Java Card™ Technology for Emerging WLAN Environments

Pascal Urien

Professor
ENST
http://www.enst.fr

Guy Pujolle

Professor
LIP6
http://www.lip6.fr

TS-0285

Goal

Learn how to design two-factor authentication tokens, built with Java Card technology and based on the open code project O*penEapSmartcard*

These token are demonstrated in a Wi-Fi platform, both for client's terminal and RADIUS server

Agenda

Introduction

The Open Source Project, *OpenEapSmartcard*

OpenEapSmartcard for Wi-Fi Infrastructures

DEMO: OpenEapSc for Wi-Fi Platform at Work!

Agenda

Introduction

The Open Source Project, *OpenEapSmartcard*

OpenEapSmartcard for Wi-Fi Infrastructures

DEMO: OpenEapSc for Wi-Fi Platform at Work!

Why Two-Factor Authentication Is Needed

Password issues

- According to a work done in 2005 by Doug Tygar, professor of computer science at U.C. Berkeley, attackers can sniff out what's typed on keyboards, simply by recording keystroke sounds
  - He recommended to enhance security with two-factor authentication that combine passwords with one-time password tokens or smart cards, or with biometric recognition, like fingerprint readers
- A well known two-factor authentication device is the RSA SecurID token
  - This token works with a proprietary authentication infrastructure called ACE

Two-Factor Authenticati

Our proposal

- In this session we introduce an open authentication infrastructure dedicated to Wireless LAN environments
- Tokens are based on the Java Card technology
- They execute Java applications supported by the open code project OpenEapSmartcard
- The authentication platform is fully based on IETF standards (mainly the Extensible Authentication Protocol, EAP), no proprietary features
- Our demonstrated authentication scenario deals with the classical SSL/TLS protocol (more precisely EAP-TLS), which is widely deployed through the WEB, and which relies on Public Key Infrastructure (PKI)
  - Each client holds an X509 certificate and a private key
  - Each authentication server holds an X509 certificate and a private key

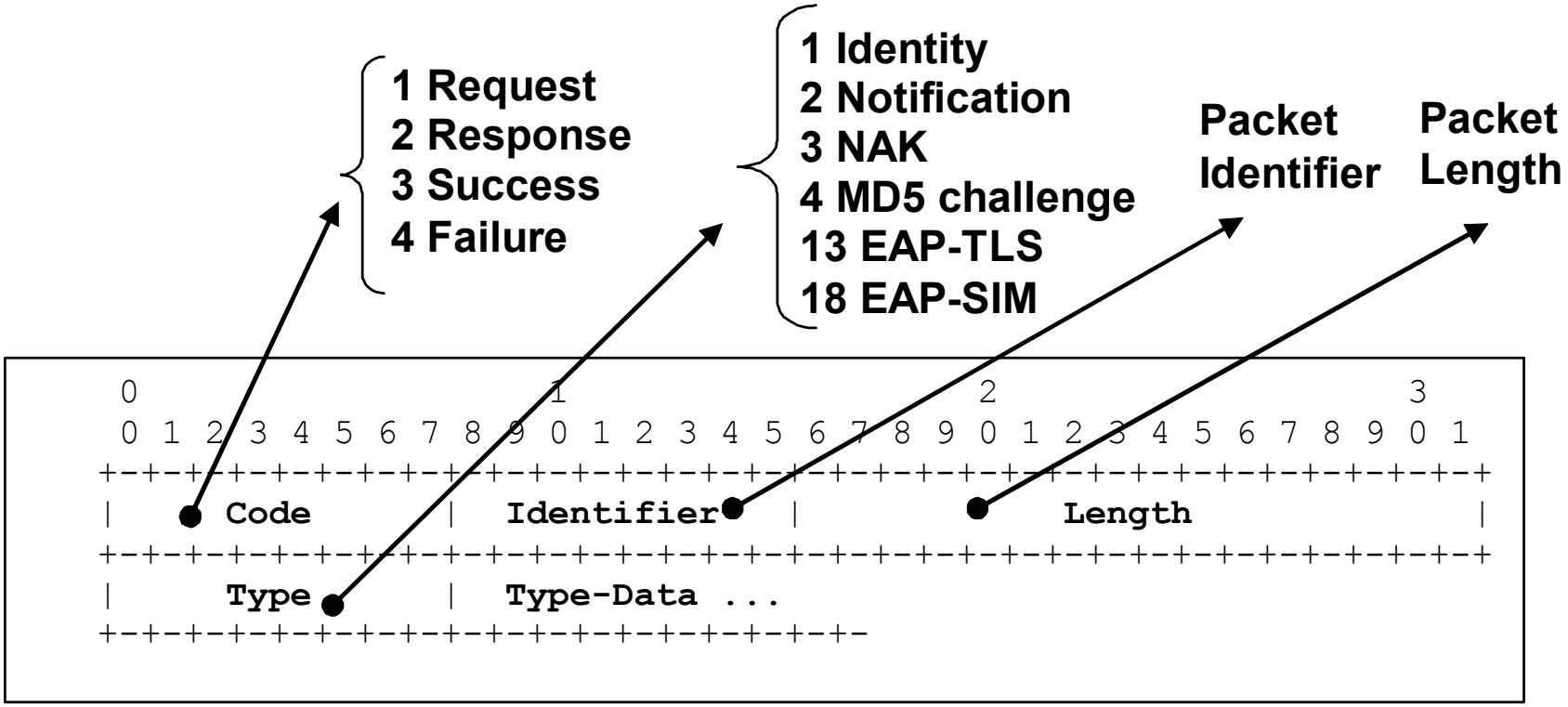
What Is EAP ?

EAP is an IETF standard

- The Extensible Authentication Protocol (EAP) was introduced in 1999, in order to define a flexible authentication framework
  - EAP, RFC 3748, “Extensible Authentication Protocol, (EAP)”
  - EAP-TLS, RFC 2716, “PPP EAP TLS Authentication Protocol”
  - EAP-SIM, RFC 4186, “Extensible Authentication Protocol Method for Global System for Mobile Communications (GSM) Subscriber Identity Modules (EAP-SIM)”
  - EAP-AKA, RFC 4187, “Extensible Authentication Protocol Method for 3rd Generation Authentication and Key Agreement (EAP-AKA)”

What Is EAP ?

EAP message format



What Is EAP ?

An *Esperanto* for access control in IP infrastructures

- Wireless LAN
  - Wi-Fi, IEEE 802.1x
  - WiMAX mobile, IEEE 802.16e , PKM-EAP
- Wired LANs
  - ETHERNET, IEEE 802.3
  - PPP, RFC 1661, “The Point-to-Point Protocol (PPP)”
- VPN (Virtual Private Network) technologies
  - PPTP, RFC 2637, “Point-to-Point Tunnelling Protocol”
  - L2TP, RFC 2661, “Layer Two Tunnelling Protocol”
  - IKEv2, RFC 4306, “Internet Key Exchange Protocol”
- Authentication Server
  - RADIUS, RFC 3559, “RADIUS (Remote Authentication Dial In User Service) Support for Extensible Authentication Protocol (EAP)”
  - DIAMETER, RFC 4072, “Diameter Extensible Authentication Protocol Application”
- Voice over IP
  - UMA, Unlicensed Mobile Access, http://www.umatechnology.org

What Is EAP ?

EAP components

- According to RFC 3748, EAP implementations conceptually consist of the four following components:
  1. The lower layer is responsible for transmitting and receiving EAP frames between the peer and authenticator
  2. The EAP layer receives and transmits EAP packets via the lower layer, implements duplicate detection and retransmission, and delivers and receives EAP messages to and from EAP methods
  3. EAP peer and authenticator layers; based on the Code field, the EAP layer de-multiplexes incoming EAP packets to the EAP peer and authenticator layers
  4. EAP methods implement the authentication algorithms, and receive and transmit EAP messages; EAP methods can be implemented in Java Card systems

What Is EAP ?

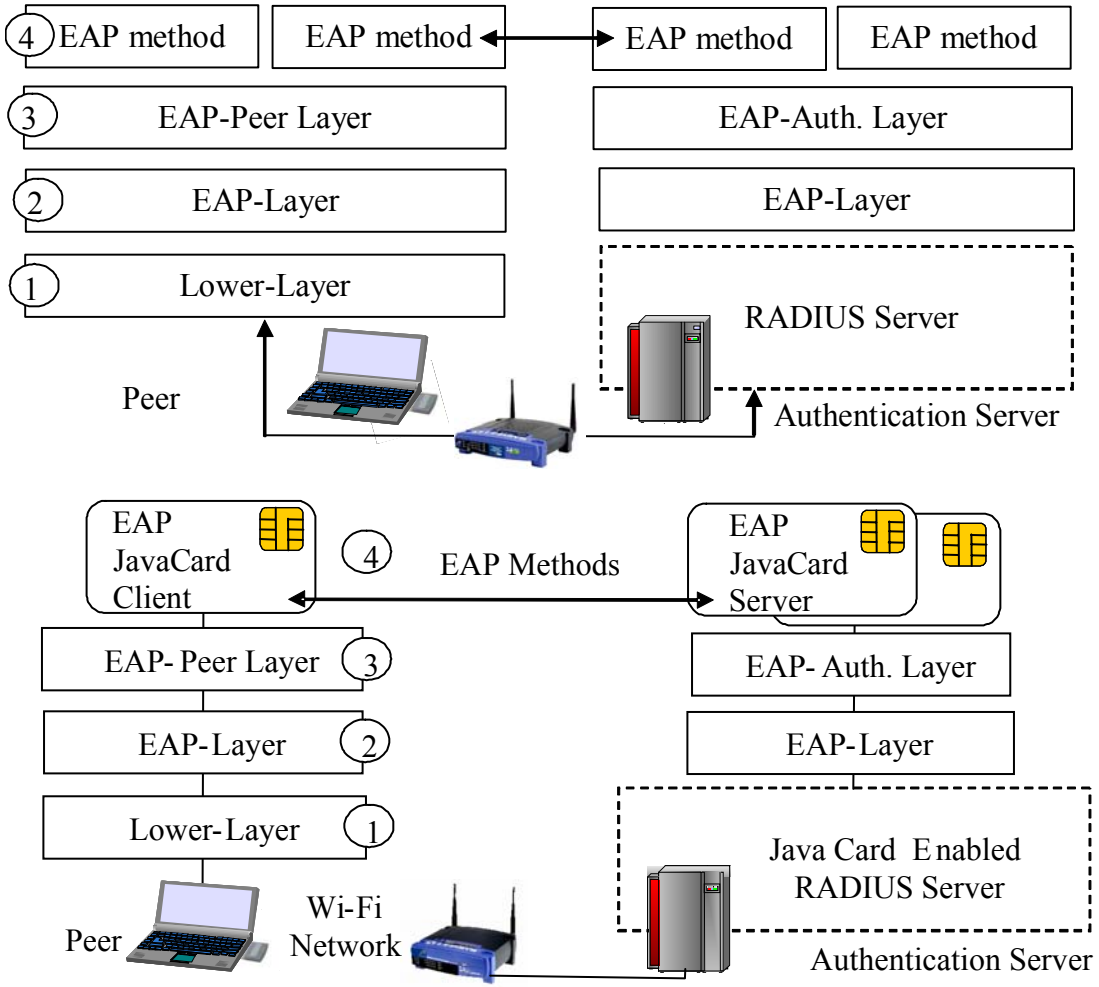
EAP Java Card Technology

Full Software Implementations

Partial Software Implementations

+

EAP JavaCard Technology



Agenda

Introduction

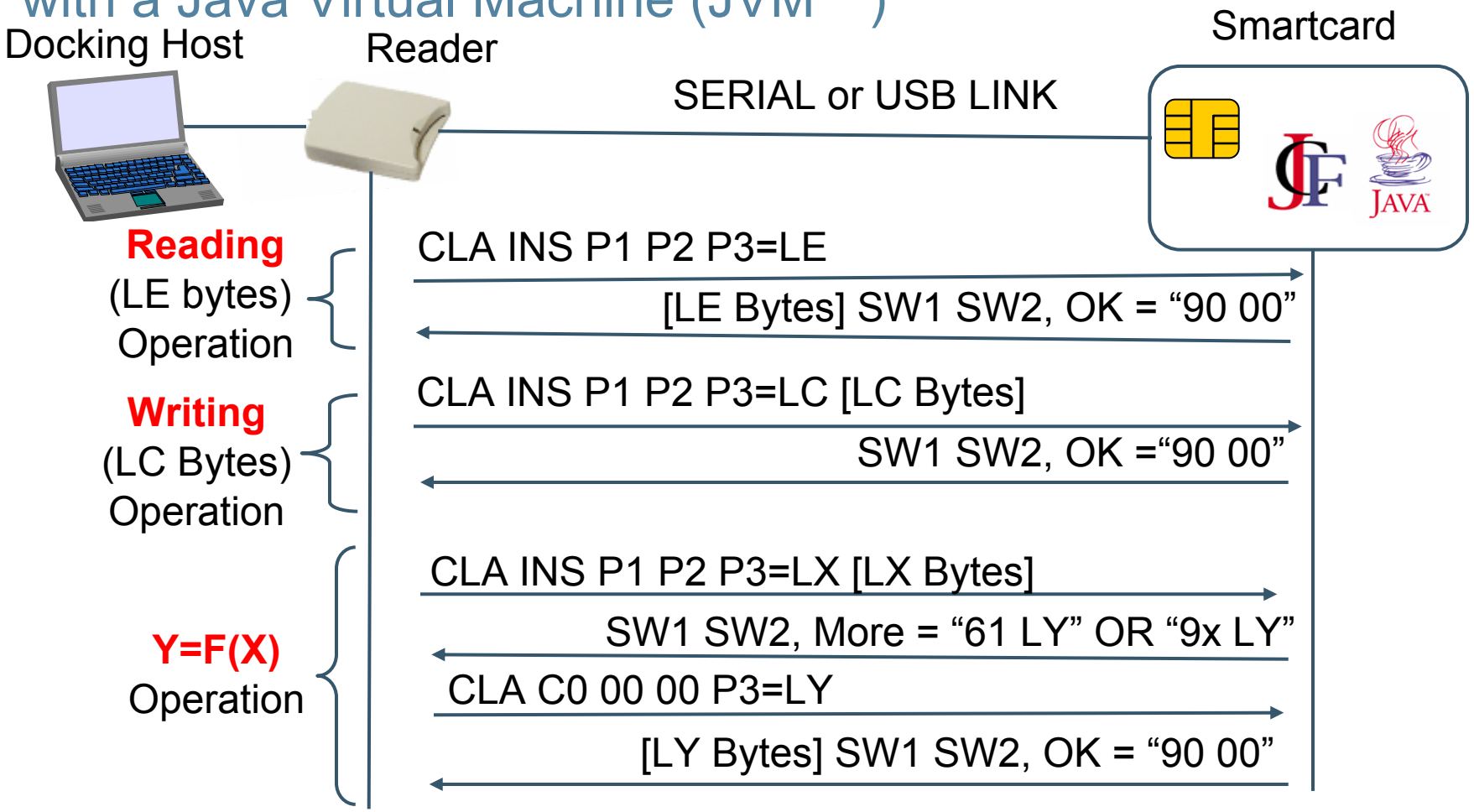
The Open Source Project, *OpenEapSmartcard*

OpenEapSmartcard for Wi-Fi Infrastructures

DEMO: OpenEap for Wi-Fi Platform at Work!

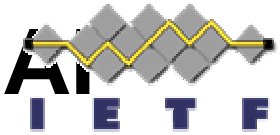
What Is Java Card Technology?

A Java Card is a tamper resistant computer (smartcard) with a Java Virtual Machine (JVM™)



The terms “Java Virtual Machine” and “JVM” mean a Virtual Machine for the Java™ platform.

Designing Smartcards for EAP



Internet (IETF) draft—*EAP support in smartcard*

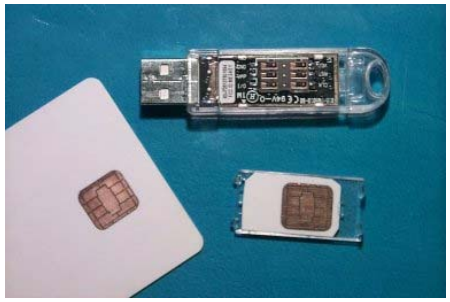
- The EAP smartcard is described in an internet draft, whose 13th version was issued in February 2007; it processes EAP requests or notifications and returns response; its logical interface is a set of four services
  - The IDENTITY service: A smartcard may manage several network accounts
  - The NETWORK service: EAP messages are processed by the smartcard; at the end of the authentication method, a session key is computed
  - The SECURITY service: This service manages PIN codes (Personal Identification Number) that are needed for security purposes
  - The PERSONALIZATION service: This service updates information stored in the smartcard

The Open Platform,

OpenEAP Smartcard

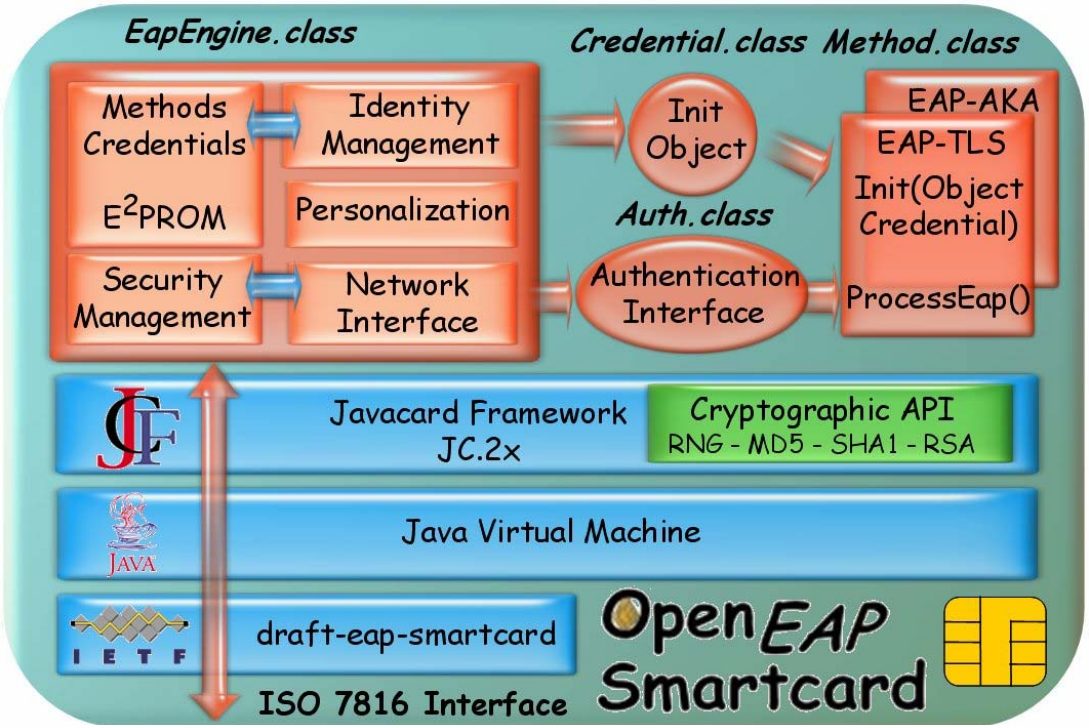
Why open Java Card technology code ?

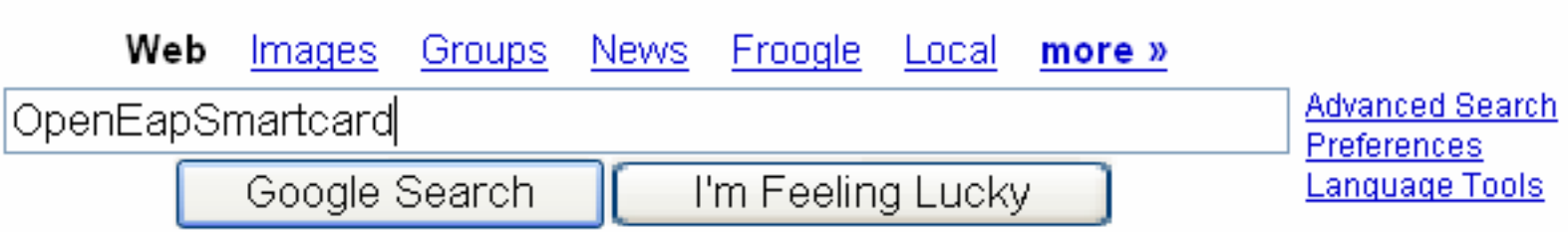
- Internet and WEB technologies are based on open code
- No proprietary features
- Good security principle that enables code reviewing
- Fair choice among multiple Java Card systems



OpenEapSmartcard

Architecture overview





The Open Platform,

Four Java components

- The **EapEngine** that implements the EAP core, and acts as a router that sends and receives packets to/from authentication methods
  - It offers four services, Network interface, Identity management, Security management, Personalization management
- A **Credential Object** that stores information needed for method initialization
- One or more **Methods** that instantiate authentication scenari like EAP-TLS or EAP-AKA
- An **Authentication Interface** that defines all services offered by EAP methods
  - The two main functions are ***Init(CredentialObject)*** and ***Process-Eap()***

The Open Platform, *OpenEapSmartcard*

Details of the Authentication Interface

Interface auth	
void	**fct**(javacard.framework.APDU apdu, byte[] in, short inlength) Method functions apdu: incoming APDU in: buffer associated to the incoming APDU inlength: P3 value
byte[]	**Get Fct Buffer**() Returns a function buffer
short	**Get Fct Length**() Returns a function buffer length
short	**Get Fct Offset**() Returns a function buffer offset
byte[]	**Get Out Buffer**() Returns the response buffer
short	**Get Out Length**() Returns the response buffer length
short	**Get Out Offset**() Returns the response buffer offset
auth	**Init**(java.lang.Object credentials) Method Initialization
boolean	**IsFragmented**() Fragmentation in progress
boolean	**IsLongFct**() Indicates that the response of a function is stored in a private buffer
boolean	**IsLongResponse**() Indicates that the response of the method is stored in a private buffer
short	**process eap**(byte[] in, short inlength) Method Processing in: incoming APDU buffer inlength: length of the incoming APDU Returns -length of the response -negative value if an error occurred
void	**reset**() Resets the method
short	**status**() Gets the method status

Agenda

Introduction

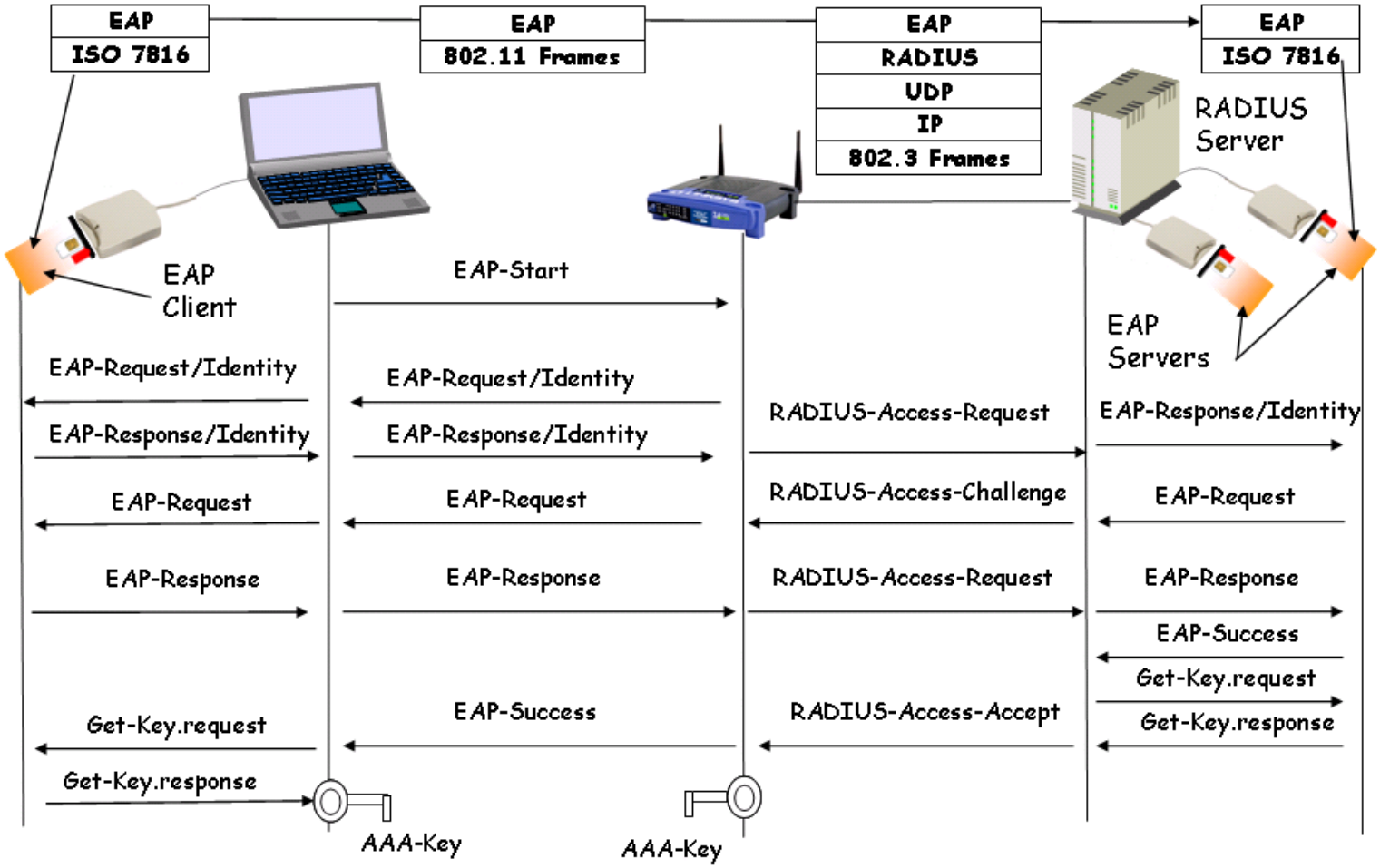
The Open Source Project, *OpenEapSmartcard*

OpenEapSmartcard for Wi-Fi Infrastructures

DEMO: OpenEapSc for Wi-Fi Platform at Work !

Authentication Platform

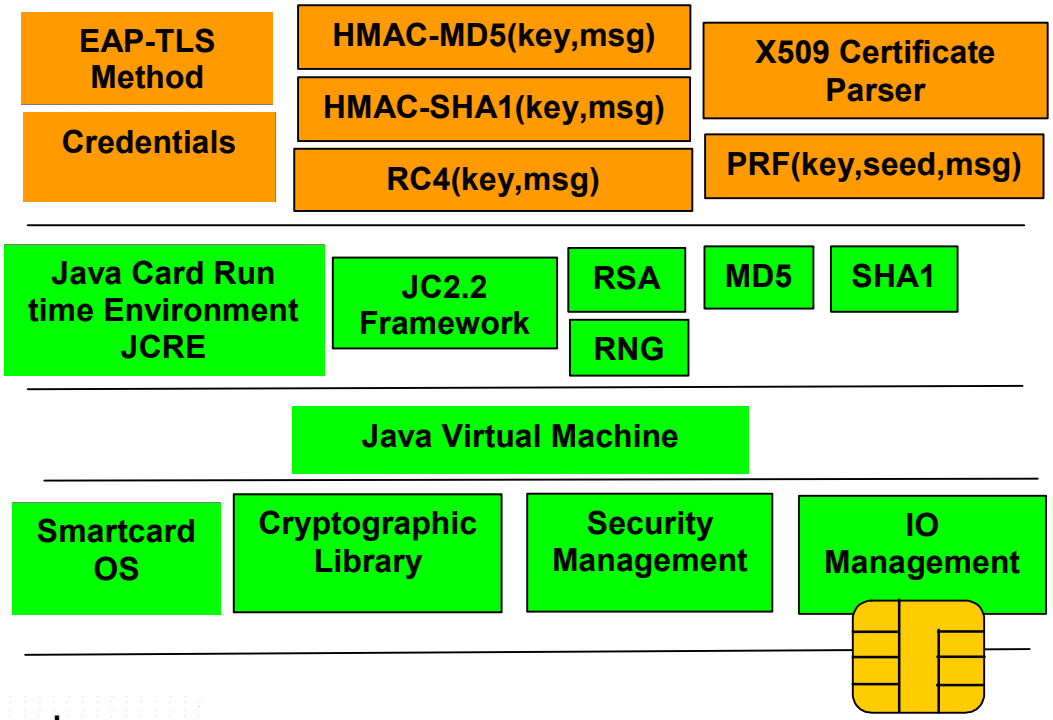
Overview



OpenEapSmartcard Details

EAP-TLS Java Card technology design

OpenEapSmartcard



Java Card 2.x packages

OpenEapSmartcard Details

About EAP-TLS Java Card technology performances

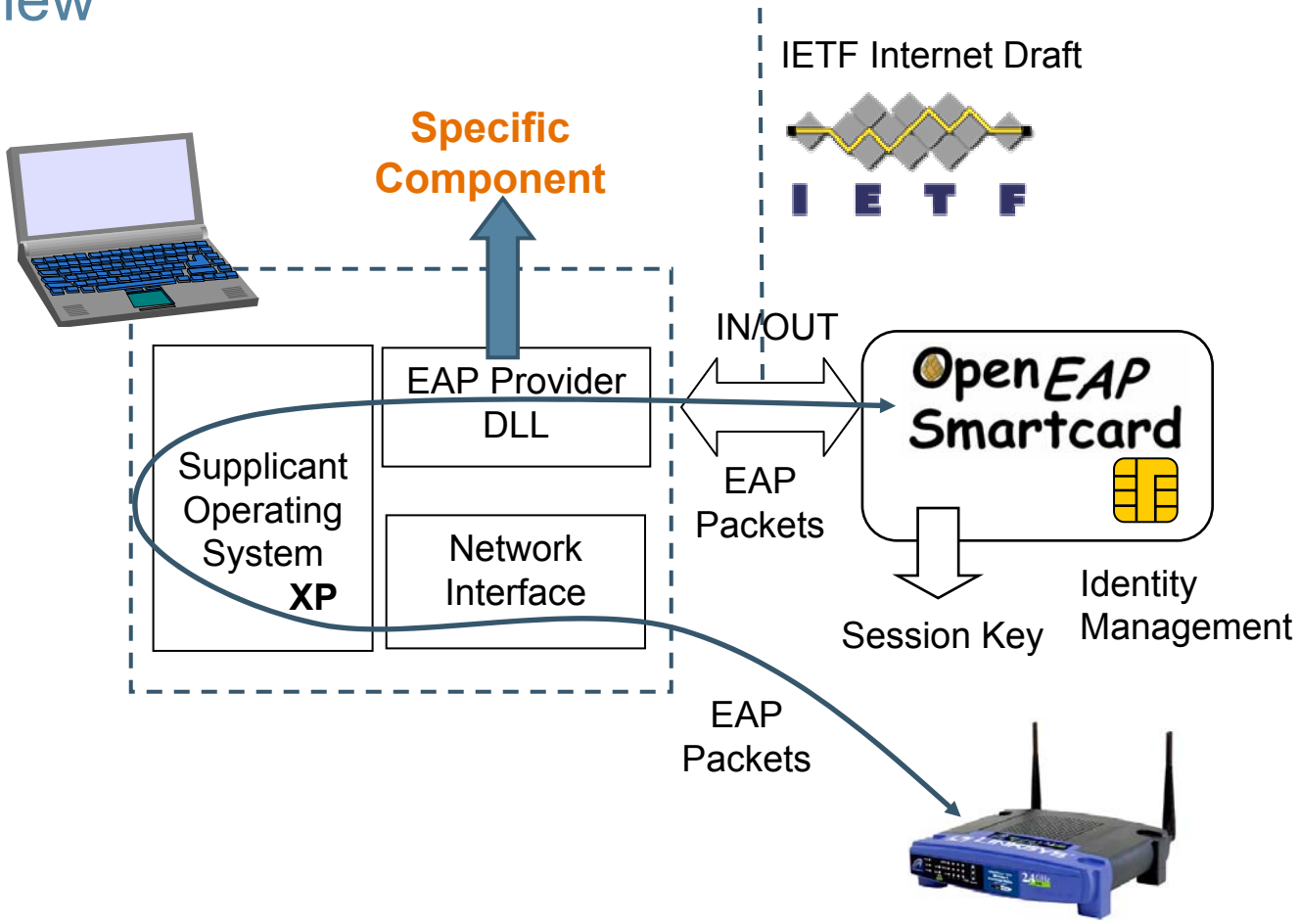
$$T_{EAP\_TLS} = T_{DataTransfer} + T_{CryptoComputing} + T_{SoftwareOverhead}$$

EAP-TLS Computing time	A (eGate)	B (JCOP)	C (Gemplus)	D (Gemplus)	Best Of (Axalto)
$T_{DataTransfer}$ (ms)	2492	5326	5219	1433	400
$T_{CryptoComputing}$ (ms)	13221	6507	7648	2117	1850
$T_{SoftwareOverhead}$ (ms)	62618	21914	14784	6827	3050
Total = $T_{EAP-TLS}$ (ms)	78331	33747	27651	10377	4900

5 seconds

OpenEapSmartcard Integration in XP

Overview



OpenEapSmartcard Integration in XP

How does it work?

Request
Notification

RasEapMakeMessage

RasEapGetInfo

EAP Provider
Initialization

RasEapBegin

RasEapEnd

RasEapInvokeConfigUI

RasEapGetIdentity

Get Smartcard Identity
Via EAP-Identity-Request
Message

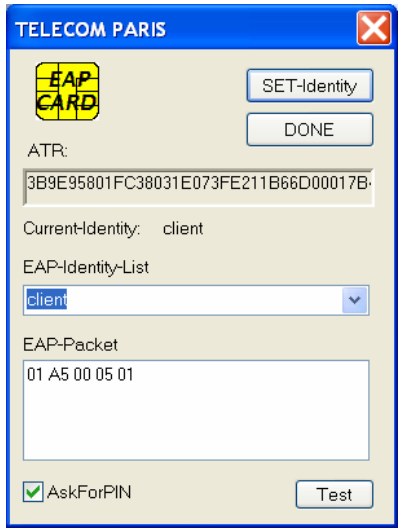
RasEapInvokeInteractiveUI

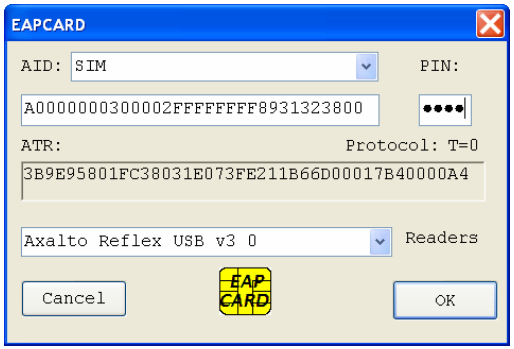
Smartcard EAP Application Selection

RAS

1 2 3 4 5

Get-Next-Identity
Set-Identity





EAP-TLS Java Card Technology Integration in RADIUS Servers

Scalability versus *Erlang B Law*

$$p_c = \frac{(\lambda / \mu)^c}{c!} \left[\sum_{k=0}^{c} \frac{(\lambda / \mu)^k}{k!} \right]^{-1}$$

- A RADIUS server manages several EAP-TLS Java Card systems; each device holds its own certificate and fully processes an EAP-TLS session
  - Pc is the probability of blocking (e.g., a RADIUS packet is silently discarded),
  - c is the number of EAP servers,
  - λ is the rate of authentication sessions
  - 1/μ the mean time of an authentication session (10s = 5s + 5s)
- Let's assume a network with 1000 users, authenticated every 10mn, then λ = 6x1000/3600=1,7 and so λ/μ = 60,000/3600 = 16,7
- The probability of blocking (pc) is about 50% with 9 smartcards (c = 9) and only 1% with 21 smartcards (c = 21)

Agenda

Introduction

The Open Source Project, *OpenEapSmartcard*

OpenEapSmartcard for Wi-Fi Infrastructures

DEMO: OpenEapSc for Wi-Fi Platform at Work!

DEMO

OpenEapSmartcard Platform for Wi-Fi Platform at Work!

DEMO

Summary

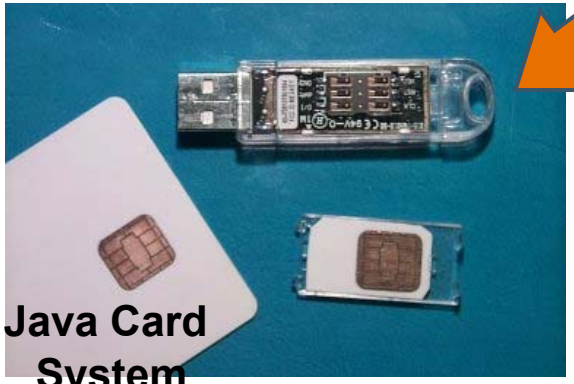
- This demonstration works with Java Card systems that run the *OpenEapSmartcard* package
- X509 certificates are issued with the well known OpenSSL software
- The client's Java Card system is personalized with credentials that include:
  - The user's certificate
  - The private key associated to the user's certificate
  - The CA certificate
  - The EAP-ID parameter
  - The user's identity, e.g., a friendly name linked to previous parameters
- RADIUS' Java Card systems are personalized with credentials that include:
  - The server's certificate
  - The private key associated to the server's certificate
  - The CA certificate
  - The user's identity, e.g., a friendly name linked to the previous parameters
- These Java Card systems are then deployed in a real Wi-Fi network

DEMO

Token production and personalization

Production
Applet Downloading

OpenEAP Smartcard



Java Card System

OpenSSL

Personalization Script

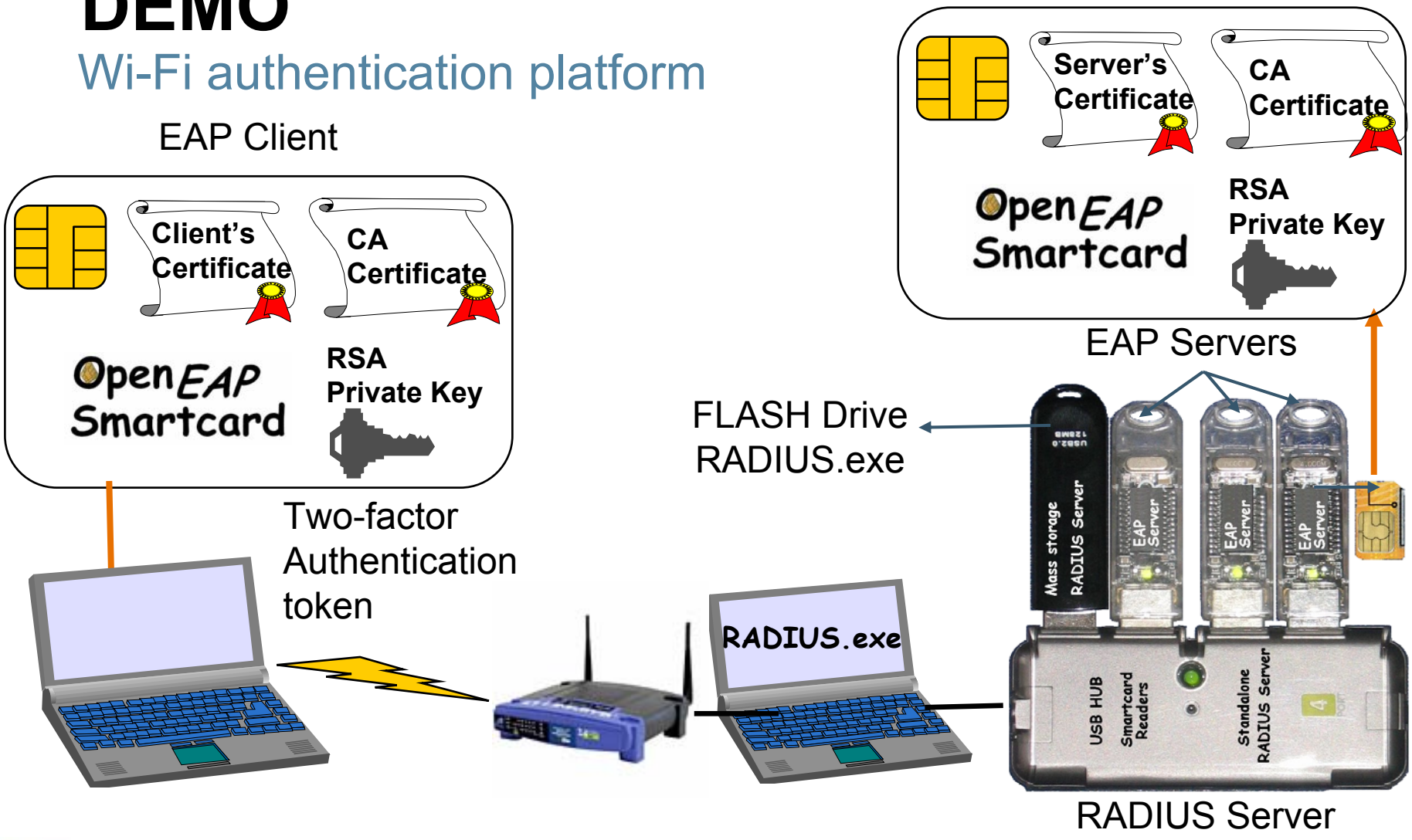
```
\MakeId>eaptools.exe enst pascal.urien@enst.fr id.bin 1024 ca.der 1024

identity:.................enst
eapid:...................pascal.urien@enst.fr
output file:.............id.bin
CA key size(bits):.......1024
CA_certificate:..........ca.der
Client key size(bits):...1024
Client certificate:......cert.der
Client private key file:.key.der

D:\Wi-Fi\javacard\OPENEAP\tlstools\MakeId>bin2script id.bin id.txt 0
Infile:...id.bin
Outfile:..id.txt
Adr:......0
D:\Wi-Fi\javacard\OPENEAP\tlstools\MakeId>
```

DEMO

Wi-Fi authentication platform



Summary

- We have presented two-factor authentication tokens, based on the Java Card technology
- We have introduced the open code project *OpenEapSmartcard*, which is used by these token
- We have built an authentication architecture fully based on IETF standards
- We have shown a real Wi-Fi platform that deals with these technologies

For More Information

- http://www.enst.fr/~urien/openeapsmartcard
- http://tools.ietf.org/wg/eap/draft-urien-eap-smartcard-12.txt.
- Urien, P, Badra, M, “EAP-TLS smartcards, from dream to reality”, http://ieeexplore.ieee.org
- Urien, P. Dandjinou, M, “Introducing Smartcard Enabled RADIUS Server”, http://ieeexplore.ieee.org
- http://www.ethertrust.com

Q&A

Pascal Urien

Guy Pujolle

Java Card™ Technology for Emerging WLAN Environments

Pascal Urien

Professor
ENST
http://www.enst.fr

Guy Pujolle

Professor
LIP6
http://www.lip6.fr

TS-0285