Project Report Phase III

Cryptsetup-Javacard

Team Members: Hitesh Lilhare (476356), Urvek C Shah (476355)

1. Issues & Code Improvement

Issue	Improved Code
States with consecutive values	STATE_IDLE = (short)0x4CCC; STATE_KEY_ESTABILISHED = (short)0x5555; STATE_AUTHENTICATED = (short)0x2AAA;

Single time state value validation (Before doing sensitive operations)	`// Double checking sensitive operation`
	`if (state == STATE_AUTHENTICATED) {`
	`if (state != STATE_AUTHENTICATED) {`
	`ISOException.throwIt(ISO7816.SW_COMMAND_NOT_ALLOWED);`
	`} else {`
	`masterPassword.update(buffer, inOffset, (byte) length);`
	`}`
	`} else {`
	`ISOException.throwIt(ISO7816.SW_COMMAND_NOT_ALLOWED);`
	`}`
	`return 0;`
	`}`

Boolean comparison-based operation	`// chnaged boolean comparision to byte comparision to counter fault induction attack.`
	`private static byte arraysEqual(byte[] array1, short offset1, byte[] array2, short offset2, short length) {`
	`byte different = (byte)0x00;`
	`for (short i = 0; i < length; i++) {`
	`different |= (byte)(array1[(short)(offset1 + i)] ^ array2[(short)(offset2 + i)]);`
	`}`
	`if(different == (byte)0x00){`
	`return SUCCESS;`
	`}else{`
	`return FAILURE;`
	`}`
	`//return different == (byte)0x00;`
	`}`

Pull request for the above-mentioned code improvement has been generated for the original repository.

2. Profiling APDU execution

Sr. No	Command	Avg Time (ns)
1	INS_GETPUBKEY (0x50)	26619862
2	INS_HANDSHAKE (0x51)	1811472034
3	INS_COMMAND (0x52) + CMD_AUTH (0x00)	158243573
4	INS_COMMAND (0x52) + CMD_CHANGEPW (0x01)	174816345
5	INS_COMMAND (0x52) + CMD_GENKEY (0x02)	186454739
6	INS_COMMAND (0x52) + CMD_STOREKEY (0x03)	219109635
7	INS_COMMAND (0x52) + CMD_LOADKEY (0x04)	198403275
8	INS_COMMAND (0x52) + CMD_DELKEY (0x05)	344636378
9	INS_COMMAND (0x52) + CMD_CLOSE (0x06)	149424505

3. **Profiling using JCProfiler**
   a. Successfully generated TRAPS
   b. Integrate with Applet code
   c. Load Applet to card
   d. Preparation of Client Application

   But did not able to use it because we did not find option wherein we can provide custom build APDU to the generated client application.

4. **Manual Profiling**

1. Profiling of fillPubKey Function

```
private short fillPubKey(byte[] buffer, short offset) throws ISOException {
    short totalSize = 0;
    RSAPublicKey pubKey = (RSAPublicKey)signingKeyPair.getPublic(); // 9909089 ns

    short modSizeOffset = offset;
    short modOffset = (short)(offset + (short)2);
    short modSize = pubKey.getModulus(buffer, modOffset); //2486662 ns
      if(fill_pub_ctr==1){
          ISOException.throwIt(ISO7816.SW_CLA_NOT_SUPPORTED);
          fill_pub_ctr++;
      }
    writeShort(buffer, modSizeOffset, modSize); //2330388 ns
    totalSize += (short)2;
    totalSize += modSize;

    short expSizeOffset = (short)(modOffset + modSize);
    short expOffset = (short)(expSizeOffset + (short)2);
    short expSize = pubKey.getExponent(buffer, expOffset); //178496 ns

    writeShort(buffer, expSizeOffset, expSize); //621202 ns

    totalSize += (short)2;
    totalSize += expSize;

    return totalSize;
}
```

2. Profiling of commandAuth function

```
private short commandAuth(byte[] buffer, short inOffset, short length, short outOffset) {
    if (state != STATE_KEY_ESTABILISHED) {
        ISOException.throwIt(ISO7816.SW_COMMAND_NOT_ALLOWED);
    }

    if (length > MAX_PW_LEN) {
        ISOException.throwIt(ISO7816.SW_WRONG_LENGTH);
    }

    if (!masterPassword.check(buffer, inOffset, (byte)length)) { // 158787203 ns
        resetSession();
        ISOException.throwIt(ISO7816.SW_SECURITY_STATUS_NOT_SATISFIED);
    }

    state = STATE_AUTHENTICATED;
    return 0;
}
```

3. Profiling of commandChangePw

```
private short commandChangePw(byte[] buffer, short inOffset, short length, short outOffset) {
    if (state != STATE_AUTHENTICATED) {
        ISOException.throwIt(ISO7816.SW_COMMAND_NOT_ALLOWED);
    }

    if (length > MAX_PW_LEN) {
        ISOException.throwIt(ISO7816.SW_WRONG_LENGTH);
    }

    masterPassword.update(buffer, inOffset, (byte)length); // 84792385 ns
    ISOException.throwIt(ISO7816.SW_INS_NOT_SUPPORTED);
    return 0;
}
```

4. Profiling of commandClose function

```
private short commandClose(byte[] buffer, short inOffset, short length, short outOffset) {
    resetSession(); //148630484 ns
    return 0;
}
```

5. **Learning:**

Problem Statement: Client application was not able to make connection to the java card whereas GPPro was able to connect to the card.

Option Explored:

i. We instigated on different way for connecting with card wherein we explored the way how GPPro is connecting to the java crad. Wherein we modified the client code which is responsible for making connection. In this approach we were able to get very little success in the form that we are successfully able to connect with javacard for one functionality of card. But this was not the end of trouble, because the code which was responsible for making connection, was very tightly integrated with the other functionality of applet. Therefore, we shifted our focus on other possibility.

ii. We invested lots of over time in getting applet functionality run on real card in which we installed about 4-5 different ubuntu LTS versions as virtual machine and host machine but did not succeeded.

iii. It was very disappointing after two failures, but we did not loose the hope and tried for pcsclite driver. Wherein we took latest code of the pcsclite, compile & installed it but again no success. Then we focus on the library used by the pcsclite driver and finally we got the solution wherein we can specify the path of the library to the java virtual machine while execution the client application. And the option is **-Dsun.security.smartcardio.library=/lib/i386-linux-gnu/libpcsclite.so.1**