Contents

CST8805 Project

Overview

Students work in teams to design, document, build and test a secure document storage and transmission system.

The scenario is a bank that uses your system to authenticate and store banking documents such as payroll deposit files it receives from its clients to perform electronic fund transfers (EFT) into employee bank accounts. Before the bank performs the EFTs, it needs a high degree of assurance that the documents it receives from its customers have not been tampered with, originate from trusted sources and cannot be repudiated. The client for its part, needs the assurance that it is uploading these sensitive payroll documents to a web server it trusts.

Note: the actual process of performing the EFTs is out of scope!

Project Requirements

As a team, you design, document, build and test a secure document storage and transmission system that meets the requirements described below:

General Requirements

- The system must ONLY use approved NIST algorithms.
- The system implements the components illustrated in the "Topology Diagram" section and described in the "System Components Overview" section. Each component is to be implemented on separate virtual or physical machine.
- The document's confidentiality, integrity and authenticity must be safeguarded throughout the document's life cycle.
- The digital signature applied to the document must provide non-repudiation.

Document Transmission (data in transit) Requirements

- The transmission of the document from the client's web browser to the Bank's web server is performed using TLS (https) according to the following requirements:
    - Transmission of documents from client to server is performed over a secure TLS (HTTPS) where the client browser acting on behalf of the user authenticates the bank's web server.
    - The TLS implementation uses asymmetric encryption for symmetric key negotiation that MUST implement forward secrecy (PFS). All other TLS key negotiation protocols MUST be disabled.
    - Only NIST approved TLS algorithms and key sizes can be used. All others MUST be disabled as their use is strictly prohibited.

Document Creation Requirements - Client

- The clients digitally signs documents with the signing keys it has generated. Algorithms used for the signing process MUST be NIST approved.

Document Reception Requirement - Server

- The bank's application verifies the signature on the documents it receives over the secure TLS connection from the client before it processes their content such as performing EFTs. The

signature verification is implemented in the server's upload.php script. The verification steps performed in the upload.php script at the server must include:

- Validating the document's signature using the public key extracted from the x509 certificate provided by the client.
- Verifying that the client's x509 certificate was issued by a trusted issuer (i.e. the trusted bank's CA)
- Verifying the client's certificate was active and not expired at the time the signature was performed.
- Verifying that the client's certificate status was not revoked at the time the signature was performed. The certificate status verification must be performed against the CRL published at the CRL distribution point (CDP) specified as a URI in the signer's certificate.
- Verifying that client's certificate has the proper digital signing and non-repudiation extensions (out of scope)
- Verifying that the client's certificate is authorized for the type of document being signed (out of scope)

- The upload.php must report the results of each validation and verification steps. The verification is valid if all steps succeed otherwise the verification reports a failure.

Document storage (data at rest) Requirements

- Whenever the file is opened on the client or server, its digital signature must be verified (out of scope).

PKI Requirements

- The bank manages is its own Certificate Authority to issue user and web server x509 certificates for the bank's applications.
- The user and web server public/private keys CANNOT be generated at the PKI/CA. As in the real world, the key pairs MUST be generated by the users that will use them. User key pairs need to be generated on the client and web server key pair on the web server. Only the public key is shared with the PKI's CA via a CSR. DO NOT GENERATE THE KEY PAIRS AT THE PKI!
- Certificates MUST be created with the proper key usages extensions and attributes that include:
  - Digital signatures
  - Non-repudiation
  - CRL published at CDP
  - Certificate attribute values that would make sense in the real word.

System Components Overview

The project's system components are illustrated in the Topology Diagram. A brief description is provided below. The client, bank web server, CRL web server and PKI/CA MUST all be running on separate operating system instances as they would be in real life. All four components can be on the same physical machine BUT separate operating system instances. The LDAP directory does not need to be implemented for the project.

Client Computer

- The client runs a web browser that establishes a secure TLS (https) connection with the bank.

- The client's web browser includes the Bank's Trusted CA in its list of Trusted Root Certification Authorities.
- The client generates different public/private keys pairs for various test case as follows:
  - Valid
  - Expired
  - Revoked
  - Not issued by Bank CA
- Client generates document signatures for each of the test cases:
  - Valid
  - Expired
  - Revoked
  - Certificate not issued by Bank CA
- Smart Card (Smart Card) for secure signing / protection of public/private key pair used for signing Documents (out of score). We will be using soft certs.

Bank Web Server

- Web Server implements TLS (HTTPS).
- Web server is configured with a web server x509 certificate issued by the banks trusted root CA having the proper attributes set for a web server certificate.
- Web server runs the application that validates the documents uploaded by clients. The application consists of two web server hosted files:
  - upload.html
  - upload.php
- Template upload.html and upload.php files are provided on the CentOS VM. Template file location can be found in the VM_Notes.txt file located in CST8805 Centos8 VM's Documents folder.
- Hardware Security Module for secure cryptographic operations / protection of web server's public/private key pair (out of scope). We will be using soft certs.

PKI Server (Certificate Authority)

- The PKI server processes certificate signing requests (CSR) it receives from the client and server.
- It assigns the appropriate x509 key attributes and corresponding values.
- It revokes certificates.
- It publishes CRLs (to the CDP. Use the http distribution point!)
- It holds the CAs certificate signing private key
- Hardware Security Module (HSM) to protect the CA's signing key (Out of scope). We will be using soft certs.

CRL Server

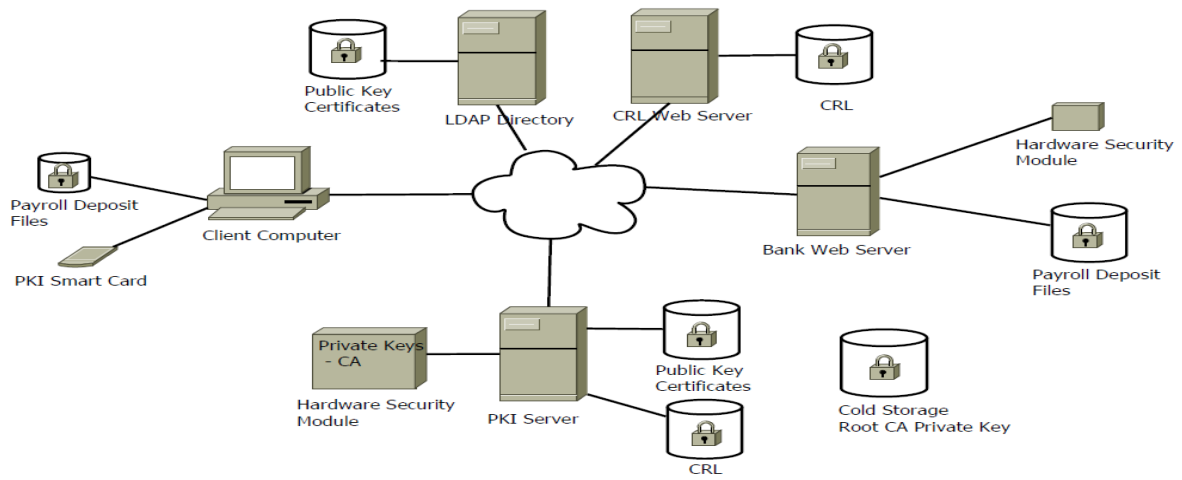
- The CRL server is simply a web server whose purpose is to host the CRLs published by the PKI.

Other Critical Requirements

To make the project implementation as close as possible to real life, the additional requirements that follow must be observed:

- Create a fully qualified domain name (FQDN) for your bank's web server that web browser's use to access the bank's web server application. The use of IP addresses to request web pages from the bank's web server is forbidden as it does not represent reality. Note that the FQDN can be any value providing it follows the general format: hostname.domain.toplevel. An example FQDN: payroll.bankofteam1.csa
- Create a fully qualified domain name (FQDN) for the web server that hosts the CRLs. The use of IP addresses in the X509's certificate CDP extension is forbidden. Note that the FQDN can be any value providing it follows the general format: hostname.domain.toplevel. An example FQDN: cdp.bankofteam1.csa
- Web browsers are to use the fully qualified domain name (FQDN) you created to access the bank's web application.
- The hosts' file on each system component will resolve the DNS requests for the bank and CRL web server's FQDN.
- PKI certificate attribute values such as cn, o, ou, email, st, etc must reflect the meaning of the attribute. Do not just enter any value that make no sense or repeat the same values across certificates or attributes. Be creative!
- The security posture of the Windows and Cent OS systems MUST be maintained. Avoid the use chmod 777 at all costs as it may pose a security risk and result in loss of project marks! Think "Principle of least privilege" when determining what an object's permission and ownership should be!

Topology Diagram



CST8805 Project Topology

Optional

- Use python scripts to implement any of the following processes:
  - Document signing at the client.
  - The transmission of the document to the server (use CURL).

Deliverables

- Demonstration of a fully tested, working and compliant system.

- A comprehensive test plan containing a set of test cases that describe the details on how to verify compliance with the documented requirements.
- Carefully review the Brightspace submission folder for submission details.

Grading Rubric

Please refer to document “Project Grading Rubric – 25W.docx”

