ISO/IEC 24727-3 Application Programming Interface

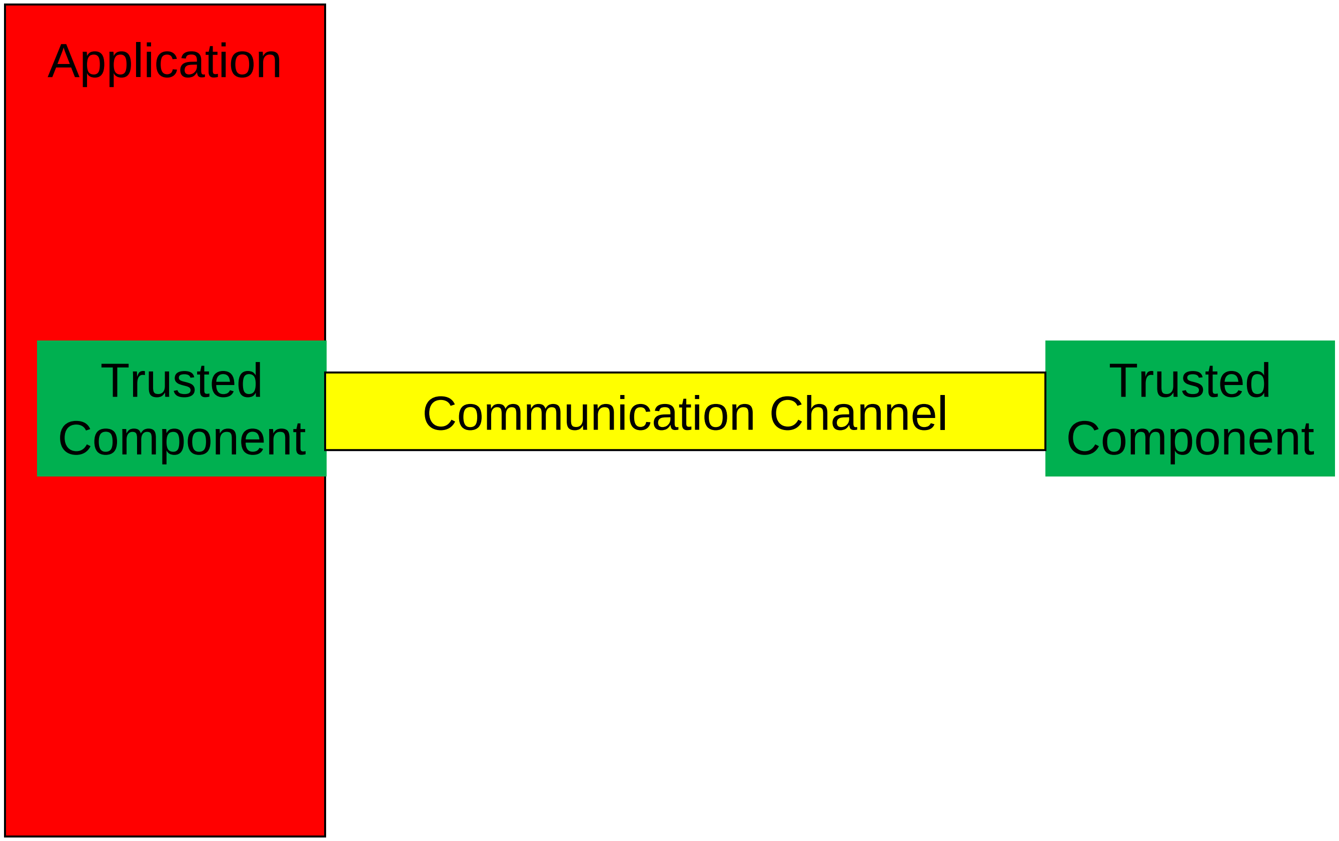
Session Objectives

- Review the API services
  - Path
  - Connection
  - Discovery
  - Authentication and Access Control
  - Use (Read, Write, Sign, etc.)
  - Conclusion (CloseSession, Disconnect)
- Consider the central role of differential-identity

A Client-Application's Token Functions

- Find it
- Talk to it
- Discover it
- Trust it
- Use it
- Detach it

The Distributed Application



ISO/IEC 24727-3: Application Interface

- Objectives
  - Client-application interface for all card-application services.
  - Client-application centric language and mechanisms
  - Multi-application interoperability
  - Long-term evolution of card-based systems
- Scope
  - Card-application services accessed through requests and responses at the client-application API
  - Programming language independent definition

ISO/IEC 24727-3: Design Themes

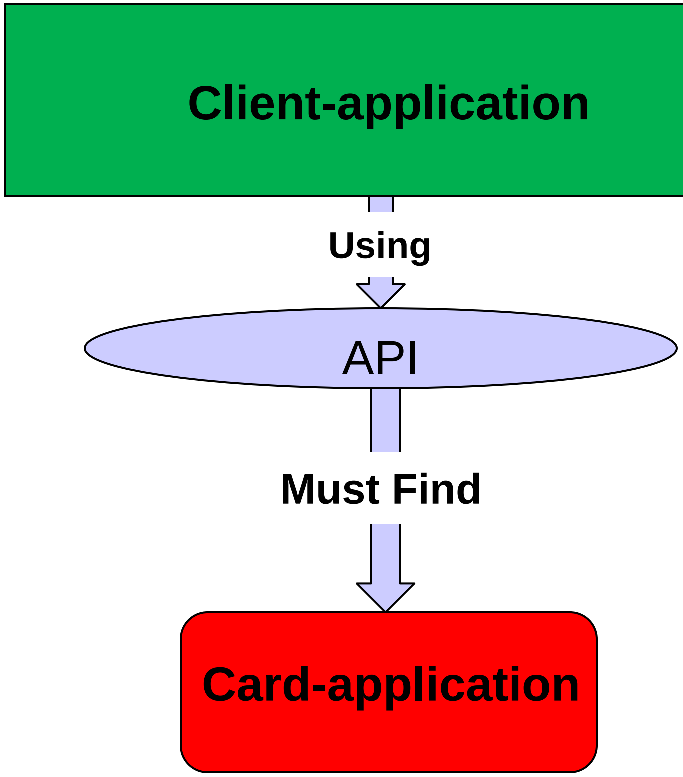
- API presents client-application semantics
- Service requests (e.g. Authenticate, Verify, Sign)
- Discovery based on semantics
- Complete object security model
- Consistent treatment of loadable request services

Resultant Characteristics of API
(ISO/IEC 24727-3: Application Interface)

- Client-application centric
- Formal definition (ASN.1)
- Provide access to token through full range of methods
- Establish a well-defined Model of Computation (MOC)
- Allow for token administration
- Provide MOC level discoverability mechanisms
- Extensible

Find It

The first order of business for a client-application is to locate the desired token.



Client-application

Using

API

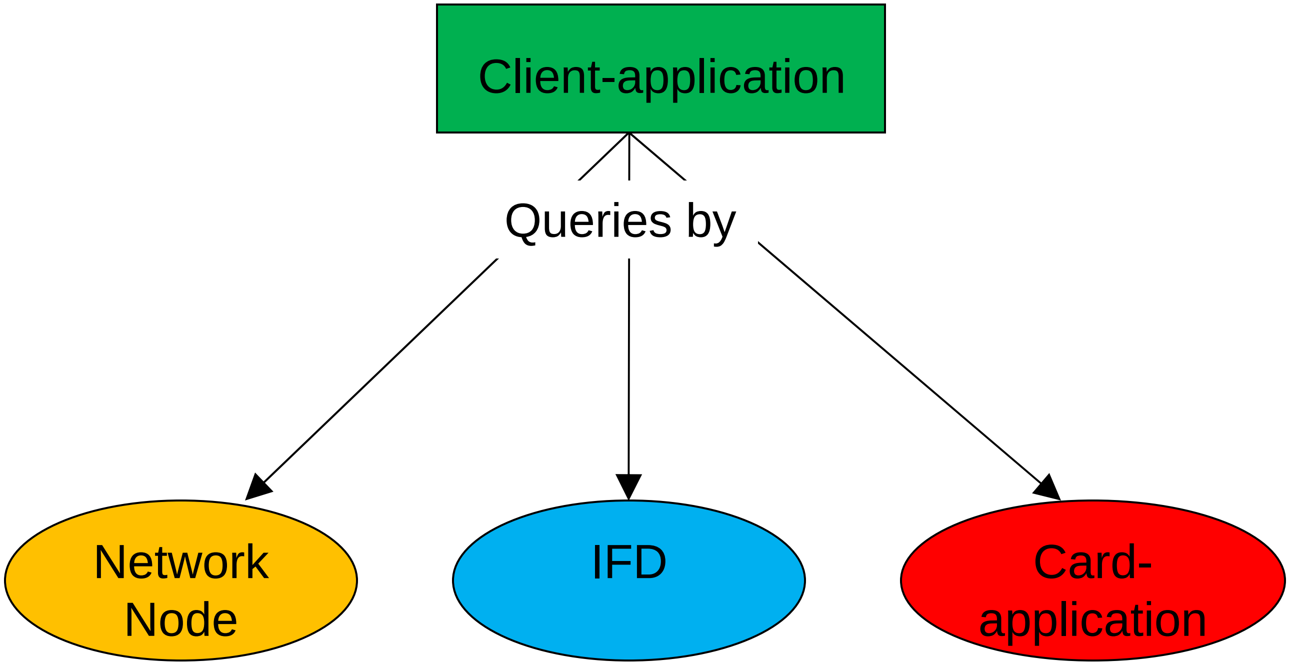
Must Find

Card-application

Path Services

- Initialize
  - This is an Administrator defined function of the API
  - This function binds the client-application to a specific stack configuration
  - The specific stack configuration defines the capability of the subsequent “Path” command to locate a token
- Card Application Path
  - Depending on the stack configuration, this function provides for responses to queries (from the client-application) to locate a specific token (card-application)

Path Function

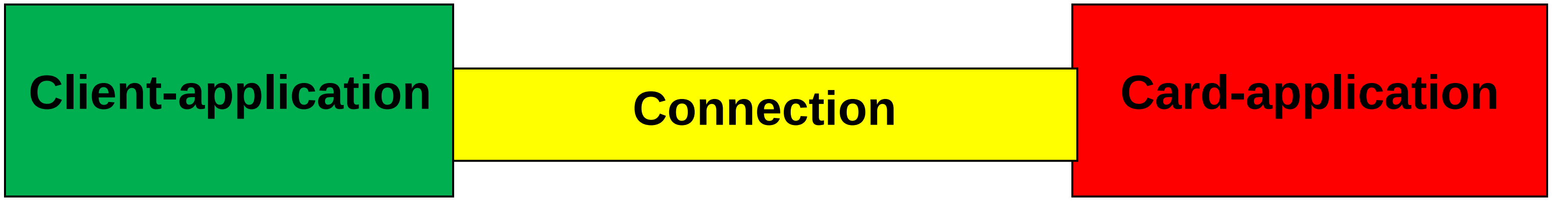


And the Path function responds with available paths to that specific point.

Talk to it

Once a client-application knows the path to a specific card-application

It must establish a communication channel (along the known path) between the client-application and the card-application.



Connection Services

- CardApplicationConnect
  - Using the stack configuration resulting from Initialize the client-application can issue a *CardApplicationConnect* to establish a communication channel to the specified card-application

    This is a completely unsecured channel.

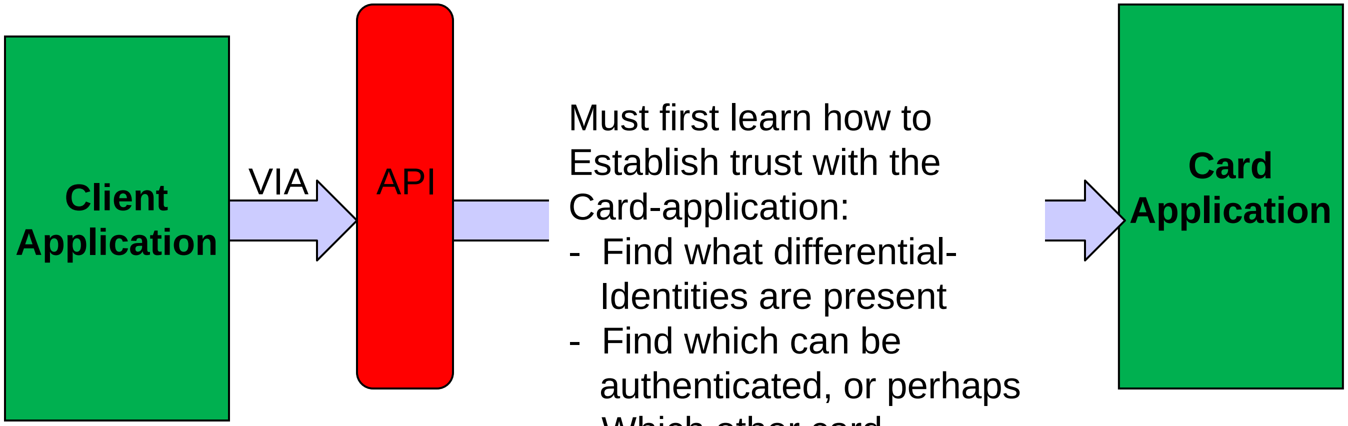
  - This channel (referred to as a “connection”) is referenced in subsequent request by an “opaque handle”

Connection Services

- CardApplicationStartSession

In order to guarantee privacy for all communication between the client-application and the card-application a secured channel, called a “session” can be established through the existing connection by using the existing “opaque handle” as an input parameter to a CardApplicationStartSession request

Discover it



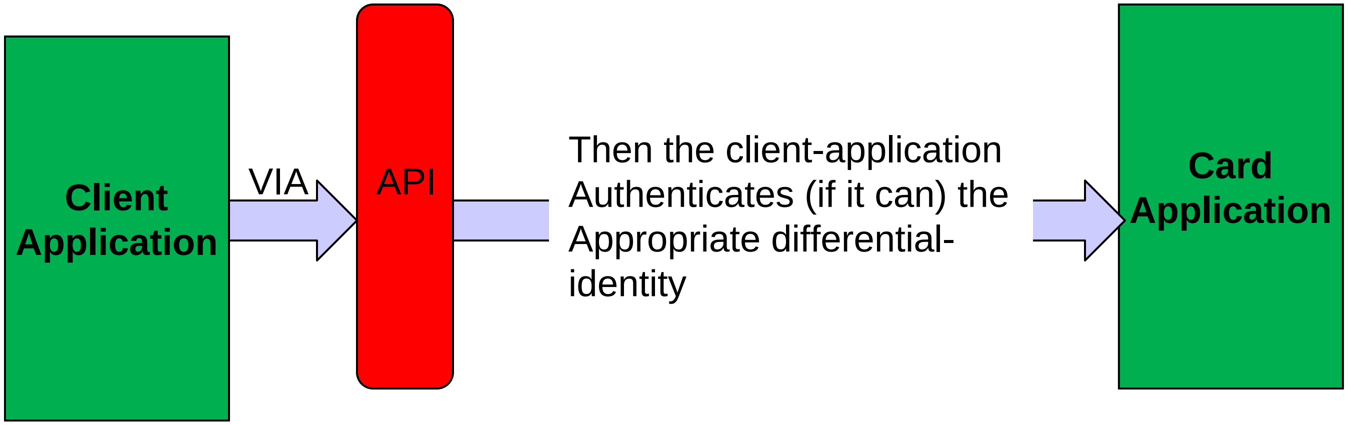
Must first learn how to Establish trust with the Card-application:

- Find what differential-Identities are present
- Find which can be authenticated, or perhaps
- Which other card-applications are available

Card Application Services

- CardApplicationList
- CardApplicationServiceLise
- DataSetList
- DSIList
- Differential-IdentityList
- ACLList
- ACLAccessRuleModify

Trust it



Client Application

VIA

API

Then the client-application Authenticates (if it can) the Appropriate differential-identity

Card Application

Differential-identity Services

- Authenticate
- Get
- Update

Cryptographic Services

Cryptographic services from the token card-application are used to establish trust within the client-application using cryptographic mechanisms such as digital signatures and digital certificates

- GetChallenge
- Sign
- Encipher
- Decipher
- VerifySignature

Authentication Protocol Standardization

- ISO/IEC 24727-3 Defines a generic method to describe an AP
- ISO/IEC 24727-3 Annex A documents and makes available twenty two (22) common APs (license free)
- ISO/IEC 24727-3 Annex A provides unique OIDs for the 22 APs

Authentication Protocols (APs)

- Existing ISO standards are very general re APs (ISO/IEC 9798, and some in 7816 series)
- Existing industry standards are very explicit re APs (EMV, GlobalPlatform etc)
- Up until the publication of ISO/IEC 24727-3 there was no generic methodology for describing a smartcard (or any other) AP
- MOST interoperability problems related to smartcards are due to subtle discrepancies between APs
- Most people think that APs and cryptographic algorithms/ciphers are the same thing – they are not

Authentication Protocol Details

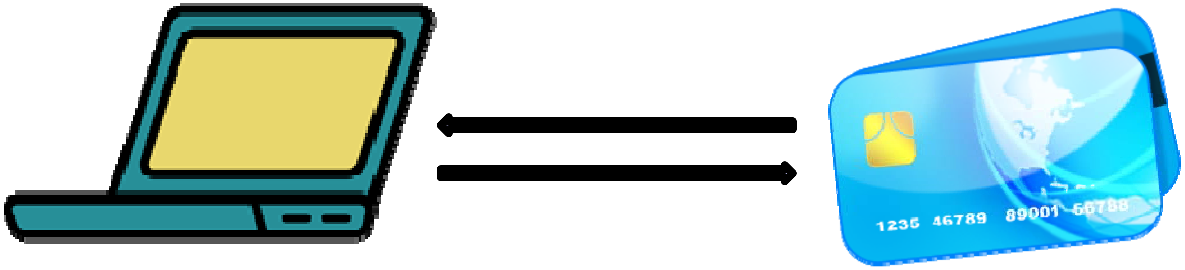
- Short name (part of ASN.1 OID)
- Short description
- General description
- Purpose
- Marker – empty or not, ASN.1 representation
- Authentication steps – step by step description with ASN.1 representation
- State Machine - rules for setting true/false

DID Commands action for Registration

- DIDCreate – didStructure parameter + ASN.1 representation
- DIDUpdate – markerList parameter + ASN.1 representation, generateFlag, publicKey/privateKey options
- DIDGet - didStructure parameter + ASN.1 representation

Getting to know the other party (using authentication protocols)

Internal Authenticate
External Authenticate



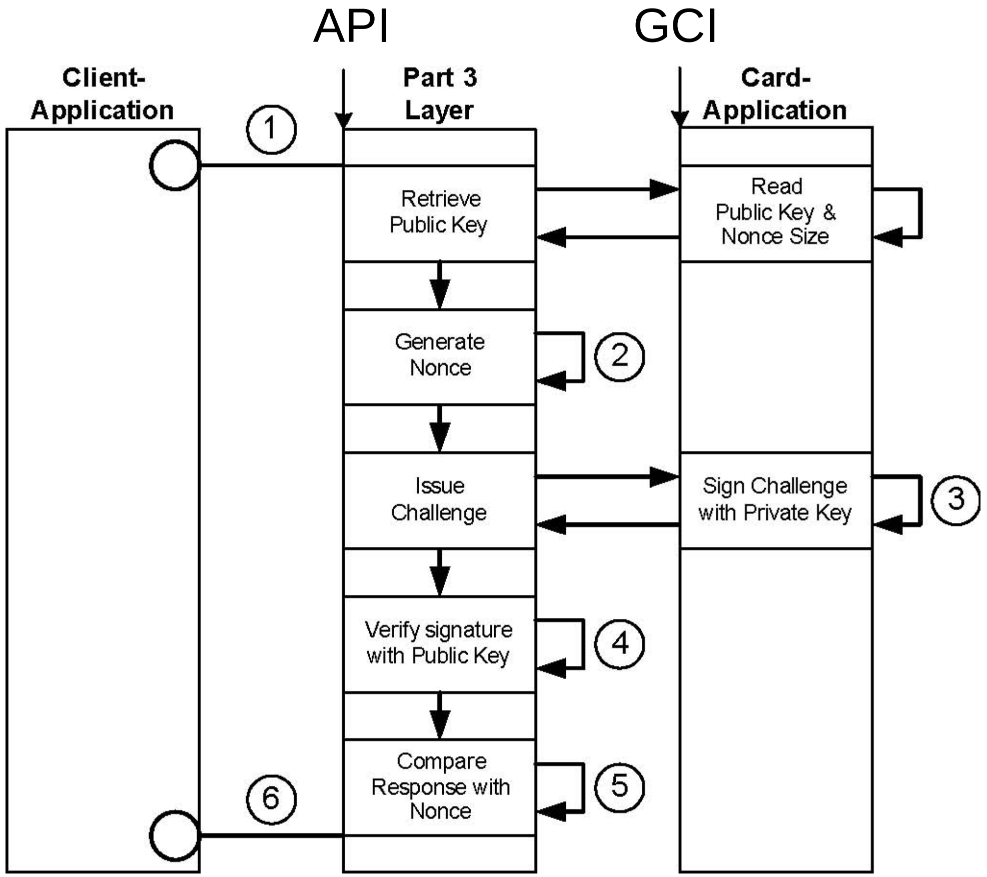
Computer

Token

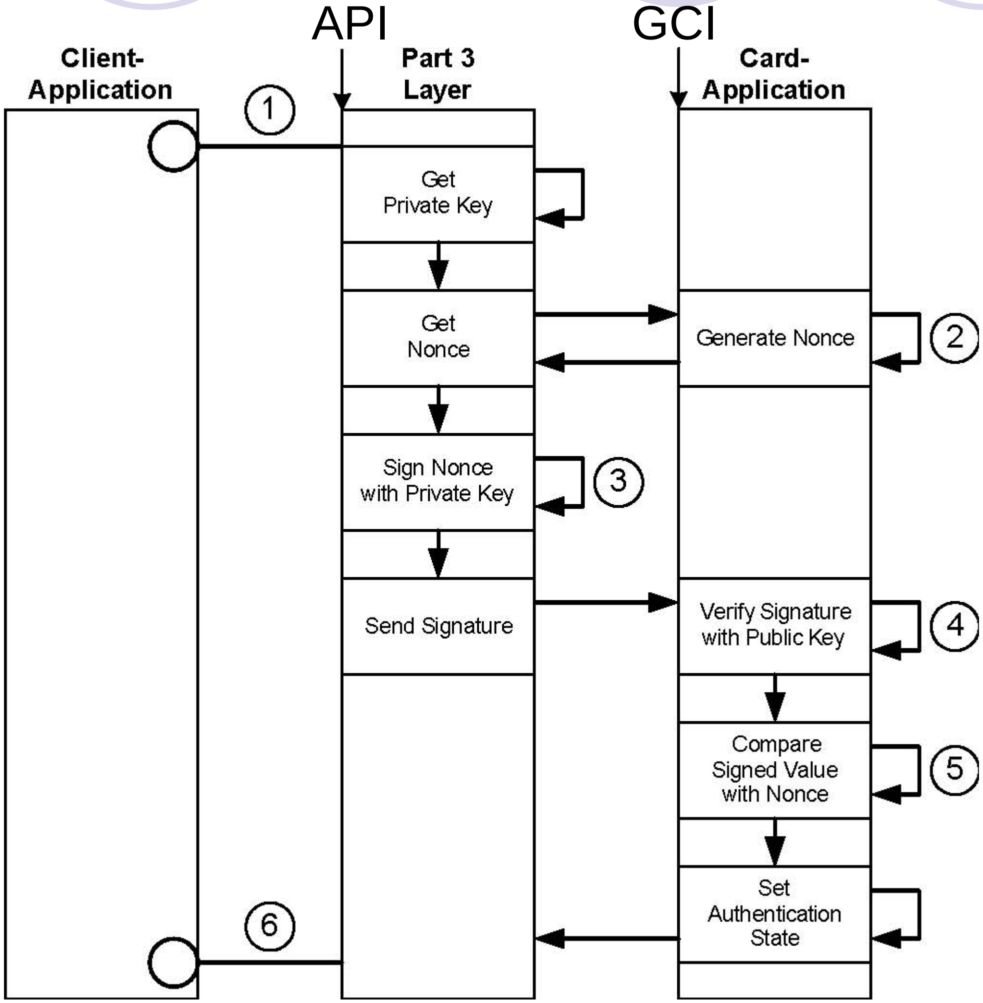
Computer authenticates the Token
Token authenticates the Computer

F

Asymmetric Internal Authenticate


API
GCI
Client-Application
Part 3 Layer
Card-Application
1
Retrieve Public Key
Read Public Key & Nonce Size
Generate Nonce
2
Issue Challenge
Sign Challenge with Private Key
3
Verify signature with Public Key
4
Compare Response with Nonce
5
6

Asymmetric External Authenticate



ASN.1 Representation of a Marker

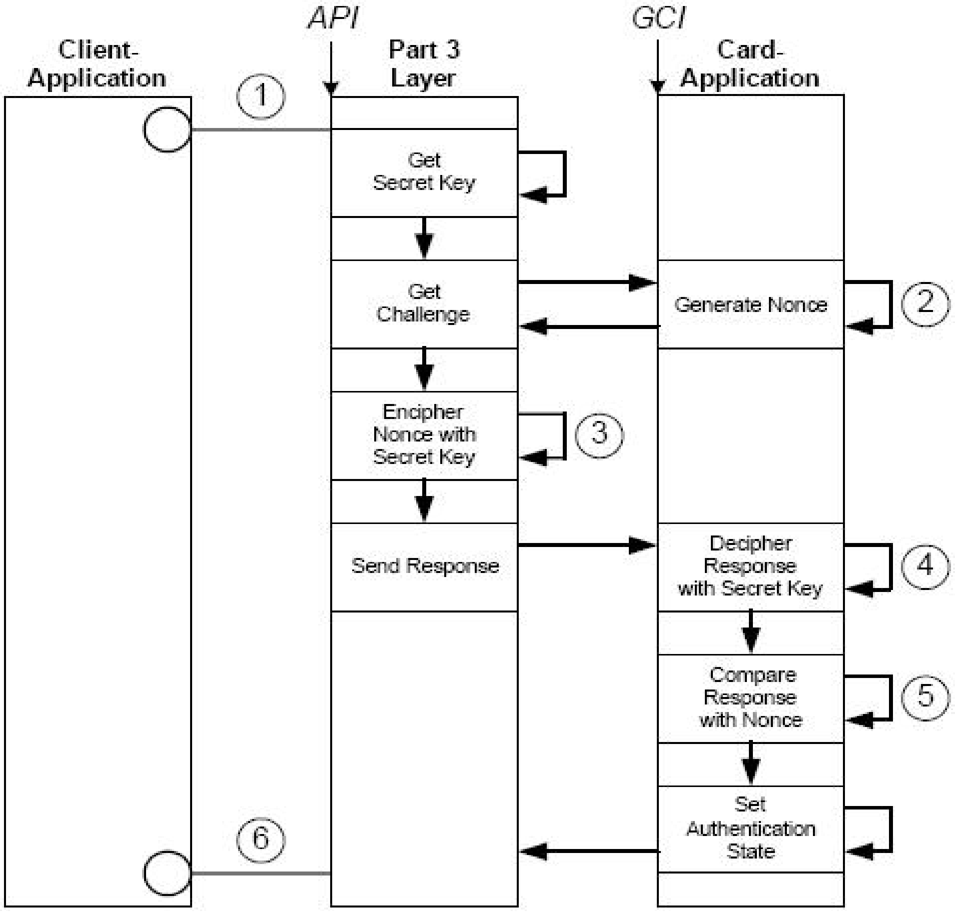


Figure A.5 — Symmetric External Authenticate

```
MarkerAP007 ::= SEQUENCE {

 encryptionAlgorithm
                  AlgorithmIDParameters,
 hashAlgorithm
                  AlgorithmIDParameters,
 keySize               INTEGER,
 secretKey             OCTET STRING,
 nonceSize             INTEGER

}
```

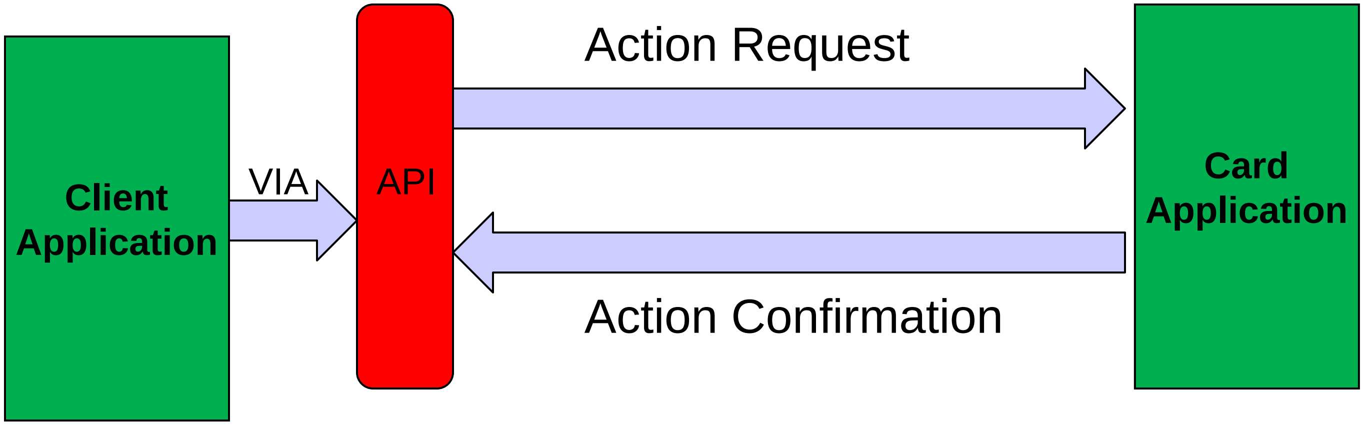
Object Identifiers (OIDs)

- Object identifiers are used in the specification of authentication protocols to distinguish cryptographic algorithms
- Each OID specifies the defining organization, and hence the detailed specification, of that cryptographic algorithm

  For example:

  id-aes256-ECB ::= { *joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) nistAlgorithm(4) 1 41* }

Use it


Client Application
VIA
API
Action Request
Card Application
Action Confirmation

Card-application Services

- CardApplicationList
- CardApplicationServiceList
- Describe
- ExecuteAction

Named Data Services

- DataSetList
- Select
- DSIList
- Read
- Write

Administration Operations

- CardApplicationCreate
- CardApplicationDelete
- CardApplicationLoad
- CardApplicationServiceList
- CardApplicationServiceDescribe
- CardApplicationServiceCreate
- CardApplicationServiceDelete
- CardApplicationServiceLoad
- Differential-IdentityList
- Differential-IdentityCreate
- Differential-IdentityGet
- Differential-IdentityUpdate
- Differential-IdentityDelete
- ACLList
- ACLAccessRuleModify
- DataSetList
- DataSetCreate
- DataSetSelect
- DataSetDelete
- DSIList
- DSICreate
- DSIRead
- DSIWrite
- DSIDelete

Creating the Registry

- The ISO/IEC 24727-3 layer is responsible for writing the Registry via the GCI.
- It creates the Registry as either a file or as an ISO/IEC 7816-4 data object.
- The structure of the Registry is defined in ISO/IEC 7816-15
- While the Registry must be created by an ISO/IEC 24727-3 layer, it may then physically reside “off-token”

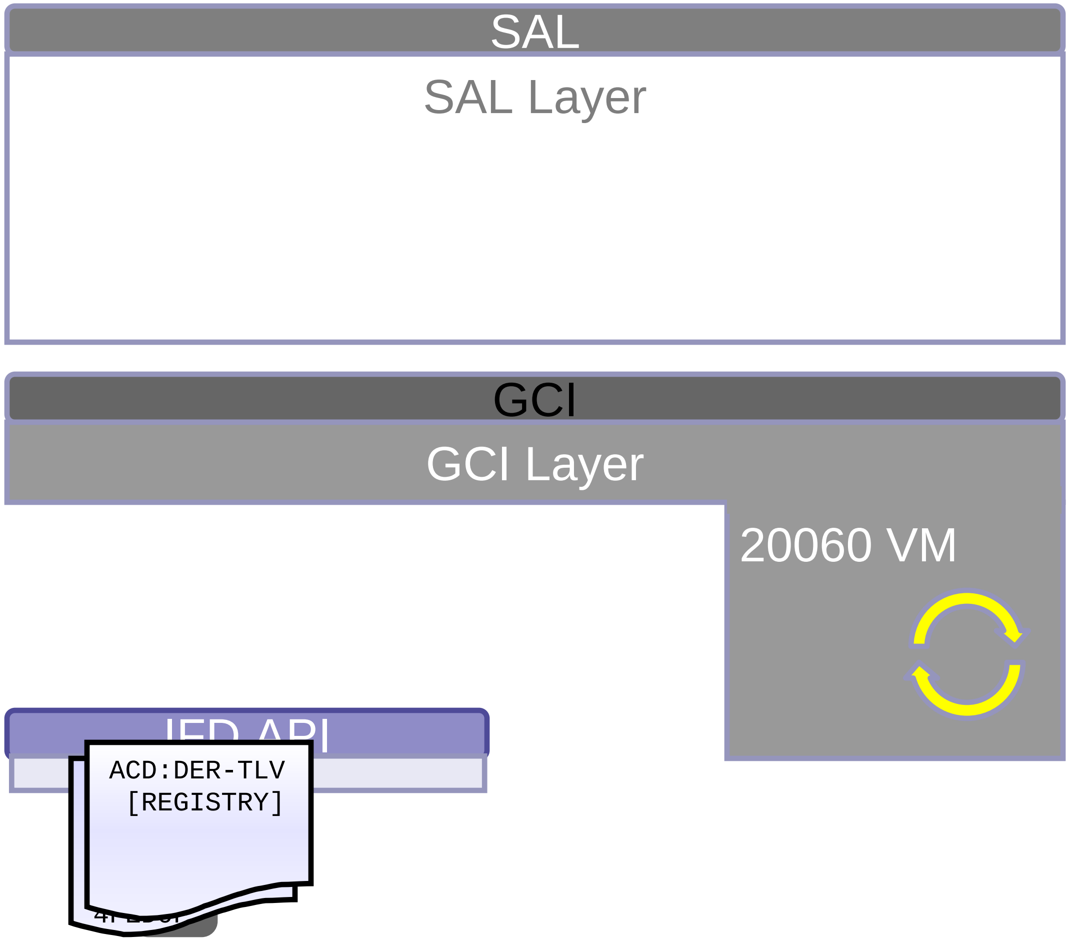
Creating the CCD

- The ISO/IEC 24727-3 layer is responsible for writing the CCD via the GCI.
- It creates the CCD as either a file or as an ISO/IEC 7816-4 data object.
- The content of the CCD is defined in ISO/IEC 24727-2
- The CCD may physically reside “on-token” or “off-token”

Creating the ACD

- The ISO/IEC 24727-3 layer is responsible for writing the ACD via the GCI.
- It creates the ACD as either a file or as an ISO/IEC 7816-4 data object.
- The content of the ACD is defined in ISO/IEC 24727-2
- The ACD may physically reside “on-token” or “off-token”

How it works: an illustration



Secure Messaging

- APDU based means of achieving data integrity and some level of privacy between the IFD and the token
- ISO/IEC 7816-4 defines the basic mechanics of secure messaging, however not to a level of interoperable specificity
- ISO/IEC 24727 addresses interoperable secure messaging in ISO/IEC 24727-4

Detach it

- In the most general case, there are three levels of connectivity between a client-application and a token:
  - a session
  - a connection
  - a stack configuration

  A client-application can discontinue each in the inverse order of their establishment

Connection Services

- CardApplicationEndSession
  - Discontinue the secured “session” but leave the connection in place

- CardApplicationDisconnect
  - Discontinue the connection from the client-application to the token but leave the stack configuration in place

- Terminate
  - Discontinue the stack configuration

A Note About Differential-Identity

- Differential-identity forms a high-level (i.e. client-application) mechanism that maps the "social" world (names, etc.) to the technical world of the token with its short file names, tags, key references, esoteric authentication protocols.

- Thus, differential-identity is a central feature in the provision of interoperability among diverse identification systems.

QUESTIONS?