PV204: Project Presentation

Rudolf Wittner, Peter Sooky, Deepak Vishwakarma

Faculty of Informatics, MU

May 5th, 2016

Overview

- JPass: the Opensource application
- Applet Design
- PC Application
- Security Analysis
- Conclusion

JPass: the Opensource application

1. Password Manager
2. *https://github.com/gaborbata/jpass*
3. Functionalities:
   - Creation/generation of passwords
   - Organize username, password, URL
   - Import/Export in XML
   - User Verification
   - Change PIN
   - Save /Open file

JPass: Features Targeted

1. User verification through smart card (PIN)
2. Password generation in smart card
3. Storage of passwords in smart card
4. Other attributes URL, ID can be stored in JPass (secured way)

Applet Design

1. User authentication
2. Change user PIN
3. Generate Passwords
4. Store passwords
5. Password Retrieval
6. Update/delete password

PC Application

1. User authentication
2. Change user PIN
3. Generate passwords request
4. Save passwords
5. Retrieve passwords
6. Update/delete password

Communication Security

Attacker Model

1. Listen to the channel

Attacker Model

1. Listen to the channel
   $\Longrightarrow$ *eavesdropping*

Attacker Model

1. Listen to the channel
   $\Longrightarrow$ *eavesdropping*
2. Collect communication data

Attacker Model

1. Listen to the channel
   $\Longrightarrow$ *eavesdropping*
2. Collect communication data
   $\Longrightarrow$ attacker can learn the *session key* or PIN

Attacker Model

1. Listen to the channel
   $\Longrightarrow$ *eavesdropping*
2. Collect communication data
   $\Longrightarrow$ attacker can learn the *session key* or PIN
3. Store and forward data

Attacker Model

1. Listen to the channel
   $\Longrightarrow$ *eavesdropping*
2. Collect communication data
   $\Longrightarrow$ attacker can learn the *session key* or PIN
3. Store and forward data
   $\Longrightarrow$ attacker can launch replay attack

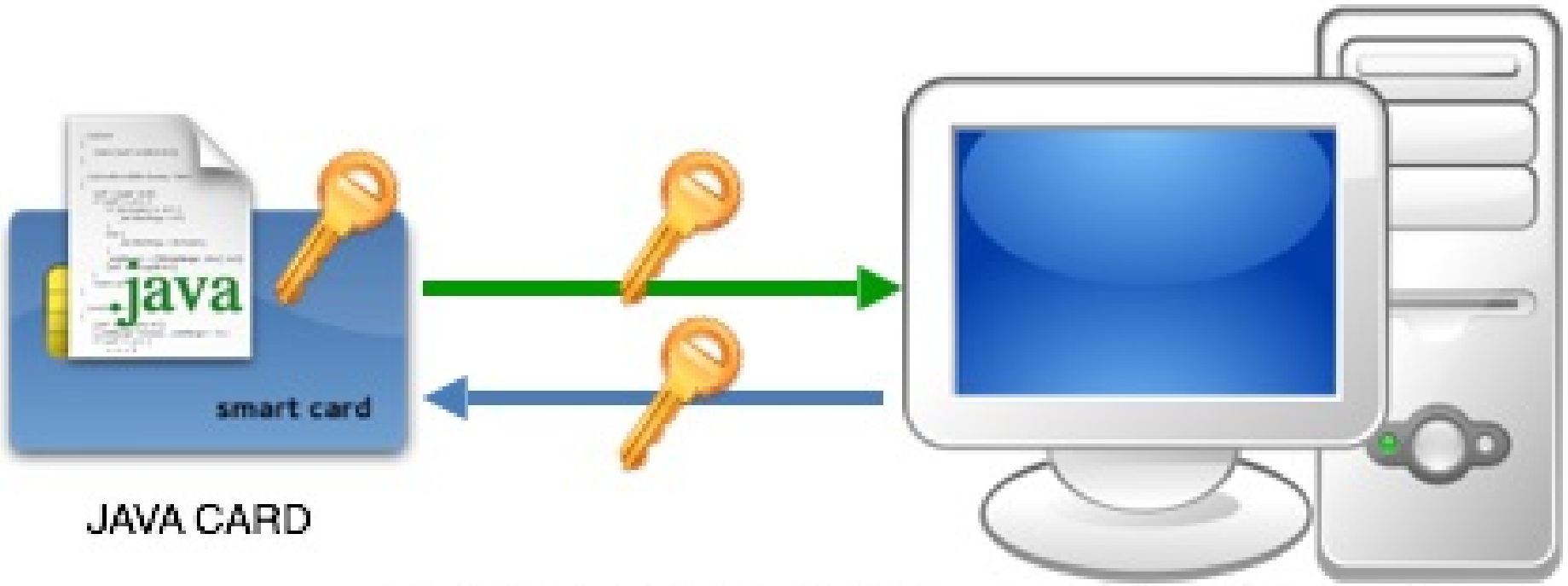
Attacker Model

1. Listen to the channel
   $\Longrightarrow$ *eavesdropping*
2. Collect communication data
   $\Longrightarrow$ attacker can learn the *session key* or PIN
3. Store and forward data
   $\Longrightarrow$ attacker can launch replay attack
4. Modify and send data to the smart card of PC application

Attacker Model

1. Listen to the channel
   $\Longrightarrow$ *eavesdropping*
2. Collect communication data
   $\Longrightarrow$ attacker can learn the *session key* or PIN
3. Store and forward data
   $\Longrightarrow$ attacker can launch replay attack
4. Modify and send data to the smart card of PC application
   $\Longrightarrow$ attacker can try to authenticate to smart card
   $\Longrightarrow$ send arbitrary commands to smart card
   $\Longrightarrow$ try to get data leaked from pc application or smart card

Security Mechanism



Security Mechanism:Session Key Derivation

1. Session Key Derivation:
   - Generate a salt (random number) in smart card
   - Session key hash (PIN + random number)
   - Different session key for every session due to salt
   - PIN acts as master secret which can be changed based on some policy
2. Freshness of data
   - applet and PC application use nonce to ensure it

Attacker Mittigation

1. Listen to the channel

Attacker Mittigation

1. Listen to the channel
   - $\Longrightarrow$ encrypted data
   - $\Longrightarrow$ all an attacker see in plain is a random number

Attacker Mittigation

1. Listen to the channel
   - $\Longrightarrow$ encrypted data
   - $\Longrightarrow$ all an attacker see in plain is a random number
2. Collect communication data

Attacker Mittigation

1. Listen to the channel
   - $\Longrightarrow$ encrypted data
   - $\Longrightarrow$ all an attacker see in plain is a random number
2. Collect communication data
   - $\Longrightarrow$ hard to do as session key is different for every session

Attacker Mittigation

1. Listen to the channel
   - $\Longrightarrow$ encrypted data
   - $\Longrightarrow$ all an attacker see in plain is a random number
2. Collect communication data
   - $\Longrightarrow$ hard to do as session key is different for every session
3. Store and forward data

Attacker Mittigation

1. Listen to the channel
   - $\Longrightarrow$ encrypted data
   - $\Longrightarrow$ all an attacker see in plain is a random number
2. Collect communication data
   - $\Longrightarrow$ hard to do as session key is different for every session
3. Store and forward data
   - $\Longrightarrow$ Nonce used by PC application and applet will protect against this

Attacker Mittigation

1. Listen to the channel
   - $\Longrightarrow$ encrypted data
   - $\Longrightarrow$ all an attacker see in plain is a random number
2. Collect communication data
   - $\Longrightarrow$ hard to do as session key is different for every session
3. Store and forward data
   - $\Longrightarrow$ Nonce used by PC application and applet will protect against this
4. Modify and send data to the smart card of PC application

Attacker Mittigation

1. Listen to the channel
   - $\Longrightarrow$ encrypted data
   - $\Longrightarrow$ all an attacker see in plain is a random number
2. Collect communication data
   - $\Longrightarrow$ hard to do as session key is different for every session
3. Store and forward data
   - $\Longrightarrow$ Nonce used by PC application and applet will protect against this
4. Modify and send data to the smart card of PC application
   - $\Longrightarrow$ Any modification of encrypted data would destroy it and can be rejected
   - $\Longrightarrow$ Use HMAC to ensure the integrity of data

THANK YOU