
AESCrypt file Encryption Software – JavaCard based security over Secure Channel

Surendra Sharma
UCO - 459205

Balaji Kommuru
UCO - 459208

Ashwin Yakkundi
UCO - 459202

Agenda

- 1. Project Brief**
- 2. AEScrypt - File Encryption with Password Input**
- 3. Design of JavaCard Based secure HSM**
- 4. Security Components of Modified AEScrypt**
- 5. Secure Storage of Secrets in Trusted Environment**
- 6. Secure Session key Establishment**
- 7. Secure Channel Establishment for Password Retrieval**

Project Brief - AESCrypt

1. Password based file encryption program in Java
2. Opensource (Github) project with command line input of file, encryption/decryption switch and password

3. Existing Usage:-

```
java -cp bin es.vocali.util.AESCrypt e|d password  
fromPathFile toPathFile
```

4. Modified usage:-

```
java -jar SimpleAPDU.jar e|d password fromPathFile  
toPathFile
```

AEScrypt

https://github.com/FreedomBen/aescrypt/tree/master/ja

ch for PDF... PDF Viewer: Enter a URL of PDF file...

FreedomBen / aescrypt

Watch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Pulse Graphs

Branch: master aescrypt / java / Create new file Find file History

FreedomBen Add Java files Latest commit 6163fdf on 11 Oct 2013

..		
bin/es/vocali/util	Add Java files	4 years ago
javadoc	Add Java files	4 years ago
src/es/vocali/util	Add Java files	4 years ago
test	Add Java files	4 years ago
LICENSE-2.0.txt	Add Java files	4 years ago
README.txt	Add Java Add Java files	4 years ago

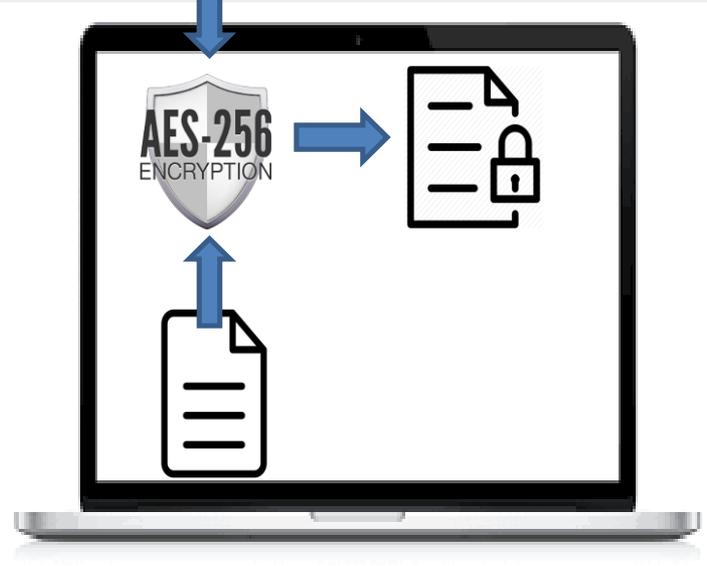
README.txt

AEScript

```
surendra@LXBOX-SS: /media/surendra/DATA/MasarykCourse/1_PV204_SecTech_8/Project_SC/aescript-java-3.2
File Edit View Search Terminal Help
[DEBUG] Last block size mod 16: 1
[DEBUG] HMAC2: [47, -37, -17, 33, -98, 8, 64, -71, 53, 127, -20, -111, 60, -88, -120, 103, 12, -101, -2, -26, 121, -43, 1, -120, -1surendra@LXBOX-SS:/media/surendra/DAT
A/MasarykCourse/1_PV204_SecTech_8/Project_SC/aescript-java-3.2$ clear

surendra@LXBOX-SS: /media/surendra/DATA/MasarykCourse/1_PV204_SecTech_8/Project_SC/aescript-java-3.2$ java -cp bin es.vocali.util.AEScript e password TestDocument TestEnc
rypt.enc
[DEBUG] Using password: [112, 0, 87, 0, 115, 0, 115, 0, 119, 0, 111, 0, 114, 0, 100, 0]
[DEBUG] Opened for reading: TestDocument
[DEBUG] Opened for writing: TestEncrypt.enc
[DEBUG] IV1: [56, -108, -93, 8, -128, 3, -91, 42, 44, -120, -21, 118, -110, 91, 24, 35]
[DEBUG] AES1: [-41, -34, 2, 111, -32, 31, -111, -102, -64, -4, 14, -77, 127, -65, -65, -63, 31, 109, 7, -7, -16, 3, 80, -64, -27, 76, 45, 80, 11, 112, -36, -82]
[DEBUG] IV2: [40, -99, 1, 85, 75, 14, 11, 56, 109, 101, 34, 16, 112, 17, 110, 11]
[DEBUG] AES2: [41, -87, -39, 24, 117, -107, 45, 43, -43, 0, -61, 76, -107, -99, 118, -66, 43, 37, -81, 112, 20, -90, 126, -116, -109, -68, -2, -60, -98, 110, -17, -29]
[DEBUG] IV2 + AES2 ciphertext: [40, 73, 90, 69, -49, -104, 11, -117, 14, -82, 39, -77, -60, -30, -32, 10, 53, 39, -27, -7, -122, 79, -58, -91, 68, -32, 115, -62, -66, -
112, 27, 22, -78, -38, 22, 78, 103, 13, -19, -35, -95, 59, -40, 74, 41, 58, 4, 111]
[DEBUG] HMAC1: [93, 67, -39, 127, -85, 30, -111, -32, -123, -86, -28, 56, -59, -84, 9, -12, 13, -120, -45, -50, 10, -126, -98, -106, 67, -35, 69, -61, -122, -116, -60,
85]
[DEBUG] Last block size mod 16: 1
[DEBUG] HMAC2: [118, -108, 87, 99, -64, 117, 113, 48, 13, 108, -80, -94, -99, 53, 81, 8, 52, -68, -71, 106, -92, -102, -93, -101, -77, 102, 23, -36, -122, -42, 83, 111]
surendra@LXBOX-SS: /media/surendra/DATA/MasarykCourse/1_PV204_SecTech_8/Project_SC/aescript-java-3.2$ java -cp bin es.vocali.util.AEScript d password TestEncrypt.enc Tes
tDecrypt.dec
[DEBUG] Using password: [112, 0, 87, 0, 115, 0, 115, 0, 119, 0, 111, 0, 114, 0, 100, 0]
[DEBUG] Opened for reading: TestEncrypt.enc
[DEBUG] Opened for writing: TestDecrypt.dec
[DEBUG] Version: 2
[DEBUG] Skipped extension sized: 0
[DEBUG] IV1: [56, -108, -93, 8, -128, 3, -91, 42, 44, -120, -21, 118, -110, 91, 24, 35]
[DEBUG] AES1: [-41, -34, 2, 111, -32, 31, -111, -102, -64, -4, 14, -77, 127, -65, -65, -63, 31, 109, 7, -7, -16, 3, 80, -64, -27, 76, 45, 80, 11, 112, -36, -82]
[DEBUG] IV2 + AES2 ciphertext: [40, 73, 90, 69, -49, -104, 11, -117, 14, -82, 39, -77, -60, -30, -32, 10, 53, 39, -27, -7, -122, 79, -58, -91, 68, -32, 115, -62, -66, -
112, 27, 22, -78, -38, 22, 78, 103, 13, -19, -35, -95, 59, -40, 74, 41, 58, 4, 111]
[DEBUG] IV2: [40, -99, 1, 85, 75, 14, 11, 56, 109, 101, 34, 16, 112, 17, 110, 11]
[DEBUG] AES2: [41, -87, -39, 24, 117, -107, 45, 43, -43, 0, -61, 76, -107, -99, 118, -66, 43, 37, -81, 112, 20, -90, 126, -116, -109, -68, -2, -60, -98, 110, -17, -29]
[DEBUG] HMAC1: [93, 67, -39, 127, -85, 30, -111, -32, -123, -86, -28, 56, -59, -84, 9, -12, 13, -120, -45, -50, 10, -126, -98, -106, 67, -35, 69, -61, -122, -116, -60,
85]
[DEBUG] Payload size: 16672
[DEBUG] Last block size mod 16: 1
[DEBUG] HMAC2: [118, -108, 87, 99, -64, 117, 113, 48, 13, 108, -80, -94, -99, 53, 81, 8, 52, -68, -71, 106, -92, -102, -93, -101, -77, 102, 23, -36, -122, -42, 83, 111]
surendra@LXBOX-SS: /media/surendra/DATA/MasarykCourse/1_PV204_SecTech_8/Project_SC/aescript-java-3.2$ diff TestEncrypt.enc TestDecrypt.dec
Binary files TestEncrypt.enc and TestDecrypt.dec differ
surendra@LXBOX-SS: /media/surendra/DATA/MasarykCourse/1_PV204_SecTech_8/Project_SC/aescript-java-3.2$ diff TestDocument TestDecrypt.dec
surendra@LXBOX-SS: /media/surendra/DATA/MasarykCourse/1_PV204_SecTech_8/Project_SC/aescript-java-3.2$
```

AEScript



AEScript - Existing



AEScript - Modified

Design of JavaCard based HSM

Components of Design

1. **Security in Storage** - Long Term Symmetric Key, PIN, AEScript Application Password, PC_ID, JC_ID
2. **Security in Transit** :-
 - (a) Secure Session Key Establishment
 - (b) Secure Channel for Password retrieval

Design of JavaCard based HSM

Components of Design

3. Authentication

- Mutual Authentication using PCID & JCID (Key Establishment phase)
- PIN verification for Setting New PIN & App Password retrieval

4. Integrity:-

- Implementation of session Key based HMAC
- Integrity check of Command APDU and Password Response through HMAC

Security of Storage - Trusted Environment

1. User PIN Set in Secure Trusted Environment
2. Long Term Symmetric key also Stored as AES key in JavaCard (Set in Trusted environment)
3. AEScrypt Application Password Stored as AES Key in JavaCard (Set in Trusted environment)
4. **keyType.setKey()** and **keyType.getKey()** functions for Password/ Long term Key Handling in JavaCard

Security of Storage - Trusted Environment

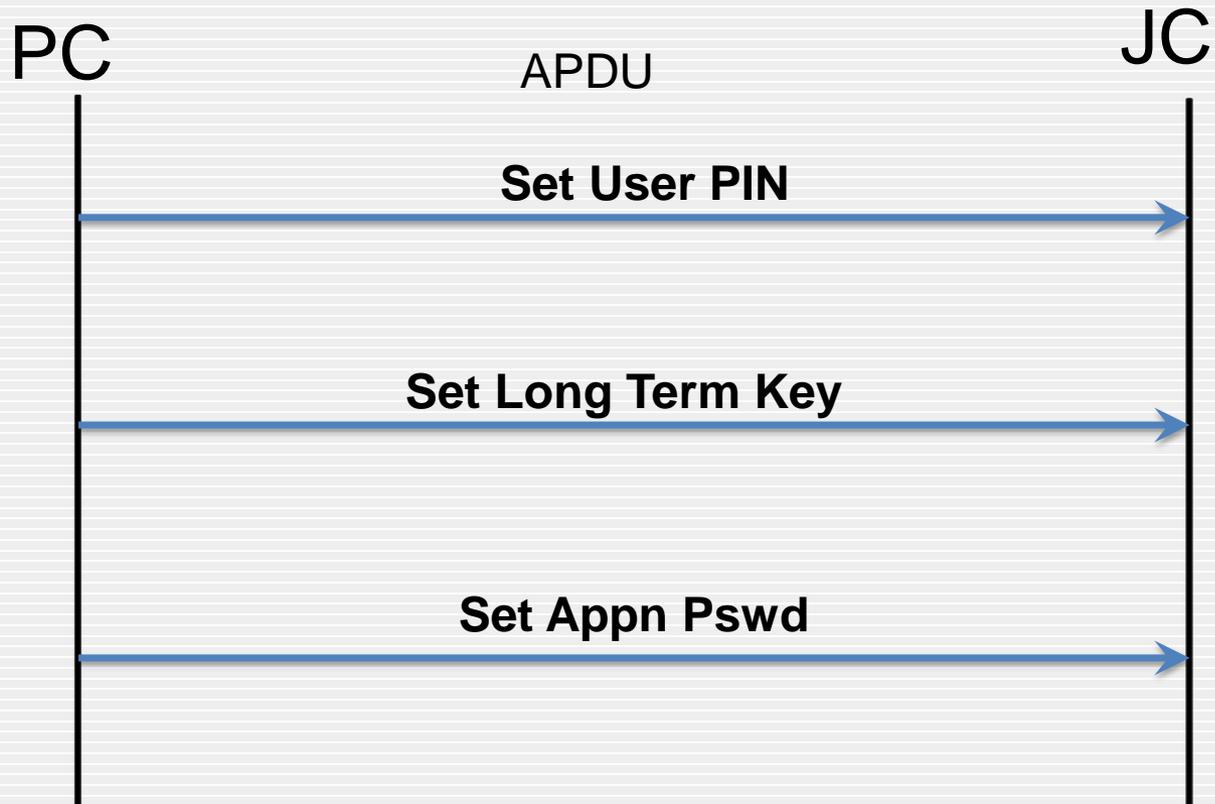


Figure: Setting Long Term Key & App Password in JavaCard

Security Channel - Key Establishment

1. Use of Long Term Key for Session Key establishment
2. Session Keys generated based on Random Nonces generated on both the PC and JC ends
3. PCID and JCID used for mutual authentication
4. **Implicit authentication by Encryption / Decryption and verification of Nonces at both ends**

Security Channel - Key Establishment

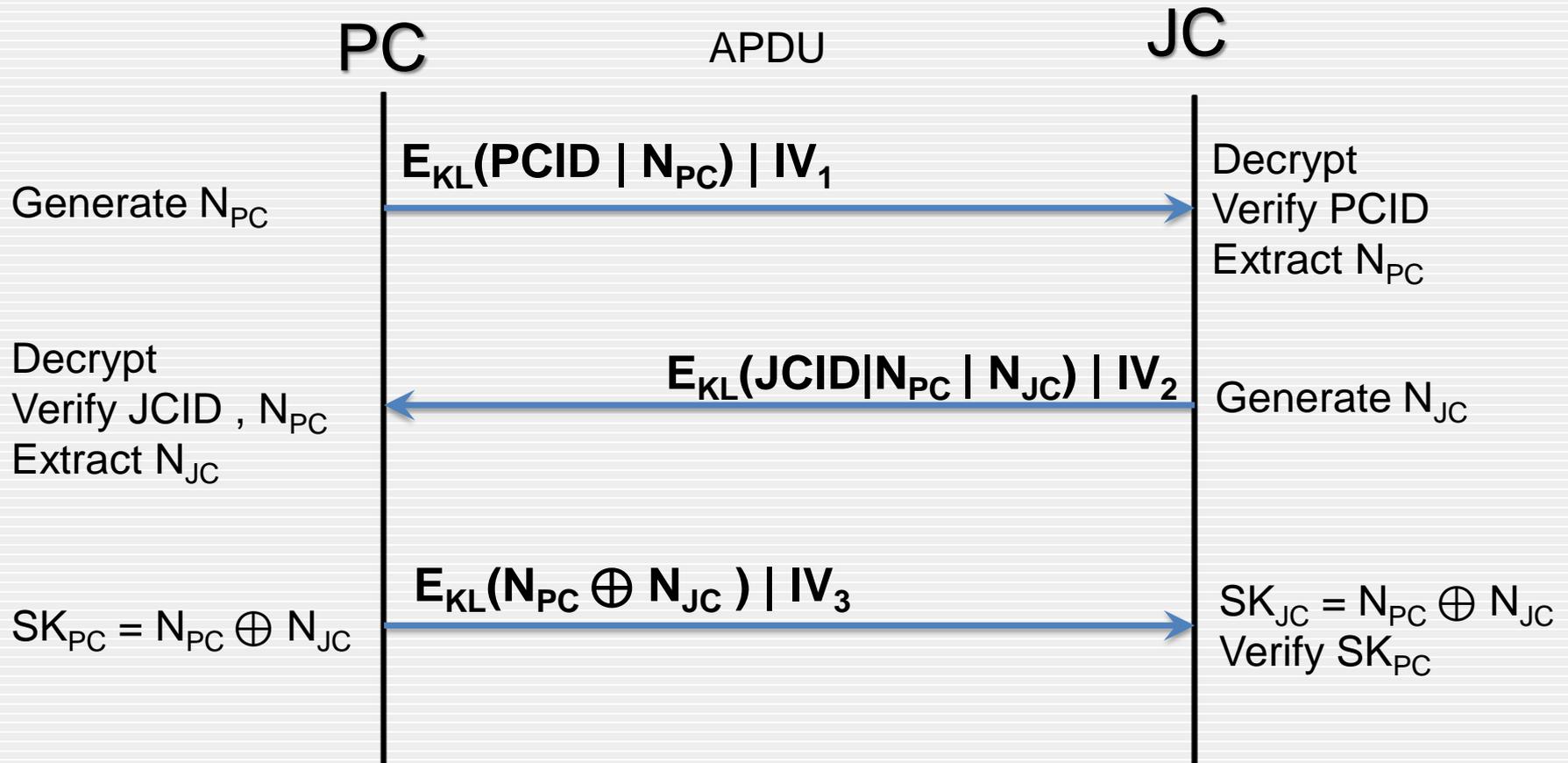


Figure: Secure Session Key Establishment

Security Channel - Secure Password retrieval

1. Secure Encrypted Channel - Symmetric Session Key
2. PIN verification for Command APDU for Password Req
3. Secrets of PIN and Password - always encrypted in channel
4. Integrity of the Channel protected by SHA2 based HMAC

Security Channel - Secure Password retrieval

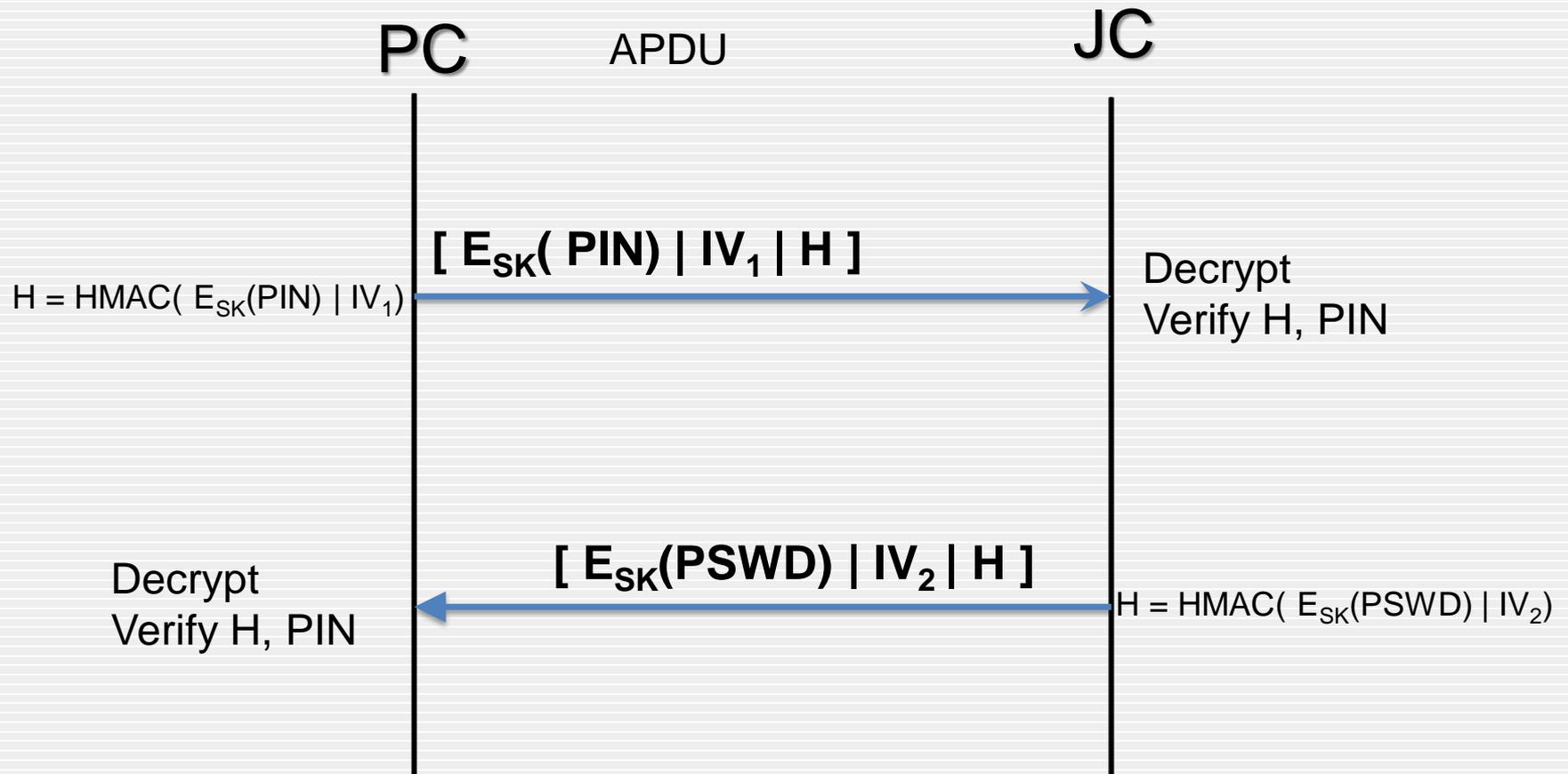


Figure: Extracting Password in Secure Channel

References

1. <https://github.com/FreedomBen/aescrypt/tree/master/java>
2. <https://docs.oracle.com/javacard/3.0.5/api/javacardx/crypto/package-frame.html>
3. <https://docs.oracle.com/javacard/3.0.5/api/javacardx/crypto/Cipher.html>
4. <https://docs.oracle.com/javacard/3.0.5/api/javacard/security/package-summary.html>
5. https://en.wikipedia.org/wiki/Secure_channel
6. https://en.wikipedia.org/wiki/Hash-based_message_authentication_code
7. https://en.wikipedia.org/wiki/Key_exchange

Questions & Discussions



Source : freeppt.net & techstories.wordpress.com