Advanced Topics in Online Privacy and Cybersecurity - 67515

PKI implementation – Inbal Mishal

Description of the solution:

1. **API:**
   In order to run the project, you use the next files: Root_CA.py that create the root certificate authority and VA_server.py that create the validation authority.
   Now you can create your entities using EntitySockets class in entity_sockets.py. You can see example of entity creation in e1.py.

2. **Libraries:**
   - **datetime** – Used to create the validity_date of the certificate.
   - **cryptography.exceptions** – Used to verify the entities signatures.
   - **cryptography.hazmat.primitives** – Used for the sign and verify functions. It also helped us get a string of the private and public keys.
   - **cryptography.hazmat.primitives.asymmetric** – Used for the private and public keys. It also helped use in sign and verify functions.
   - **dateutil.relativedelta** – Used to compare between dates.
   - **socket –** Used in the implementation of the communication in the system.
   - **threading –** Used to create client socket that can send messages to others and server socket that listens.
   - **_thread –** Used to help the server treat many entities at the same time.
   - **Time –** Used to do sleep at the beginning of the running to get nicer print.

3. **Classes & methods:** In the code.

4. **Files and Architecture:**

   - **certificate.py –** Includes the Certificate class. Object of this class represents a certificate of entity in the system. It includes many parameters as you can see in the code (domain, public_key, CA_signature…). It has *str* function implementation and *cert_to_sign* function that return string of the certificate without the CA_signature – this is the string that the CA sign on.

   - **entity.py –** Includes the Entity class. Object of this class represents the data of entity in the system. It includes some parameters: is_CA, domain, private_key, public_key and certificate. When the entity is created, it doesn't has certificate (certificate = None) until CA sign on it. It has *signature* function that return the signature of the entity on a message.

- **entity_sockets.py –** Includes the EntitySockets class. Object of this class represents an entity in the system – with the ability to communicate with other entities. This entity can be a CA or not. It includes some parameters: entity (the data of the entity), IP, port, and others. When we create an object and call to *start* function, the system create two sockets – one that can execute actions like issue the entity on CA, revoke his certificate and turn into a CA. The other socket listens to messages that the entity may get like issue other entities (if it is a CA) or message to verify. When we run the program we can distinguish between the green server messages and the blue client messages on the cmd.

- **VA_server.py –** Includes the VA class. Object of this class represents the validator authority and has many parameters such as IP, port and others. One of the parameters is the *cancelled_certificates* list that includes all the certifications that cancelled. The role of this class is to verify certificates of entities in the system. It has the *verify_cert* function that checks that the certificate of the entity is valid and reliable by checking the certificates of all the CAs back until the root. It has the function *revoke_cert* that can insert a cert to the *cancelled_certificates* list.

- **Root_CA.py –** In this file we create a CA entity and a certificate with the root_ca constants. Then we create for it an EntitySockets object and call to start function to run it.

- **e1.py –** In this file we create a regular entity and use it to create EntitySockets object and call to start function to run it. We can create many objects like this object (with different parameters).

- **utils.py –** Includes many helpful function like generate_keys, move from keys to string and back and move from certificate to string and back.

- **constants.py –** Includes many necessary constants for the sign action, the IP and ports of the root CA and the VA, others IP and port for testing the system, colors and others.