DIGITAL SIGNATURE

- **A Digital Signature is an electronic method of signing documents that:**
  - Proves who signed the document (authentication)
  - Ensures the document was not altered (integrity)
  - Makes the signer legally responsible (non-repudiation)
  - **It works using Public Key Infrastructure (PKI) and cryptographic keys.**

TYPES OF DIGITAL SIGNATURE

TYPE	PURPOSE
Class 1 -	Basic identity verification (rarely used)
Class 2 -	Moderate assurance (discontinued)
Class 3 -	High security, mandatory for govt portals

Most government systems require Class 3 DSC.

TYPES OF DIGITAL SIGNATURE

When we purchase a DSC from a provider such as eMudhra Limited or other licensed authorities, we get:-

- **USB Crypto Token**
- **DSC stored inside token**
- **Token password (PIN)**
- **Download link for drivers & signer utility**

JUDICIAL ACADEMY
LEARNING
ENDURING JUSTICE

HOW IT WORKS

Digital signatures rely on asymmetric cryptography, which means two mathematically related keys are used:

- Private Key → Secret, stored securely inside your USB token
- Public Key → Shared openly inside your Digital Certificate

JUDICIAL ACADEMY
LEARNING
ENDURING JUSTICE

HOW IT WORKS

We Have a Private Key (Secret)

and a Public Key (Shared)

- **When a Digital Signature Certificate (DSC) is created:**
  - **A special algorithm (like RSA or ECC) generates a key pair.**
  - **The two keys are linked mathematically.**
  - **What one key encrypts, only the other can decrypt.**

HOW IT WORKS

- **Private Key-**
  - **Stored inside the USB crypto token.**
  - **Never leaves the token.**
  - **Used only for signing.**
  - **Protected by PIN.**
- **Public Key-**
  - **Embedded in the digital certificate.**
  - **Can be shared with anyone.**
  - **Used only for verification.**

Even if someone gets your public key, they cannot derive your private key.

HOW IT WORKS

- When a File is signed, the Private Key Creates a Unique Encrypted Hash.
  - Before signing:
    - The computer runs the document through a hash function (example: SHA-256).
    - The hash function converts the entire file into a fixed-length string of characters.
      - ex- Original File:
      - "Salary Bill January 2026.pdf"
      - Hash Output:
      - 9F86D081884C7D659A2FEAA0C55AD015A3BF4F1B2B0B822CD15D6C15B0F00A08
- Properties of hash:-
  - Same file → same hash
  - Smallest change → totally different hash
  - Impossible to reverse back to original file

HOW IT WORKS

- Encrypting the Hash with Private Key
  - Now:
    - That hash is encrypted using your private key.
    - The encrypted hash becomes the digital signature.
    - Signature is attached to the document.
      - So-
      - Document + Encrypted Hash = Digitally Signed Document

Private key never leaves the token

Only encrypted hash is produced.

- **Anyone Can Verify the Signature Using Your Public Key**
  - **When someone opens your signed document:**
    - **Recalculate Hash-**
      - **Their system:**

        Takes the received document.

        Runs same hash algorithm.

        Produces a new hash.

- **Decrypt Signature Using Public Key**
    - **System uses your public key.**
    - **Decrypts the signature.**
    - **Obtains original hash (created during signing).**

- **Compare Both Hashes-**
  - New Hash == Decrypted Hash → VALID SIGNATURE
  - New Hash != Decrypted Hash → INVALID / TAMPERED

<u>SIGNING SIDE (You)</u>

Document
↓
Hash Function
↓
Hash Value
↓
Encrypt with Private Key
↓
Digital Signature
↓
Signed Document

<u>VERIFYING SIDE (Receiver)</u>

Signed Document
↓
Extract Signature
↓
Decrypt using Public Key → Original Hash
↓
Hash Document Again → New Hash
↓
Compare
↓
Match → Valid
No Match → Invalid

Authentication

Confirms who signed the document.

Integrity

Confirms document not changed

Non-Repudiation

Signer cannot deny signing later.

Summary-

Private key signs (encrypts hash).

Public key verifies (decrypts signature).

Matching hashes prove authenticity and integrity.

HOW TO INSTALL-PROXKEY

1. Install java 1.8

Go the location of script file-

```
sudo sh install_java.sh
```

The system will reboot

2. Install mToken driver-

```
sudo dpkg -i mToken_CryptoIDATools-1.0.3.amd64.deb
```

3. Install proxkey

```
sudo dpkg -i proxkey_ubuntu.deb
```

1. Open the Application Digital Sign a PDF-
2. Select any pdf file
3. Drag and drop box for Sign a Document
Insert the proxkey token
Add the library file key
Key Path-
/lib/WatchData/ProxKey/lib/libwdpkcs_SignatureP11.so