένο AES κλειδί και το απόκρυπτογραφεί.
5. Ζητάει από τον χρήστη να του εισάγει ένα μήνυμα και στην συνέχεια το στέλνει κρυπτογραφημένο με τον AES.
6. Ο server λαμβάνει το κρυπτογραφημένο μήνυμα, το απο κρυπτογραφεί με το AES κλειδι το τυπώνει και το στέλνει κρυπτογραφημένο πίσω στον client.
7. Ο client το αποκρυπτογραφεί πάλι και το τυπώνει για να σιγουρευτούμε ότι όλα πήγαν καλά.

8. Στο μήνυμα εφραμόζεται zero-padding για να είναι πολλαπλάσιο του 16.