CSCI4145/5409: Network & Security

Reminder: This is an individual assignment. You are not allowed to collaborate with anyone else when completing this assignment. You can borrow code and configuration snippets from internet sources that are not from students in this class, however that code must be cited and include comments for how you have modified the original code.

Let's be spies!

This assignment is quite small and should not take you much time. You will build a REST API that lets us play around with encryption. Understanding encryption is important when working in cloud computing, so we'll give you some experience using public and private keys to encrypt and decrypt data.

Learning Outcomes

- Learn and apply an encryption algorithm (RSA)
- More experience working with AWS
- More experience building REST APIs, and working with JSON

Requirements

You will build a REST API using any language or framework you like. You can choose any deployment model for your API (IaaS or FaaS are your most likely candidates) and any AWS service to implement the functionality described below. This assignment could be completed, for example, by combinations of services like this:

- API Gateway + Lambda(s)
- EC2 or Beanstalk + Flask
- EC2 or Beanstalk + NodeJS

You will need the provided private_key.txt and public_key.txt files that come with these instructions on Brightspace. These files contain a 1024 bit public and private RSA key pair.

Once again, we will program a server which your app will correspond with. Your app will provide two endpoints:

- /decrypt This endpoint will use a private key to decrypt data
- /encrypt This endpoint will use a public key to encrypt data

Much like the Compute & Storage assignment we will follow this sequence:

1. You POST to our app's **/start** with your IP address and Banner ID.
2. We will use the provided public key to encrypt a string. We will then POST to your app's **/decrypt** endpoint with some JSON, this JSON will include the encrypted string's binary data encoded with **Base64** (so that we can send it as text through JSON easily). Your app will convert the Base64 encoded data back into binary data and then use the provided

private key to decrypt the data to reveal the hidden message. You will return the hidden message to us in your JSON response with a 200 (OK) status code. Obviously, the hidden message you return to us should match what we encrypted.

3. We POST to your app's **/encrypt** endpoint with a string we want you to encrypt. You will use the provided public key to encrypt the string. Your app will then convert the binary data of the encrypted string to **Base64** and return it to us in your JSON response along with a 200 (OK) status code. We will reverse the Base64 encoded data and use the private key to decrypt the message and compare it to the string we sent to you. Obviously, they must match.
4. We will return feedback and your score in response to your POST to **/start**, so you will be able to see your grade right away.

We will encrypt and decrypt with RSA and OAEP padding (RSAES-OAEP). You'll also see this referred to as the PKCS1_OAEP cipher.

This website has excellent examples for how to perform PKCS1_OAEP encryption and decryption in Python:
https://www.delftstack.com/howto/python/rsa-encryption-python/

This is a package that handles RSA encryption and decrypting (including PKCS1_OAEP padding) for Node.js:
https://npm.io/package/node-rsa

You will need to research how to encode and decode Base64 messages in your chosen programming language.

Our /start Endpoint Expects:

Your app will send the following JSON to our **/start** endpoint:

```
{
   "banner": "<e.g. B000123456>",
   "ip": "<e.g. 123.123.123.123:5000>"
}
```

Note the following:

- <>'s are placeholders intended to mean you replace this with some other text

We Will Send This JSON To Your /decrypt :

We will send the following JSON to your /decrypt endpoint:

```
{
   "message": "<e.g.
3hdsjk288ijklhasldfhljhfljhsdf83ljkhkjahfkjhdjksdfkjhsdfkjhasdfk
```

```
jhasdfkjhasdkfjhasdkjfhaskdjfhjshdfu83uh2hlkjh298h29hlnsdflknsdf
n2o3hr2o3n4l2n34lj2h34l2h34klj32h4kljh234kljn>"
}
```

You will return the following JSON response, along with a 200 (OK) status code:

```
{
   "response": "<e.g. I am the decrypted string!>"
}
```

We Will Send This JSON To Your /encrypt :

We will send the following JSON to your /encrypt endpoint:

```
{
   "message": "<e.g. Encode this secret message!>"
}
```

You will return the following JSON response, along with a 200 (OK) status code:

```
{
   "response": "<e.g.
3hdsjk288ijklhasldfhljhfljhsdf83ljkhkjahfkjhdjksdfkjhsdfkjhasdfk
jhasdfkjhasdkfjhasdkjfhaskdjfhjshdfu83uh2hlkjh298h29hlnsdflknsdf
n2o3hr2o3n4l2n34lj2h34l2h34klj32h4kljh234kljn>"
}
```

Marking Rubric

Your submission will be marked by the app that we will write. Just like in the Compute & Storage assignment you'll be able to get your feedback and grade immediately as a response to your POST to our /start endpoint.

Your mark is entirely based on the success of the steps defined in the requirements:

- **Your app posts to /start from an AWS IP address: 10%**
- **Your app successfully decrypts the data we send to you, returning the original string we encrypted: 45%**
- **Your app successfully encrypts the message we send to you, returning the Base64 encoded binary data representing the encrypting string: 45%**

I care about learning. If it takes you multiple attempts to learn the outcomes of this assignment, that's fine with me because in the end you learned! **Therefore, we will build our app such that only the performance of your highest scoring call to /start counts towards grading.**

How To Submit

Create a folder in your repository labeled **A3**. Put all your source code (whether it's python, node.js or lambda functions) in this folder and push it to your individual repository on gitlab **before the assignment deadline.**

I will publish the IP of my running app a few days before the assignment deadline, I will leave it running. You must build, provision, and execute your app such that it makes its call to /start **before the assignment deadline.**