Bharatiya Vidya Bhavan's
SARDAR PATEL INSTITUTE OF TECHNOLOGY
(Autonomous Institute Affiliated to University of Mumbai)
Munshi Nagar, Andheri (W), Mumbai – 400 058.

Experiment No. 8 - Experiment on X.509 Certificates

Aim – To implement i) X.509 Certificate parsing and structure analysis, ii) Certificate chain validation and trust verification, and iii) Simple Certificate Authority (CA) operations and certificate lifecycle management

---

Problem Definition – X.509 is an ITU-T standard for public key infrastructure (PKI) that defines the format of public key certificates. These certificates bind public keys to entities through digital signatures from trusted Certificate Authorities. An X.509 certificate con3. Cipher Feedback Mode (CFB)tains version, serial number, signature algorithm, issuer name, validity period, subject name, subject public key, and extensions. Certificate chains establish trust hierarchies from root CAs to end-entity certificates through intermediate CAs. PKI enables secure communications, authentication, and digital signatures in distributed systems. Understanding X.509 certificates is essential for web security (HTTPS), email security, code signing, and enterprise authentication systems.

Laboratory Task (2 hours):

A) **X.509 Certificate Parser and Analyzer – Build comprehensive certificate analysis tool:**

- Parse ASN.1 DER/PEM encoded certificates
- Extract and display certificate fields (version, serial, validity dates)
- Decode subject and issuer Distinguished Names (DN)
- Analyze public key algorithms and parameters
- Parse certificate extensions (Key Usage, SAN, Basic Constraints)
- Implement certificate fingerprint generation

B) **Certificate Chain Validation Engine** – Create trust verification system:

- Build certificate chain from end-entity to root CA
- Verify digital signatures in certificate chain
- Check certificate validity periods (not before/not after)
- Validate certificate policies and path length constraints
- Implement name chaining and key usage verification
- Handle certificate revocation checking simulation

C) **Simple Certificate Authority Simulation – Develop basic CA operations:**

- Generate CA key pair and create self-signed root certificate
- Process Certificate Signing Requests (CSR)
- Issue end-entity certificates with proper extensions
- Maintain certificate database with serial numbers
- Implement certificate revocation and CRL generation
- Create certificate renewal and lifecycle management

Home Assignment: Research and implement OCSP (Online Certificate Status Protocol) client and responder simulation. Compare OCSP with Certificate Revocation Lists (CRL) for real-time certificate status checking. Analyze modern certificate transparency logs and their role in PKI security.

Useful YouTube Video Links –

1. X.509 Certificates Deep Dive – https://www.youtube.com/watch?v=kAaIYRJoJkc
2. PKI and Certificate Chains – https://www.youtube.com/watch?v=0ctat6RBrFo
3. Certificate Authority Operations – https://www.youtube.com/watch?v=x_I6Qc35PuQ

Assumptions – Use cryptographic libraries for ASN.1 parsing and basic certificate operations but implement validation logic manually. Create realistic test certificates for demonstration. Handle both PEM and DER formats.

Input – Menu-driven program: 1) Parse X.509 Certificate, 2) Verify Certificate Chain, 3) CA Operations (Issue/Revoke), 4) Certificate Status Check, 5) Quit. Users can load certificate files or generate test certificates. A single source file having all three programs is expected; and three separate files is NOT allowed as submission.

Input, Output and Submission –

Three files required:

1] Source file: Complete PKI implementation with certificate parsing, validation, and CA operations

2] Input-Output file: Test results showing certificate analysis, chain validation, and CA operations with sample certificates and CSRs

3] Report: Handwritten explanation of PKI trust models, certificate lifecycle management, and real-world deployment challenges

The names of three submission files must in the form: Exp06-<Your-Name>-<Your-UID>-<Source/Input-Output/Report>. For example, a student named Issac Newton with UID 202500145, the names of three files will be

Exp06-IssacNewton-202500145-Source.py
Exp06-IssacNewton-202500145-Input-Output.txt
Exp06-IssacNewton-202500145-Report.pdf