Ministry of Education, Culture and Research of the Republic of Moldova
Technical University of Moldova
Department of Software and Automation Engineering

REPORT

Laboratory work No. 5
Discipline: Cryptography and Security

Elaborated: Pereteatcu Arina FAF - 223,

Checked: asist. univ.,
Dumitru Nirca

Chișinău 2024

Topic: Public Key Infrastructure(PKI) and Digital Signature Algorithm(DSA)

Conditions:

Create an internal PKI using the OpenSSL tool. The generation of the root private key and the initialization of a Certified Authority (CA) are required. A self-signed certificate is created for the CA. System must be able to issue and revoke the private key for the user so that he can subsequently generate a digital signature. Each user/entity that obtains a signature will be able to sign the document/file and verify this signature. For the realization of this laboratory, the use of any language is allowed, including script languages such as Bash, PowerShell, zsh etc.

Requirements:

1 Use Algorithm RSA for generating private keys

2 Users' private keys validity is 365 days and dimension keys are minimum 2048 bits.

3 Private key dimension for CA is 4096 bits and expired time for self-signed certificate in 10 years (3650 days)

Theoretical notes:

Infrastructure with Public Keys (PKI): PKI is a system for safely handling public-key encryption and digital certificates. Digital signatures, authentication, and secure communication are all made possible by it.
Essential elements:
Authority for Certificates (CA): issues digital certificates and cancels them.
Certificates: Attach public keys to users or servers, for example.
Mechanisms for revocation: Make sure that compromised certificates are rendered invalid.

The Certificate Authority's (CA) function: In PKI, the CA is the trusted root entity. Its duty is ranting users or entities certificates following identity verification, removing certificates that have expired or been compromised, notifying others about revoked certificates through the publication of a Certificate Revocation List (CRL).

Digital Certificates

An electronic document that certifies possession of a public key is called a digital certificate.
It includes:

- The public key
- Details about the certificate owner, such as name, email address, and company.
- The digital signature of CA
- Period of validity (beginning and ending dates).
- Distinct serial number.

Key Pairs and RSA Algorithm

- A **key pair** consists of:
  - **Private key:** Kept secret and used for signing or decrypting.
  - **Public key:** Shared publicly for verifying signatures or encrypting data.
- **RSA algorithm:**
  - Asymmetric cryptographic algorithm.
  - Widely used for key pair generation in PKI.
  - Key sizes (e.g., 2048-bit, 4096-bit) determine security level.

Certificate Signing Request (CSR)

- A **CSR** is a request for a certificate from a CA.
- It contains:
  - Public key.
  - Identification information (e.g., organization, common name).
  - Optional extensions.
- CSRs are digitally signed by the requestor's private key to ensure authenticity.

Digital Signatures

- A **digital signature** ensures:
  - **Integrity:** The data hasn't been tampered with.
  - **Authentication:** Verifies the sender's identity.
- Process:
  - **Signing:** The sender hashes the data and encrypts the hash using their private key.
  - **Verification:** The recipient decrypts the hash using the sender's public key and compares it with a newly computed hash.

Certificate Revocation

- Certificates can be revoked before their expiry due to:
  - Compromise of the private key.
  - Misuse or invalid identity information.
- **Certificate Revocation List (CRL):**
  - A list of revoked certificates published by the CA.
  - Used to verify the validity of a certificate.

OpenSSL

- **OpenSSL** is an open-source toolkit for implementing SSL/TLS protocols and PKI operations.
- It supports:
  - Key generation
  - Certificate signing
  - Digital signing and verification
  - CRL management.

Implementation:

My implementation includes a script that automates setting up a Public Key Infrastructure PKI environment using OpenSSL.

```
REM Create directory structure
mkdir C:\pki\ca\certs C:\pki\ca\crl C:\pki\ca\newcerts C:\pki\ca\private
icacls C:\pki\ca\private /grant:r "%USERNAME%":(OI)(CI)F
echo. > C:\pki\ca\index.txt
echo 1000 > C:\pki\ca\serial
```

Figure1. Create Directory

Mkdir: creates directories for the PKI files:

- certs: Stores certificates.
- crl: Stores Certificate Revocation Lists (CRLs).
- newcerts: Stores newly issued certificates.
- private: Stores private keys securely.

icacls:

Adjusts the permissions of the private directory to give full access to the current user. This ensures the sensitive private keys are protected.

echo:

creates an empty file named index.txt, which tracks issued certificates.

echo 1000:

Initializes the serial file with the value 1000. This file tracks the serial numbers of issued certificates.

```
REM Generate root CA key
openssl genrsa -out "C:\pki\ca\private\ca_root_key.pem" 4096

REM Generate root CA certificate
```

Figure2. Generate Root CA key

Here it is generated a 4096-bit RSA private key for CA. The key is stored in ca_root_key.pem within the private directory.

```
REM Generate root CA certificate
openssl req -new -x509 -days 3650 ^
    -key "C:\pki\ca\private\ca_root_key.pem" ^
    -out "C:\pki\ca\certs\ca_root_cert.pem" ^
    -subj "/C=US/ST=Internal/L=PKI/O=Organization/OU=IT/CN=Internal Root CA"
```

Figure3. Generate Root CA Certificate

openssl req: Creates a new X.509 certificate request.

Flags:

- -new -x509: Specifies the certificate is self-signed.
- -days 3650: The certificate will be valid for 10 years.
- -key: Uses the CA private key generated earlier.
- -out: Outputs the certificate to ca_root_cert.pem.
- -subj: Provides the subject details (e.g., country, organization, etc.) directly, avoiding manual prompts.

```
REM User certificate generation function
IF "%1"=="gen" (
    IF "%2"=="" (
        echo Usage: script.cmd gen username
        exit /b 1
    )
    openssl genrsa -out "C:\pki\ca\private\%2_key.pem" 2048

    openssl req -new ^
        -key "C:\pki\ca\private\%2_key.pem" ^
        -out "C:\pki\ca\certs\%2_csr.pem" ^
        -subj "/C=US/ST=Internal/L=PKI/O=Organization/OU=Users/CN=%2"

    openssl x509 -req -days 365 ^
        -in "C:\pki\ca\certs\%2_csr.pem" ^
        -CA "C:\pki\ca\certs\ca_root_cert.pem" ^
        -CAkey "C:\pki\ca\private\ca_root_key.pem" ^
        -set_serial "%RANDOM%" ^
        -out "C:\pki\ca\certs\%2_cert.pem"
)
```

Figure4. User Certification Generation Function

- **IF "%1"=="gen"**: Checks if the first argument (%1) is "gen". If true, proceeds to generate a user's key and certificate.
- **User Key (%2_key.pem):** Generates a 2048-bit RSA key for the user (%2).
- **Certificate Signing Request (CSR):**
- Generates a CSR (%2_csr.pem) for the user with the specified subject details.
- **Signed Certificate (%2_cert.pem):**
- Signs the CSR with the CA's private key and certificate.
- The user certificate is valid for 1 year (365 days).

```
REM Certificate revocation function
IF "%1"=="rev" (
    IF "%2"=="" (
        echo Usage: script.cmd rev username
        exit /b 1
    )
    openssl ca -revoke "C:\pki\ca\certs\%2_cert.pem" ^
        -keyfile "C:\pki\ca\private\ca_root_key.pem" ^
        -cert "C:\pki\ca\certs\ca_root_cert.pem"

    openssl ca -gencrl ^
        -keyfile "C:\pki\ca\private\ca_root_key.pem" ^
        -cert "C:\pki\ca\certs\ca_root_cert.pem" ^
        -out "C:\pki\ca\crl\ca_crl.pem"
)
```

Figure5. Certificate Revocation Function

- **IF "%1"=="rev":** Checks if the first argument is "rev". If true, revokes the specified user's certificate.
- **openssl ca -revoke:** Revokes the certificate by marking it invalid.
- **openssl ca -gencrl:** Updates the Certificate Revocation List (CRL) to include the revoked certificate.

```
REM Display root CA certificate details
openssl x509 -in "C:\pki\ca\certs\ca_root_cert.pem" -text -noout
```

Figure6. Display Root CA Certificate Details

Displays the details of the root CA certificate in human-readable format:

- Subject, issuer, validity period, public key, and more.

Output:

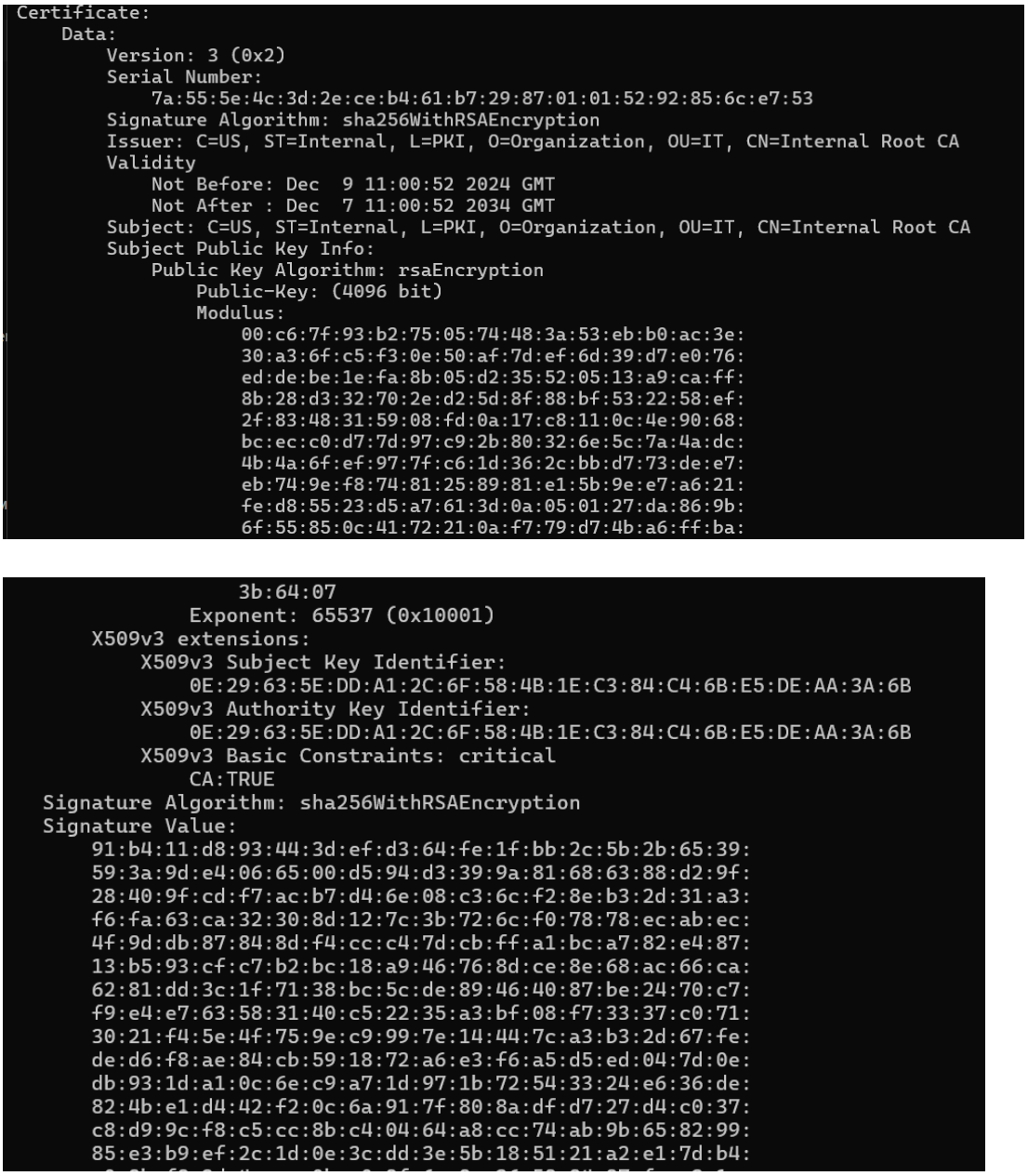

Figure7. Output

The output shows that a certificate has been successfully generated and signed by your internal Root CA. This certificate will be used for secure communications and can be trusted as it is signed by a CA that you control. The public key and other information in the certificate help verify the identity and authenticity of the certificate holder.

Conclusion:

This lab project demonstrated the significance of Public Key Infrastructure (PKI) in cryptography and digital security by offering insightful information on its real-world use with OpenSSL. In order to ensure compliance with the specified requirements, the assignment entailed managing digital certificates for users and entities, establishing a root Certificate Authority (CA), and developing a secure internal PKI system.

The following crucial elements were highlighted by the practical approach:

Role of the Certificate Authority: The CA was successfully set up as the trusted root, with the ability to issue, sign, and revoke certificates. This emphasized the CA's fundamental function in a PKI.

Digital Certificates: RSA techniques were used to generate and sign the certificates, which followed the recommended key sizes and validity durations. These certificates enable digital signatures and offer safe authentication.

Revocation Mechanism: By putting in place and maintaining a Certificate Revocation List (CRL), system security was strengthened by guaranteeing that compromised or invalid certificates could no longer be trusted.

OpenSSL Utility: OpenSSL's ability to handle PKI tasks including key generation, CSR creation, certificate signing, and CRL management was shown by its adaptability and effectiveness.

This experiment highlighted the value of simplifying cryptographic processes by using scripting to automate PKI setup and certificate administration. The dual objectives of data integrity and authenticity were further demonstrated by the incorporation of digital signatures.

All things considered, this exercise demonstrated the importance of PKI in building confidence in secure communications and its applicability to contemporary cryptographic techniques.