Open Source JavaCard Development

FOUND ON JAVA CARD.PRO
Smart(Java)Card

... 

What & Why
What - smart card

- Tiny PC without Human Interface capabilities
  - **CPU**: 16b/32b RISC @ handful of MhZ
    - Math co-processor: RSA/DES/AES/ECC
  - **RAM**: X KB
  - **HDD**: XX..XXX KB (EEPROM)
  - **NET**: "Ethernet" (contact) or "WiFi" (contactless)
    - "The size of a Raspberry Pi but with specs worse than XT!"
Application Protocol Data Unit
BIBO

Bytes go In & Bytes come Out
(request - response)
What - JavaCard

- **BASIC in BIOS**: Java VM
- **DOS**: App(let) manager (GlobalPlatform)
Choose your Weapon

- **ASM / C** (OpenCard* by CryptoExperts)
- **C** (MULTOS)
- **BASIC** (BasicCard by ZeitControl)
- **Java** (JavaCard)
Why - JavaCard

- Meaningful abstraction layer
- Commodity platform
  - Multiple vendors
- Multiple applications
- "Open platform" - Oracle ...
- "Portable" - Java ...
From **Academia** and **Business** to **Open Source Developers**
Step 1
Get the necessary hardware
Open JavaCard

• At least 3 online shops in EU (in English)
  • + Canada, US

• Must be OPEN Java Card
  • No "rooting" yet ;(

• Form factor: ID-1 ("credit card") or USB token

• From 5€ (Feitian) to 50€ (NXP)

• javacard.pro / Google:

"JavaCard Buyer's Guide of 2015"
What to ask / look out for

- **JavaCard version**: 2.2.2, with 3.0.1 and 3.0.4 becoming popular. **Bigger is better** but for actual rollout check the necessary features, algorithms and maybe try to aim for 2.2.2-compatible code.

- **GlobalPlatform version**: 2.1.1 and 2.2 are common. This relates to loading your application to the card and 2.1.1 is sufficient, while 2.2 adds SCP03 support, which uses AES instead of 3DES.

- **EEPROM size**: 64K, 72K, 128K, 144K and bigger sizes are common. **Bigger is better**, but when actually rolling out your card, choose a size that is with optimal price/size depending on actual requirements.

- **GlobalPlatform default keys** (or test keys, with the value 404142434445464748494A4B4C4D4E4F or 40...4F for short): only if you get default test keys (or otherwise known keys) shall you be able to load applications to the card. **You shall not be able to load your application to the card without the keys.** Always be sure to ask for test keys for sample cards!

- **Contact/Contactless interface**: for creating NFC applications you want to get a card with **dual interface** or even contactless-only.

- **Proximity cards**: for opening doors, usually a different chip is present on the card for this single purpose. But a vendor can usually combine necessary physical access cards with a suitable JavaCard chip module.

- **Common Criteria / FIPS validation**: most **serious** smart cards have some form of certification. CC EAL5+ and FIPS 140 level 3 being common for the JavaCard part. **Bigger is better** but keep in mind, that "the use of a validated cryptographic module in a computer or telecommunications system does not guarantee the security of the overall system." (excerpt from FIPS 140-2)

- **GlobalPlatform lifecycle**: should OP_READY, but keep in mind that certain pre-personalization steps (like changing physical characteristics of the chip) can only be done before this state and are usually proprietary. Keep this in mind when actually rolling out.

Manufacturers
Supported cards (with a confirmed shop link)

- SmartCafe Expert 3.2 72K JavaCard 2.2.1/GlobalPlatform 2.1.1 12€
  - Virgin cars require unlocking with "-env"
  - and other products with same name but different EEPROM + connectivity:

- J COP J3D081 v2.4.2 JavaCard 3.0.1/GlobalPlatform 2.2 40€
  - and other products with same
    - http://www.cryptoshop.com/products/smartcards/jcop-31-v2-4-1-72k.html
    - JCOP v2.4.1 NXP J3A080 Dual Interface Card (10 pcs.) 110€

- Gemalto
  - GEMALTO IDCORE 10 - GEMALTO TOP IM GX4 JavaCard 2.2.1/GlobalPlatform 2.0.1 16€
  - Virgin cars require unlocking with "-visa2 -key"
  - Instead of a reasonable error code a failed transaction is returned on errors.
    - GEMALTO IDCORE 3010 CC / TOP DM CC 25€

- Feitian JavaCard-s
Smart Card Reader

- Any* will work (Contact)
  - Ludovic Rousseau's **USB CCID** driver (298/323)
  - Google: "Readers sorted by 'section' field"
- Carefully consider contactless
- **PC/SC is not a hardware standard!**
Step 2
Prepare your tools
1. Normal **Java development**
   - Favourite editor, IDE, compiler
   - **Catch**: running requires emulation

2. **Conversion** into card-loadable format (CAP file)
   - Against Oracle's JavaCard SDK

3. **Loading** onto card
   - Using GlobalPlatform
JavaCard SDK

- From Oracle ...
  - No OpenJavaCardSDK :(
- Java components are cross-platform
- Suitable max version depends on card version.
ant-javacard

• ANT task for turning Java source code into a loadable CAP file
  
  • **Any platform** (Linux, OS X, Windows)
  
  • **Any version** of JavaCard SDK
  
  • Simple. Easy to use. **Seriously.**
<property name="src" location="applet/src"/>
<property name="build" location="applet/bin"/>
<property name="test.build" location="test/bin"/>
<property name="test.lib" location="test/lib"/>
<!-- Load user specified extra properties -->
<property file="${user.home}/javacard.properties"/>
<property name="JAVA_PACKAGE" value="openpgpcard"/>
<property name="JAVA_PACKAGE_DIR" value="openpgpcard/"/>
<property name="APPLET_NAME" value="OpenPGPApplet"/>
<property name="PACKAGE_AID" value="0xd2:0x76:0x00:0x01:0x24:0x01"/>
<property name="APPLET_AID" value="0xd2:0x76:0x00:0x01:0x24:0x01:0x02:0x00:0x00:0x00:0x00"/>
<property name="VERSION" value="0.1"/>
	<target name="init">
		<mkdir dir="${build}/"/>
		<mkdir dir="${test.build}/output"/>
	</target>
	<target name="compile" depends="init" description="compile the source" unless="test:
		<javac srcdir="${src}" destdir="${build}" includeantruntime="false" source="1.5" t
		<classpath>
			<path element="${JAVACARD_HOME}/lib/api.jar"/>
		</classpath>
		</javac>
	</target>
	<target depends="compile" name="convert" description="convert to .cap" unless="test:
		<java classname="com.sun.javacard.converter.Converter" fork="true" failonerror="tr
		<arg line="-classdir ${build}"/>
		<arg line="-verbose"/>
		<arg line="-exportpath ${JAVACARD_HOME}/api_export_files"/>
		<arg line="-out CAP JCA EXP"/>
		<arg line="-applet ${APPLET_AID} ${APPLET_NAME}"/>
		<arg line="${JAVA_PACKAGE} ${PACKAGE_AID} ${VERSION}"/>
		<classpath>
			<path element="${JAVACARD_HOME}/lib/converter.jar"/>
			<path element="${JAVACARD_HOME}/lib/offcardverifier.jar"/>
		</classpath>
		</java>
	</target>
</target>
TYPICAL APPLE PRODUCT...

A GOOGLE PRODUCT...

YOUR COMPANY’S APP...

STUFFTHATHAPPENS.COM BY ERIC BURKE
Similar projects

- standard JavaCard SDK Ant tasks
  - : as cumbersome to use as the command line utilities
  - : (not declarative/DWIM enough
  - :) very explicit interface with all details exposed
- gradle-javacard (Apache 2.0) - https://github.com/fidesmo/gradle-javacard
  - :) nice declarative interface
  - :( requires gradle (40M download)
  - :( JavaCard 2.2.2 only
- EclipseJCDE (Eclipse 1.0) - http://eclipse-jcde.sourceforge.net/
  - :( JavaCard 2.2.2 only
  - :( not possible to integrate in CI - depends on eclipse
  - :( essentially an Eclipse GUI wrapper for JC SDK
- JCOP Tools
  - :( not open source
- NetBeans IDE JC support
  - :( not possible to integrate into CI
  - :( JavaCard 3.0 only
  - :( Netbeans, not cross platform
- Maven2 task from FedICT (LGPL3) - https://code.google.com/p/eid-quick-key-toolset
  - :( Maven downloads half the internet before even simple tasks
  - :( JavaCard 2.2.2 only
- Ant script files with templates
  - :( XML is a very bad and verbose programming environment
<taskdef name="javacard" classname="pro.javacard.ant.JavaCard" classpath="lib/ant-javacard">
  <javacard>
    <cap jckit="${JAVACARD_HOME}" output="MyApplet.cap" sources="src/openpgpcard">
      <applet class="openpgpcard.OpenPGPApplet" aid="D276000124010200000000000000010000"/>
    </cap>
  </javacard>
</taskdef>
Application IDentifier
GlobalPlatform

• Every package (CAP file) has an AID
  • Each applet (class) has an AID
• Every on-card entity has an AID
  • Packages and classes and instances
• 5..16 bytes (5+11)
GlobalPlatformPro

• Easy to use Java tool to:
  • LOAD CAP files to the card
  • INSTALL applets (AID-s)
  • CREATE applet instances (AID-s)
  • DELETE applets and packages (AID-s)
  • Add/change/delete keys
  • And more ...
Lock/Unlock
$ gp -l
AID: A000000003000000 (|........|)
   ISD SECURED: Security Domain, Card lock, Card terminate, Default selected, CVM (PIN) management

AID: A0000000035350 (|.....SP|)
   ExM LOADED: (none)
   A000000003535041 (|.....SPA|)

$ gp -lock B4F75CE0A95EA3F86BBD051CB77C0FAE
Card locked with: DES3:B4F75CE0A95EA3F86BBD051CB77C0FAE
Write this down, DO NOT FORGET/LOSE IT!

$ gp -l
openkms gp.GPException: STRICT WARNING: Card cryptogram invalid!
Card: CC73F92AD03A131D
Host: A358609D53744EEB
!!! DO NOT RE-TRY THE SAME COMMAND/KEYS OR YOU MAY BRICK YOUR CARD !!!
   at openkms gp.GlobalPlatform.printStrictWarning(GlobalPlatform.java:156)
   at openkms gp.GlobalPlatform.openSecureChannel(GlobalPlatform.java:476)
   at openkms gp.GPTool.main(GPTool.java:348)

$ gp -key B4F75CE0A95EA3F86BBD051CB77C0FAE -unlock
Default DES3:404142434445464748494A4B4C4D4E4F set as master key.

$ gp -l
AID: A000000003000000 (|........|)
   ISD SECURED: Security Domain, Card lock, Card terminate, Default selected, CVM (PIN) management

AID: A0000000035350 (|.....SP|)
   ExM LOADED: (none)
   A000000003535041 (|.....SPA|)
Step 4
Learn, Learn, Learn
Read

- JavaCard API Specification
  - and Runtime Environment
  - Google: "JavaCard Tutorial"
- ISO 7816-4 (and javacard.framework.APDU)
  - Google: "University Smart Card Paper"
  - CLA/INS/P1/P2/Lc/Le/SW/0x9000
- ISO 7816/14443, ETSI, BSI, NFC, NIST etc etc
- Beware of outdated/wrong/irrelevant information on the web!
ISO 7816-7 (1999)
Structured Card Query Language
AppletPlayground

• "Ready to eat" dog food from the internet
  • Almost all open source applets that _may_ do _something_
• Import into Eclipse
• Build with ANT (eclipse/cmdline)
Included applets

- MuscleApplet - as was found in martinpaljak/MuscleApplet@d005f36209bddd7020bac0d783b228243126fd2f8 (BSD)
- CoolKeyApplet - r105 from http://svn.fedorahosted.org/svn/coolkey/trunk/applet (BSD/LGPL2.1)
- PKIApplet - r65 from http://svn.code.sf.net/p/javacardsign/code/pkiapplet/src (LGPL2.1)
- OpenPGPApplet - Yubico/ykneo-openpgp@ed928351994b053f3d87ec00ec4a9696d4ff20fe (GPL2)
- FluffyPGPApplet* - FluffyKaon/OpenPGP-Card@545da17f82ff4627758674bbcb0e602e959d9dd (GPL3)
- YkneoOath - Yubico/ykneo-oath (GPL3)
- PassportApplet - http://sourceforge.net/p/jmrt/code/HEAD/tree/trunk/passportapplet/ (LGPL3)
- BTChip* - LedgerHQ/btchipJC (AGPL3)
- NDEF - slomo/ndef-javacard (DO WHAT THE FUCK YOU WANT TO PUBLIC LICENSE :))
- BeID* - r62 (LGPL)
- OpenEMV - r3 from svn://svn.code.sf.net/p/openemv/code/trunk (LGPL2)
- ISOApplet - philipWendland/IsoApplet (GPL3)
- DriversLicense* r175 from svn://svn.code.sf.net/p/isodl/code/ (LGPL2)
- PLAID - (public license?, no source link)

Note: applets marked with * have obvious blocking errors (missing casts from int to short for 2.2.X target) removed from source.
Step 5
Engage with the Community
- Oracle JavaCard Forum / kenai.com: **dead**
- Stack Overflow: "javacard", "globalplatform", "smartcard" tags: **some life**
- OpenSC / pcsc-lite lists: **open source but no Java**
- GitHub: **depends**
- IRC: **#opensc**
javacard.pro