FACULTY
OF INFORMATICS
Masaryk University

jpass-smartcard code review

Adam Janovský, Marie William Gabriel Konan, Matěj Plch

jpass-smartcard

- works with the Java Card simulator
- database really processed through the card - nice!
- authors have not done testing on a real card
  - compilation errors
  - reported to the team, we fixed it together
  - applet has not worked on a card, reason unknown

FindBugs

- static code analysis
- total: 54 issues
  - 28 in the original version
  - Applet: 9
  - jpass.smartcard: 16
  - jpass.ui: 1
- dead stores, unused methods…
- Unread field Applet.SecureChannel.keyMAC
  - $\Longrightarrow$ no message integrity?

Symmetric Key Derivation

- shared secret established using Diffie-Hellman key exchange
- result truncated to 256 bits and directly used as an AES key
- result of DH should be first hashed and after that used as a symmetric key

PIN Verification

- redundant check for remaining tries
- method OwnerPIN.check does that for you

```
if(ins == INS_VERIFYPIN) {
  if((short)m_pin.getTriesRemaining() > (short)0) {
    VerifyPIN(apdu);
  } else {
    ISOException.throwIt(SW_BAD_PIN);
  }
}
```

PIN Verification II

- redundant call for PIN check

```
// in method VerifyPIN
if (m_pin.check(apdubuf,
        ISO7816.OFFSET_CDATA,
        (byte) dataLen) == false) {
    ISOException.throwIt(SW_BAD_PIN);
} else {
    m_pin.reset();
    m_pin.check(apdubuf,
        ISO7816.OFFSET_CDATA,
        (byte) dataLen);
}
```

PIN Verification III

- method `OwnerPIN.reset` only called right between successful PIN verifications (previous slide)
- `reset` is not called in methods `select` or `deselect`
- after successful PIN verification card stays in an authenticated state
- after authentication, another application can select the applet and do commands without PIN authentication

Padding oracle

- padding functionality potentionally vulnerable to padding oracle attack

```
// in method CBC_BULK_Finish
short outLen = m_bulk_cbc_cipher.doFinal(... // 16
short unpad = padding.unpad(... // [-1..15]
if(unpad >= 0) {
        outLen = unpad;
}
Util.arrayCopyNonAtomic(...
this.wrapAndSend(apdu, outLen); // 16b => invalid pad
```