U.S. Government Smart Card Program: Interoperability Features of the GSC-IS

John Wack

NIST

Smart Card Interoperability Workshop, CTST 2003

2 PM, May 12, 2003

Overview

- Some basic terminology
- GSC-IS interoperability
- GSC-IS architecture
- The Basic Services Interface
- The Card Edge for File System Cards
- The Card Edge for VM Cards

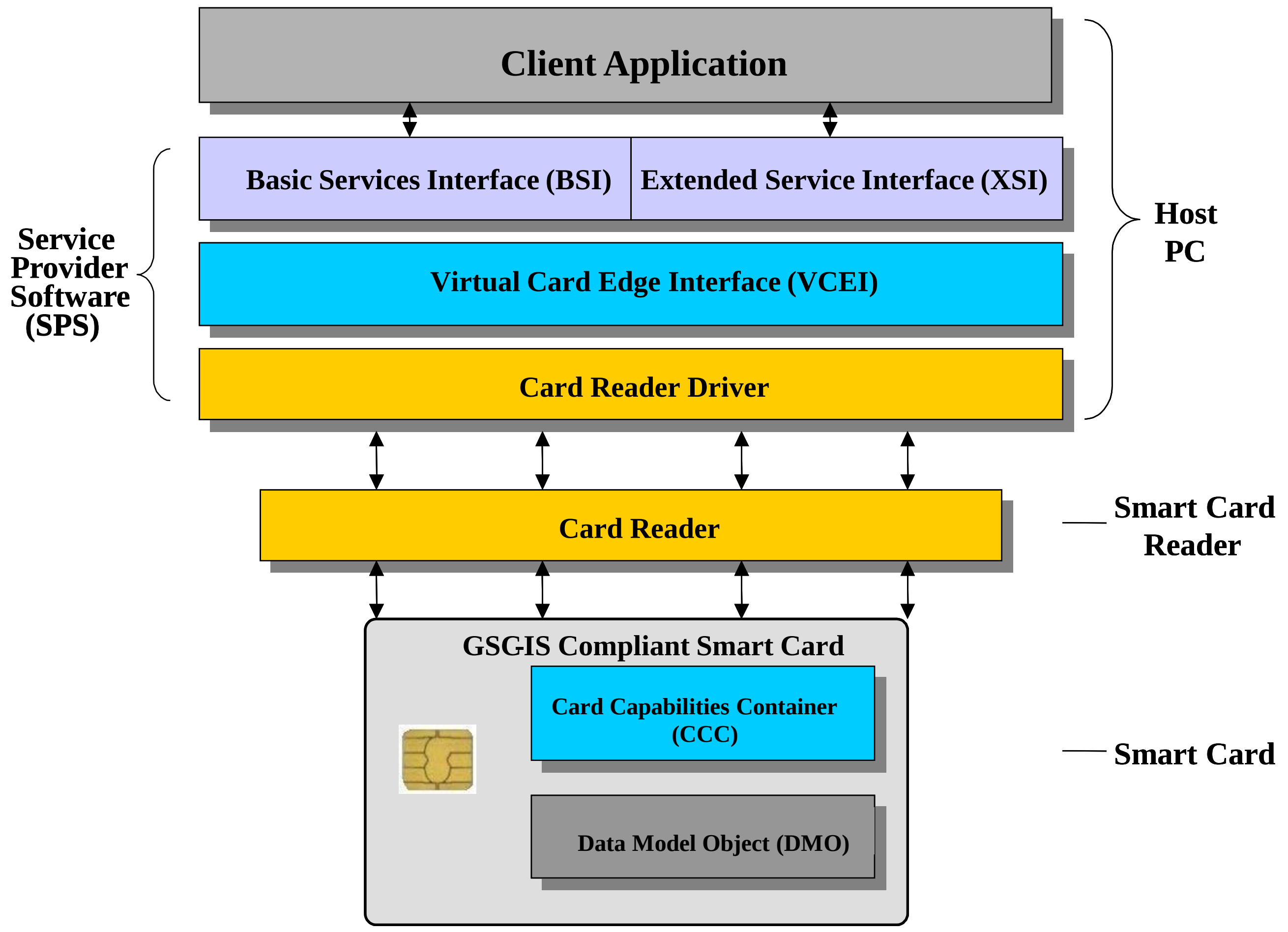
GSC-IS Terminology

- SPS – Service Provider Software
  - Talks to client apps and to smart card reader
  - Includes smart card reader driver (e.g., PC/SC)
- BSI – Basic Services Interface
  - Client application API
- XSI – Extended Services Interface
  - Vendor-specific
  - For card reader connects, non-interoperable functions
- CEI – Card Edge Interface
  - For File System and Virtual Machine cards
- CCC – Card Capability Container

What Is Interoperable Here?

- File reads and updates with access control
- Internal Authentication
- External Authentication
- PIN verification
- Signature generation
- And, an APDU pass-thru capability for sending raw, card-specific APDUs

GSC-IS Architecture


Client Application
Basic Services Interface (BSI)
Extended Service Interface (XSI)
Virtual Card Edge Interface (VCEI)
Card Reader Driver
Service Provider Software (SPS)
Host PC
Card Reader
Smart Card Reader
GSC-IS Compliant Smart Card
Card Capabilities Container (CCC)
Data Model Object (DMO)
Smart Card

Basic Services Interface

- A simple API, 21 functions
- Interface used by client applications to communicate with the card
- 3 categories
  - Utility – card connect, status, properties, pass-thru
  - Container – read, update, delete, properties
  - Access Control/Crypto – authentication, signature
- Only what is necessary, simple
- C and Java bindings available

BSI Utility Functions

- gscBsiUtilAcquireContext()
- gscBsiUtilConnect()
- gscBsiUtilDisconnect()
- gscBsiUtilGetVersion()
- gscBsiUtilGetCardProperties()
- gscBsiUtilGetCardStatus()
- gscBsiUtilGetExtendedErrorText()
- gscBsiUtilGetReaderList()
- gscBsiUtilPassthru()
- gscBsiUtilReleaseContext()

BSI Container Functions

- gscBsiGcDataCreate()
- gscBsiGcDataDelete()
- gscBsiGetContainerProperties()
- gscBsiGcReadTagList()
- gscBsiGcReadValue()
- gscBsiGcUpdateValue()

BSI Authentication/Crypto Functions

- gscBsiGetChallenge()
- gscBsiSkiInternalAuthenticate()
- gscBsiPkiCompute()
- gscBsiPkiGetCertificate()
- gscBsiGetCryptoProperties()

Extended Services Interface

- For functions outside the scope of GSC-IS interoperability
- For vendor-specific functions
- Examples include
  - Card reader power-up
  - Card initialization
  - Secure messaging

What Happens Next?

- The BSI functions “talk” to the Card Edge Interface, which talks to the card reader driver
- BSI-packaged data must be repackaged into APDU commands and vice versa
- As necessary, default APDUs must be mapped to card-specific APDUs

The Card Edge Interface

- The CEI is where card interoperability occurs
- 2 CEIs:
  - File system cards – 13 default ISO 7816-compliant APDUs
  - Virtual machine, Java cards – 12 APDUs
- CEI passes APDUs to card reader driver and card, and vice versa
- CEI must format the APDUs correctly according to which card is in use

Data Models

- 2 defined data models, GSC and CAC
- Others can be defined
- Models are composed of containers (files) with associated access-control conditions
- Containers are in SIMPLE TLV format
- Containers are accessed by client applications via AIDs
- AIDs are composed of RID + FID

Card Capabilities Container

- Only required container for a conformant card
- Contains various version information
- Contains encoded description of how card APDUs differ in syntax and operation from defaults for file system cards
- Includes access control requirements for containers
- Includes credential mappings

APDU Mapping – File System Cards

- There are 12 ISO 7816 default APDUs
- Card vendor encodes differences between default and card’s native APDUs in the CCC
- APDUs can differ in syntax as well as operation
- Encoding uses the CCC Grammar rules
- Grammar contains mechanisms for dealing with operational differences

CCC Grammar Features

- Native APDUs are encoded as 2-byte Tuples, as many as necessary for each APDU
- Only differences between native and default need to be encoded
- Pre- and post-command APDUs can be encoded, e.g., For Internal Authentication, Verify must precede and Get Challenge must follow
- Operational differences are encoded for APDU parameters and data fields

Descriptor Codes

- Encoded in Tuples to describe differences in APDU operation and execution
- Used to map differences in APDU parameter values and meanings
- Used to map differences in APDU input and output data fields
- Requires use of pre-defined table of descriptor codes with well-defined meanings

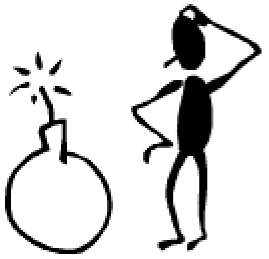
VM Card Edge

- VM cards can use either CEI, VM or FS
- APDU encoding is not applicable for VM
- VM-specific APDUs
- TLV containers on VM cards are split:
  - A TL buffer containing only the tags and lengths
  - A V buffer containing only the values
- Corresponding APDUs for reading and updating each buffer

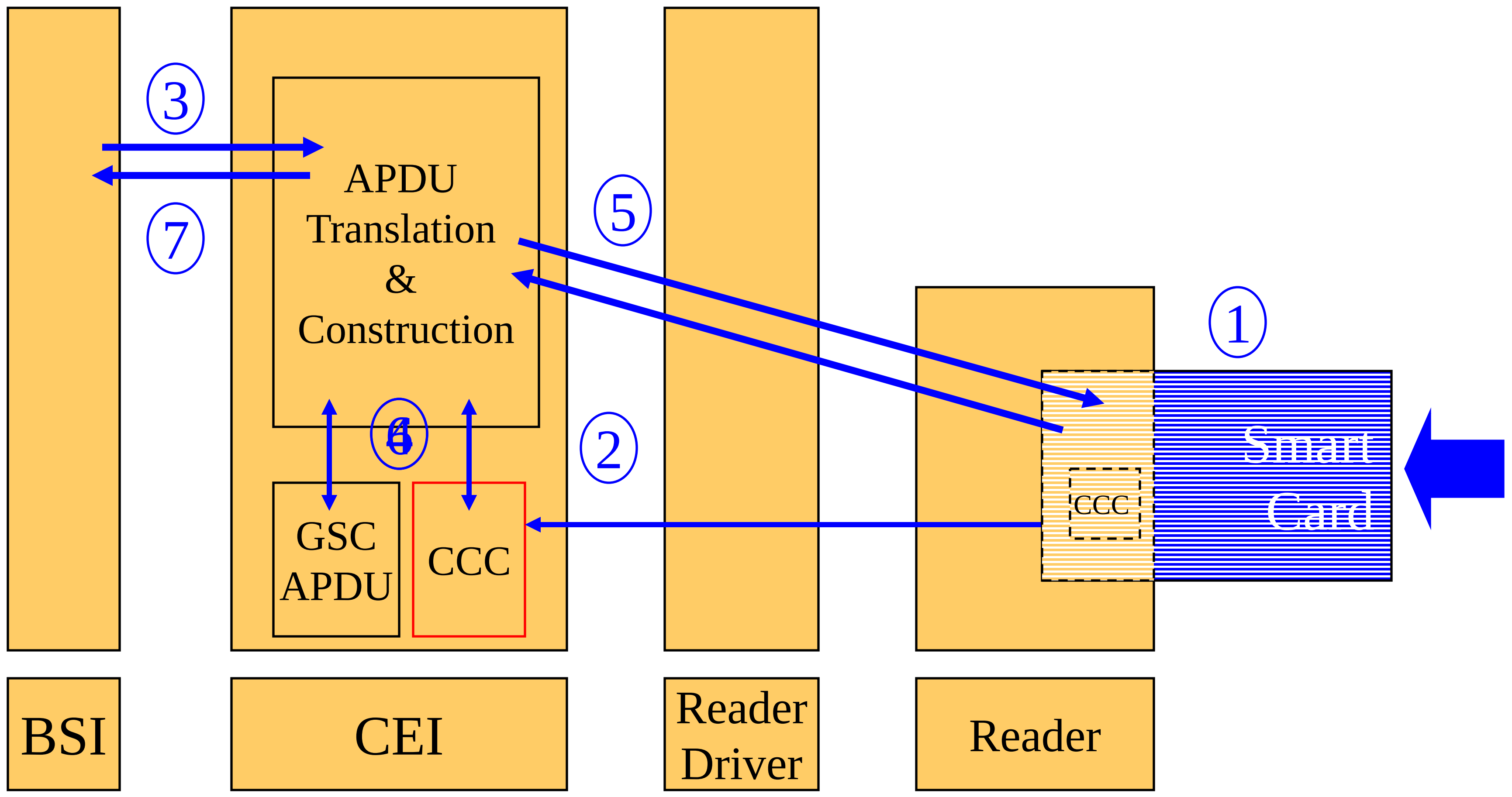
CEI Initialization

- Occurs when a GSC-IS-conformant SPM loads
- SPM locates the CCC and reads it according to following:

1. Detect whether VM or file system card;
2. If file system card edge being used,
3. If APDU encoding present,
4. Begin APDU mapping;



How Does It Work?



Contact Information

John Wack
U.S. Government Smart Card Program, NIST
Voice: 301.975.3411
email: john.wack@nist.gov

Website: http://smartcard.nist.gov (NISTIR 6887)