**LAB 9**

**RSA Public-Key Encryption and Signature Lab**

**Task:1-Deriving the Private Key**

* Consider the prime numbers p, q, and e. Let's say n = p\*q. As the public key, we'll use (e, n). Please figure out what the private key d is. The following is a list of the hexadecimal values for p, q, and e. It should be emphasized that, while the integers p and q utilized in this work are quite huge, they are not secure.

Text

Description automatically generated with medium confidence

A picture containing text

Description automatically generated

**Task 2: Encrypting a Message**

* The public key is (e, n). Please encrypt the message "A top secret!" (Without the quotation marks).
* Here we can find the encrypted message.

Graphical user interface, text, application, Word

Description automatically generated

**Task 3: Decrypting a Message**

* This task's public/private keys are identical to those used in Task 2. Please decrypt and convert the following ciphertext C to a plain ASCII string.

Text

Description automatically generated

**Task 4: Signing a Message**

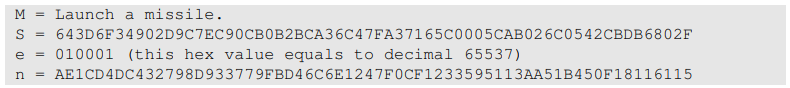
* This task's public/private keys are identical to those used in Task 2. For the following message, I'm creating a signature.

A picture containing text

Description automatically generated

**Task 5: Verifying a Signature**

* Bob receives a message M = "Launch a missile." from Alice, with her signature S.



* The signature above is corrupted, and the last byte of the signature changes from 2F to 3F, resulting in a single bit change. Please repeat this exercise and explain how the verification process will go.

Graphical user interface, text, application

Description automatically generated

**Task 6: Manually Verifying an X.509 Certificate**

* Obtaining a certificate from PayPal at www.paypal.com. Each certificate between the Begin Certificate and the line containing "END CERTIFICATE" should be copied and pasted into a file and saved as C0.pem and C1.pem, respectively.

**Step 1: Download a certificate from a real web server.**

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

A picture containing graphical user interface

Description automatically generated

* **C0.pem**

Text

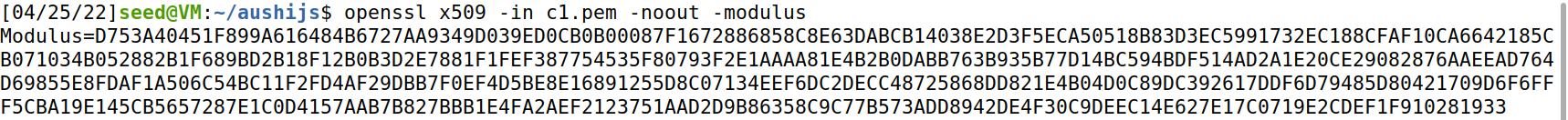
Description automatically generated

* **C1.pem**

**Text

Description automatically generated**

**Step 2: Extract the public key (e, n) from the issuer’s certificate.**

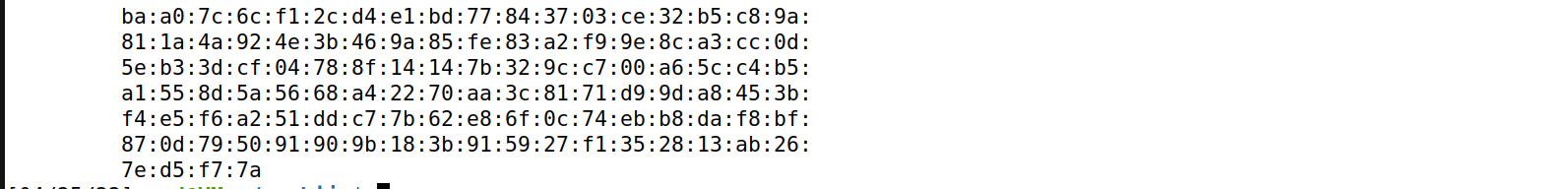
****

**Text

Description automatically generated**

**Text

Description automatically generated**

****

**Step 3: Extract the signature from the server’s certificate.**

**Text

Description automatically generated**

**Text

Description automatically generated**

* Extracting the issuer's certificate's public key (e, n). Certain attributes from an x509 certificate can be extracted using Openssl instructions. Using -modulus, we can get the value of n.
* Although there is no command to extract e, we may print all of the fields and readily discover the value of e. The exponent value can be found here. The command openssl x509 -in c1.pem -text -noout | grep exponent can be used. The exponent value can be found.

**Text

Description automatically generated**

**Text

Description automatically generated**

* To get a hex string that we can feed into our software, we need to remove the spaces and colons from the data. This can be accomplished using the command commands shown below.
* cat signature | tr -d '[:space:]:'

****

**Step 4: Extract the body of the server’s certificate.**

* X.509 certificates are encoded using the ASN.1 (Abstract Syntax Notation.One) standard, we can simply extract any field from a certificate if we can interpret the ASN.1 structure. Openssl includes a command called asn1parse that can parse our X.509 certificate and extract data from ASN.1 structured data.

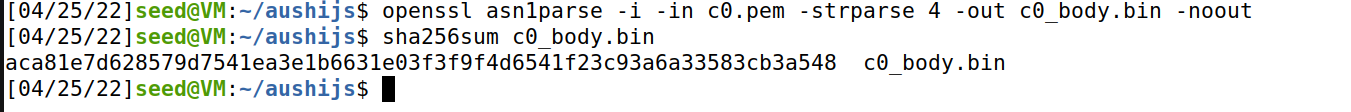
**Text

Description automatically generated with medium confidence**

**Text

Description automatically generated**

* X.509 certificates are encoded using the ASN.1 (Abstract Syntax Notation.One) standard, we can simply extract any field from a certificate if we can interpret the ASN.1 structure. Openssl includes a command called asn1parse that can parse our X.509 certificate and extract data from ASN.1 structured data.

****

**Step 5: Verify the signature.**

* We now have all of the data, including the public key of the CA, the signature of the CA, and the content of the server's certificate. We can use our own program to check whether or not the signature is legitimate.
* Although Openssl provides a command to verify the certificate for us, students must do it using their own programs; otherwise, they will receive 0 credit for this work.

**A picture containing text

Description automatically generated**

* **Signature is VALID**

**Text

Description automatically generated with low confidence**