SECURE SMART CARD SIGNING WITH TIME-BASED DIGITAL SIGNATURE

Hossein Rezaeighaleh
University of Central Florida

2018 Workshop on Computing, Networking and Communications (CNC)
Maui, Hawaii, USA
March 2018

OUTLINE

- What is smart card?
- Is smart card secure for digital signature?
- Our innovative approach to use timestamped digital signature
- Implementation
- Performance evaluation
- Conclusion

WHAT IS SMART CARD?

Smart Card

- Several Usages: ID, Access,
  Metro/Subway, Telephone Card, etc.

As Personal ID and Cryptography device

- Microcomputer on a chip
- Stores User's Keys and Certificates securely
- Used to Encryption and Digital Signing

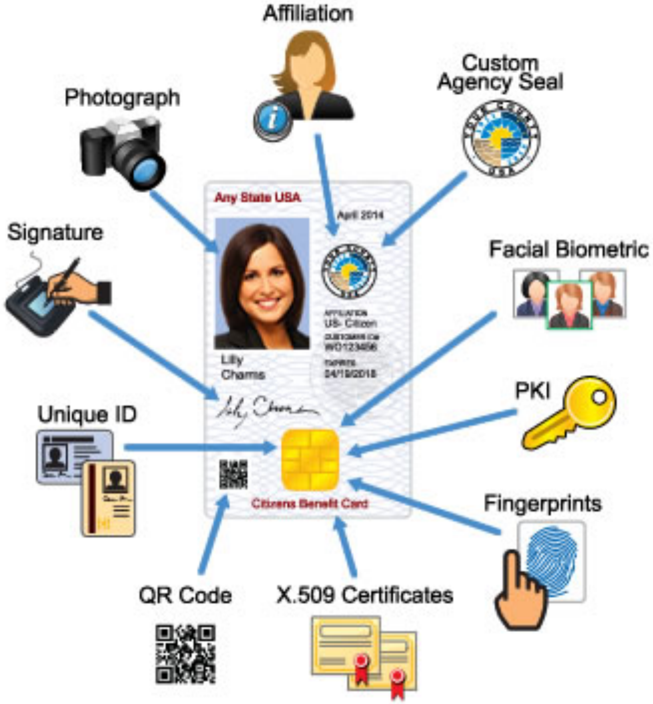
WHAT IS SMART CARD? (CONT.)

PIV Smart Card

Specific smart card for
Personal Verification in all systems (Interoperable)

NIST standard (SP 800-73)

- Name, Organization, …
- Face image/Fingerprint
- Keys/Certificates for digital signature



Source: NextgenID website

IS SMART CARD SECURE FOR DIGITAL SIGNATURE?

Essential smart card security challenges:

1. No direct user input
   - Keyboard, mouse, touch screen, …
2. No direct user output
   - Monitor, display, speaker, …

The user has to trust in terminal (PC, Laptop, smartphone) for input/output with his smart card

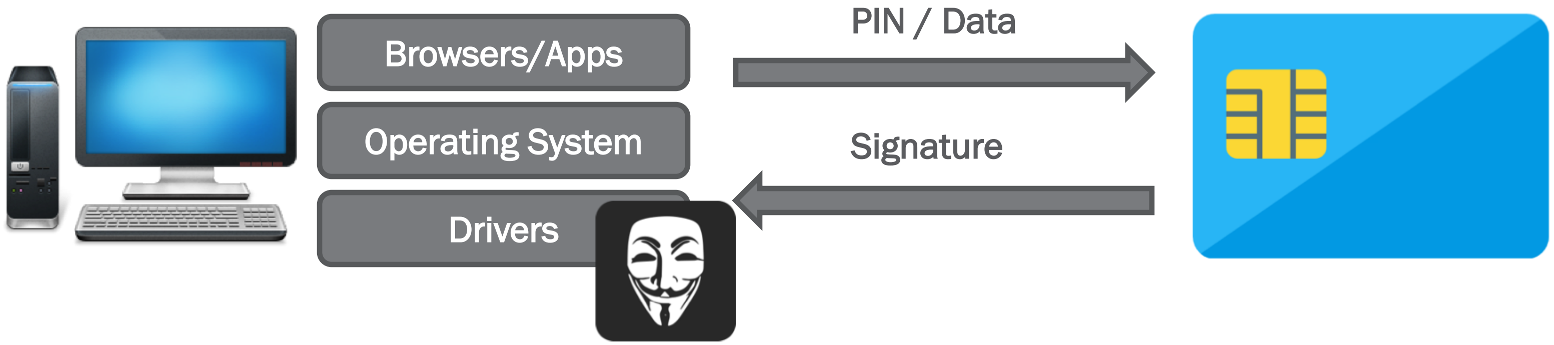
IS SMART CARD SECURE FOR DIGITAL SIGNATURE? (CONT.)

Terminal’s vulnerability:

Sniffing user’s credential (smart card’s password: PIN)

Altering data just before sending to the smart card for signing

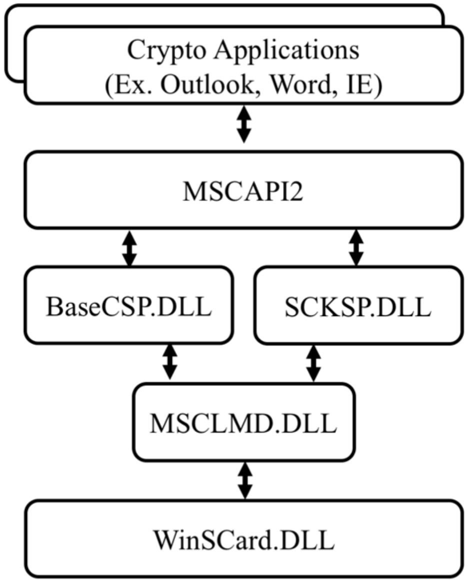
?



IS SMART CARD SECURE FOR DIGITAL SIGNATURE? (CONT.)

Case study: Windows

- Sniffing PIN
- Altering data to be signed



Crypto Applications
(Ex. Outlook, Word, IE)

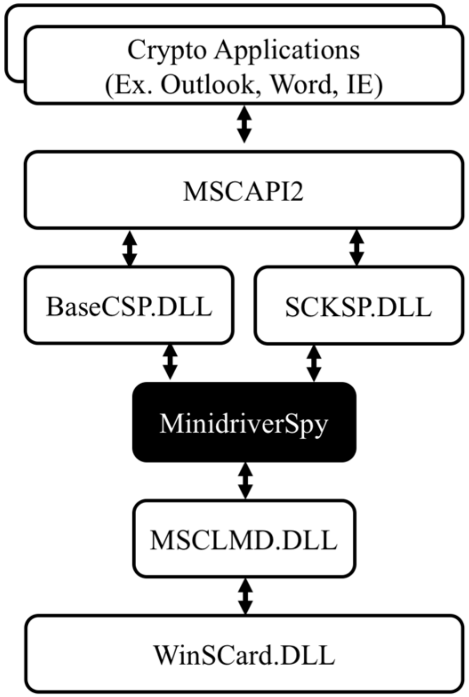
MSCAPI2

BaseCSP.DLL

SCKSP.DLL

MSCLMD.DLL

WinSCard.DLL



Crypto Applications
(Ex. Outlook, Word, IE)

MSCAPI2

BaseCSP.DLL

SCKSP.DLL

MinidriverSpy

MSCLMD.DLL

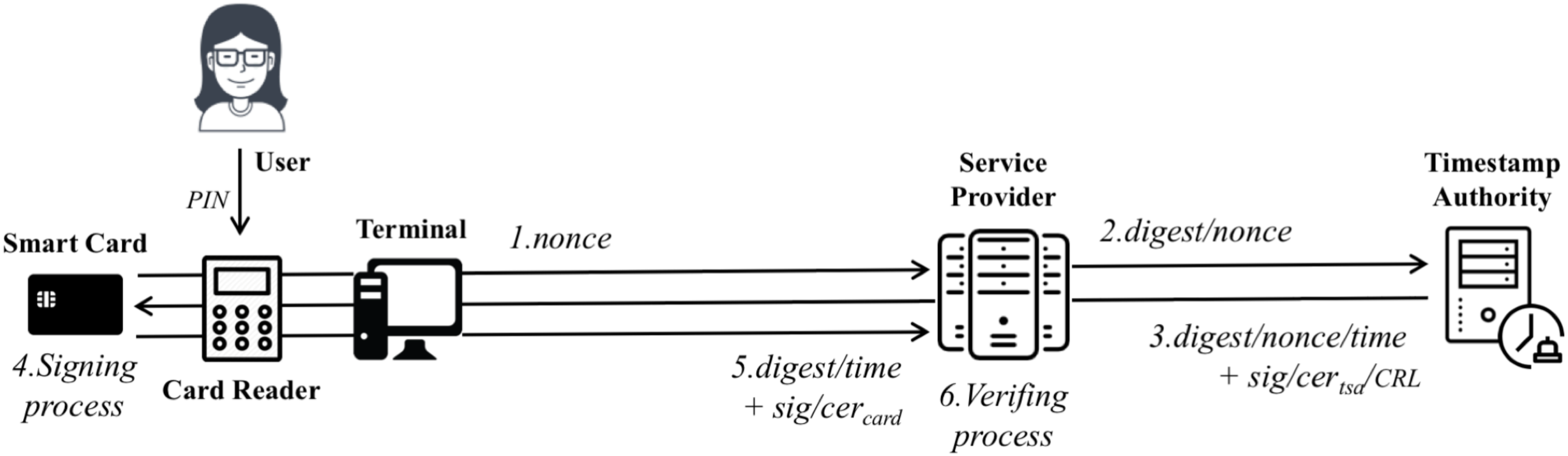
WinSCard.DLL

Open source:

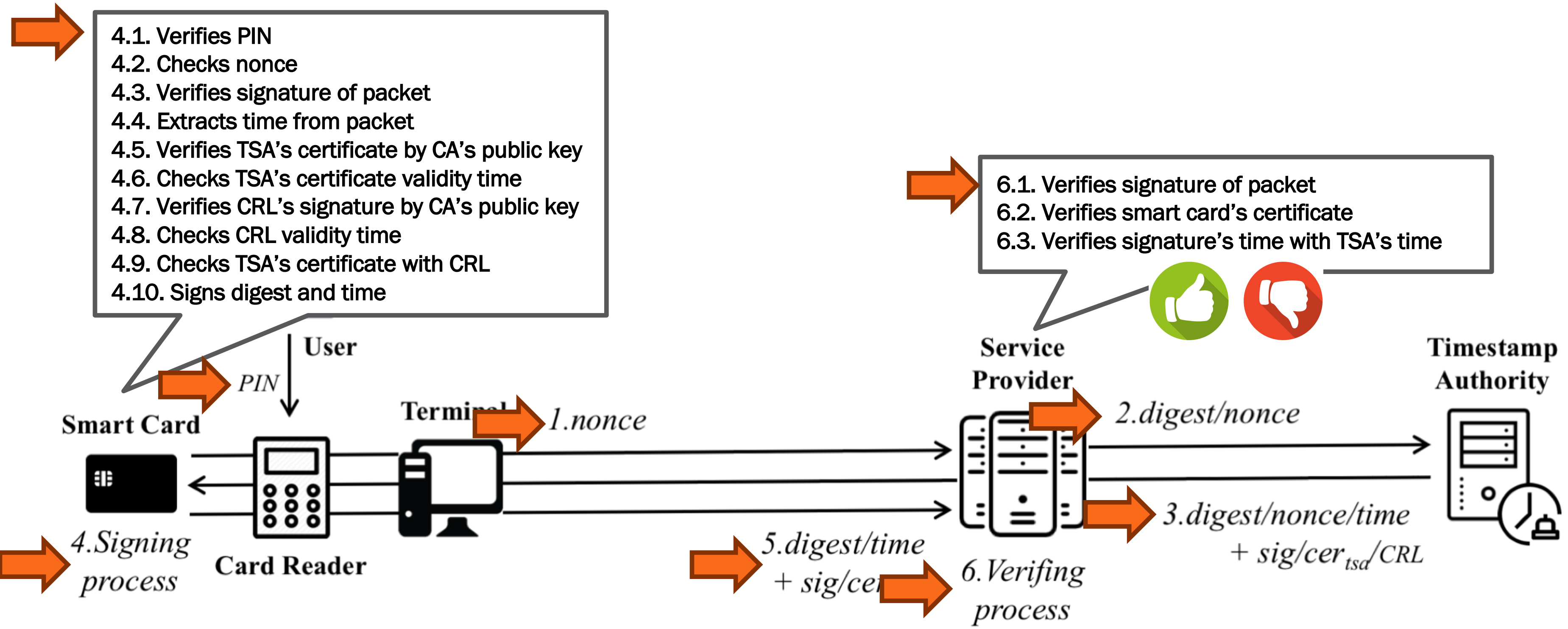
https://github.com/hosseinpro/MinidriverSpy

OUR INNOVATIVE APPROACH TO USE TIMESTAMPED DIGITAL SIGNATURE

1. Using an external trusted authority to pre-sign data
2. Moving entire process from the terminal to the smart card



OUR INNOVATIVE APPROACH TO USE TIMESTAMPED DIGITAL SIGNATURE (CONT.)



4.1. Verifies PIN
4.2. Checks nonce
4.3. Verifies signature of packet
4.4. Extracts time from packet
4.5. Verifies TSA's certificate by CA's public key
4.6. Checks TSA's certificate validity time
4.7. Verifies CRL's signature by CA's public key
4.8. Checks CRL validity time
4.9. Checks TSA's certificate with CRL
4.10. Signs digest and time

6.1. Verifies signature of packet
6.2. Verifies smart card's certificate
6.3. Verifies signature's time with TSA's time

IMPLEMENTATION

Java Card Applet

Challenges:

- No API or open source libra or Certificate, CRL, TSP and TLV (DER encoder/decoder)
- Limited memory: ~3 kilobyte
- Limited heap: performance penalty for recursive functions

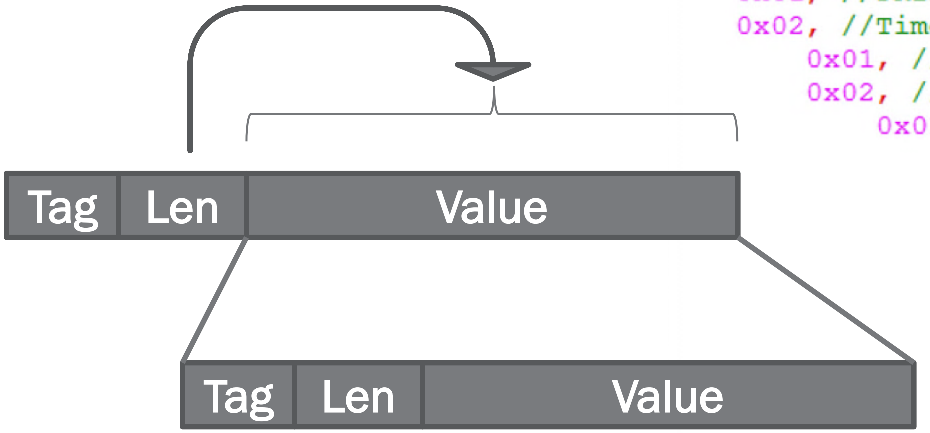
Write everything from scratch ;-)

Share just one byte array to do everything

One-time scanning technique !

IMPLEMENTATION (CONT.)

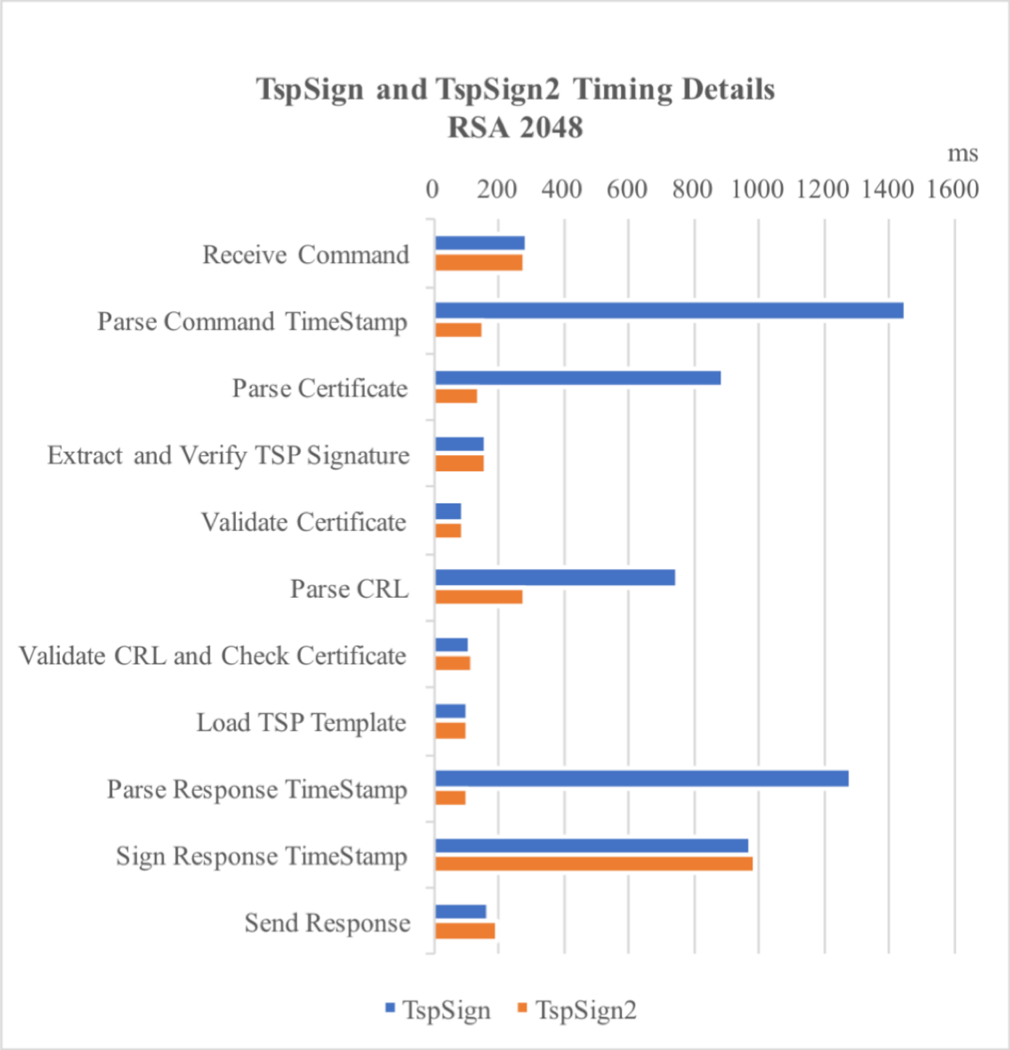
One-time scanning technique

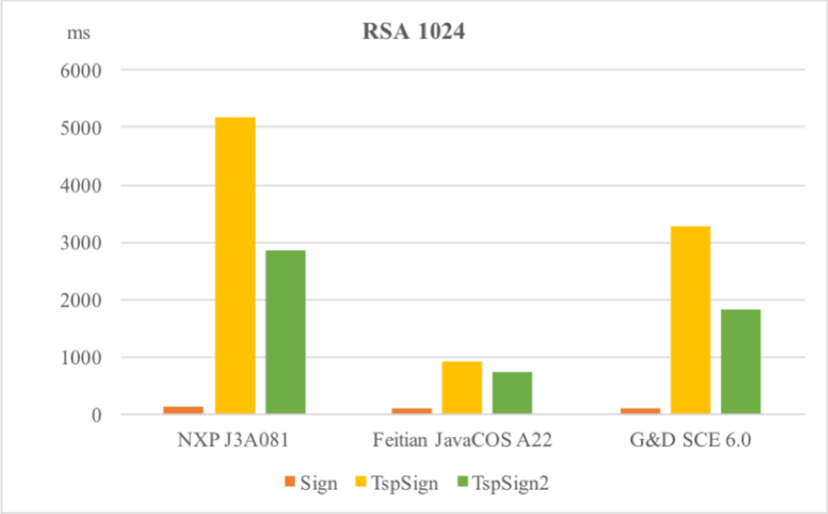


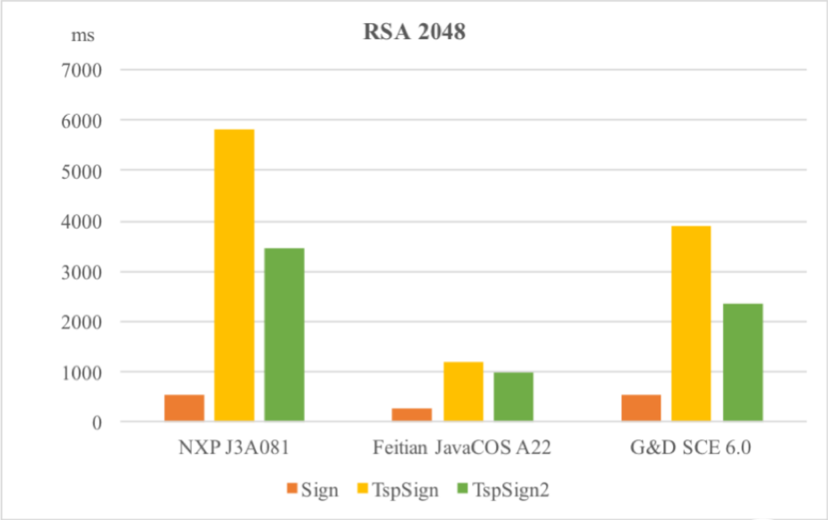
TLV: (DER-Encoded content) Certificate, CRL, TSP, …

```
0x00,//TimeStampResp
    0x01, //PKIStatusInfo
    0x02, //TimeStampToken (ContentInfo)
        0x01, //ContentType
        0x02, //content [0]
            0x01, //SignedData
                0x01, //CMSVersion
                0x02, //DigestAlgorithmIdentifiers
                0x02, //EncapsulatedContentInfo
                    0x01, //ContentType
                    0x02, //eContent [0]
                        0x01, //octetStringTypeIndex
                            0x01, //TSTInfo
                                0x01, //version
                                0x02, //TSAPolicyId
                                0x02, //MessageImprint
                                    0x01, //hashAlgorithm
                                    0x02, //hashedMessage
                                0x02, //serialNumber
                                0x02, //genTime
                                0x02, //nonce
                0x02, //certificates [0]
                    0x01, //certificate
                0x02, //crls [1]
                    0x01, //crl
                0x02, //SignerInfos
                    0x01, //SignerInfo
                        0x01, //CMSVersion
                        0x02, //SignerIdentifier
                        0x02, //DigestAlgorithmIdentifier
                        0x02, //signedAttrs [0]
                        0x02, //SignatureAlgorithmIdentifier
                        0x02 //SignatureValue
```

PERFORMANCE EVALUATION







Open source: https://github.com/hosseinpro/TspSign

CONCLUSION

- **Smart card is good but not enough!**
  - No direct I/O with the user
  - Relies on unsecure terminals
- **Our solution: Time-based digital signature**
  - Using an external trusted authority to pre-sign data
  - Moving process from the terminal to the smart card
- **Smart card implementation challenges**
  - No API or library => develop from scratch
  - Limited resource => resource sharing and one-time scanning
  - **Result: less than 1 second for 2048 RSA digital signature**