CSE 345/545: Foundations to Computer Security

Assignment: 1

Maximum marks: 100

Instructions:

- Follow the name convention and submission instructions for every question carefully.
- Your file names should follow the convention <Roll no.>_q0.py or <Roll no.>_q0_report.pdf.
- Keep your code efficient, make sure output matches the requirements.
- Post your queries on Google Classroom.
- **Strict plagiarism checks will be conducted for each question. The assignment must be done individually**
- Deadline for submission: 14th October 2022

---

1. Cryptography **35 marks**

You might be familiar with symmetric and asymmetric key cryptography. Both methods have their advantages and limitations. People who want to correspond via symmetric key cryptography have to share the key. The process is faster compared to asymmetric cryptography since the keys are much shorter, but if the channel being used for key exchange gets compromised, the entire system becomes insecure.

A popular algorithm to achieve asymmetric cryptography is RSA. One method to facilitate secure symmetric key exchange is to share them via asymmetric cryptography. Thereafter, all communication between the two parties would be secured through symmetric key cryptography. Suppose, Bob and Alice want to communicate via this paradigm.

Consider that **Fernet** will be used for symmetric encryption.

a. Alice generates the shared symmetric key: Use python's **Cryptography package** to generate the shared secret key K. Return the generated secret key. **[5]**

b. Bob generates his asymmetric keys: Use the **GMP library** to implement RSA. Take prime numbers 'p' and 'q' as inputs and generate 'n', 'e' and 'd' for Bob. <e,n> is the public key, <d,n> is the private key. Return Bob's public key. **[10]**

c. Alice uses Bob's public key to encrypt K: Use the symmetric key generated in part 1.a.encrypt m and return the ciphertext 'c'. as the message and encrypt it using Bob's public key <e,n>. Return the ciphertext 'c'. **[5]**

d. Bob obtains the shared symmetric key: Given the ciphertext ‘c’, use Bob’s private key to decrypt the message. Bob has now received the shared symmetric key K. Return K. **[5]**

e. Bob encrypts a message using shared symmetric key: Use the key K to encrypt a given message ‘m’ on Bob’s end and return the encrypted message. **[5]**

f. Alice decrypts Bob’s message: Given the ciphertext, use the shared key K to decrypt Bob’s message at Alice’s end and return the decrypted message. **[5]**

Please note:

- Integers p, q and m are up to 1023 digits long and should be taken as inputs.
- Avoid using loops to choose ‘e’.
- Use the template file provided to write your code.

Submission: A single python file <Roll_no>_q1.py

2. Authentication **25 marks**

A “JSON Web Token”, or “JWT”, is a stateless method of authentication that has recently grown in popularity. The JWT string contains the signature of its payload, signed using a private key/secret and following a chosen algorithm – this ensures the token’s integrity. Read more about the structure, and other standards of the JWT here.

a. Define a Python function `verifyJwt(token, secret)` that takes in a JWT and a secret as arguments. Validate the token’s signature against the supplied secret. If it is valid, return the decoded payload. Otherwise, throw an exception. The function should implement checks for at least two symmetric algorithms of your choice.
Note: You are not allowed to use the “PyJWT” library, or any other library implementing JWT verification. **[10]**

b. Consider the following JWT:

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJmY3MtYXNzaWdubWVu
dC0xIiwiaWF0IjoxNTE2MjM5MDIyLCJleHAiOjE2NzI1MTE0MDAsInJvbGUiOiJ1c
2VyIiwiZW1haWwiOiJhcnVuQGlpaXRkLmFjLmluIiwiaGludCI6Imxvd2VyY2FzZS
1hbHBoYW51bWVyaWMtbGVuZ3RoLTUifQ.LCIyPHqWAVNLT8BMXw8_69TPkvabp57Z
ELxpzom8FiI
```

Your task is to retrieve the secret used to sign the JWT. Once retrieved, create a new JWT with the same secret, and the role changed to “admin”. Document and explain your steps. **[10]**

c. Using a *single* secret to sign all JWTs that identify users can present a threat to the integrity and confidentiality of all users’ data in that application if it is leaked. What modification to the authentication architecture can you propose such that widespread damage can be prevented, if a JWT signing secret is cracked? **[5]**

Submission: A single python file <Roll_no>_q2.py, a report for parts b and c, <Roll_no>_q2_report.pdf

3. Digital Certificates 20 marks

This question would require you to get familiar with crt.sh and dnsdumpster. Your task here is to simply use dnsdumpster and crt.sh to fetch all the subdomains of iiitd.edu.in.

a. Once you have done that, fetch the private IP addresses of these subdomains and list them (subdomain:[PRIVATE_IP]). **[7.5]**

b. Explain your methodology and try automating this process as much as you can by writing a script in any language you are comfortable coding in. **[7.5]**

c. What according to you can be the security implications of private IP addresses being leaked to the public, i.e. if this list of subdomain and private IP addresses is given to an attacker outside the IIITD network, how can they leverage the same? **[5]**

Submission: A single code file in any language you are comfortable with <Roll_no>_q3.xxx, a report for parts a, b and c <Roll_no>_q3_report.pdf

4. Port Knocking 20 marks

Install and configure knockd on your VM such that by knocking a certain sequence of ports you are able to open and close port 22 SSH. The iptable rules should be made keeping all corner conditions in mind.

a. Explain as to how you went ahead and did all of the above alongwith the choice of iptable rules and the sequence of ports. **[15]**

a. Why should one prefer doing this over TCP instead of UDP? **[2.5]**

b. What is the default choice of ports in the knockd configuration. Is it safe? **[2.5]**