A Raconteur's Java Card™ Technology Overview and How to Work the University Research Bureaucracy

Seth Meltzer, Doris Baker

Headquarters' Research
IRS

TS-9764

Goal of This Talk

Explain Smartcard and Java Card technology motivation and basics

Share Lessons learned from university research (a year as a Carnegie-Mellon Professor)

Agenda With Section Highlights

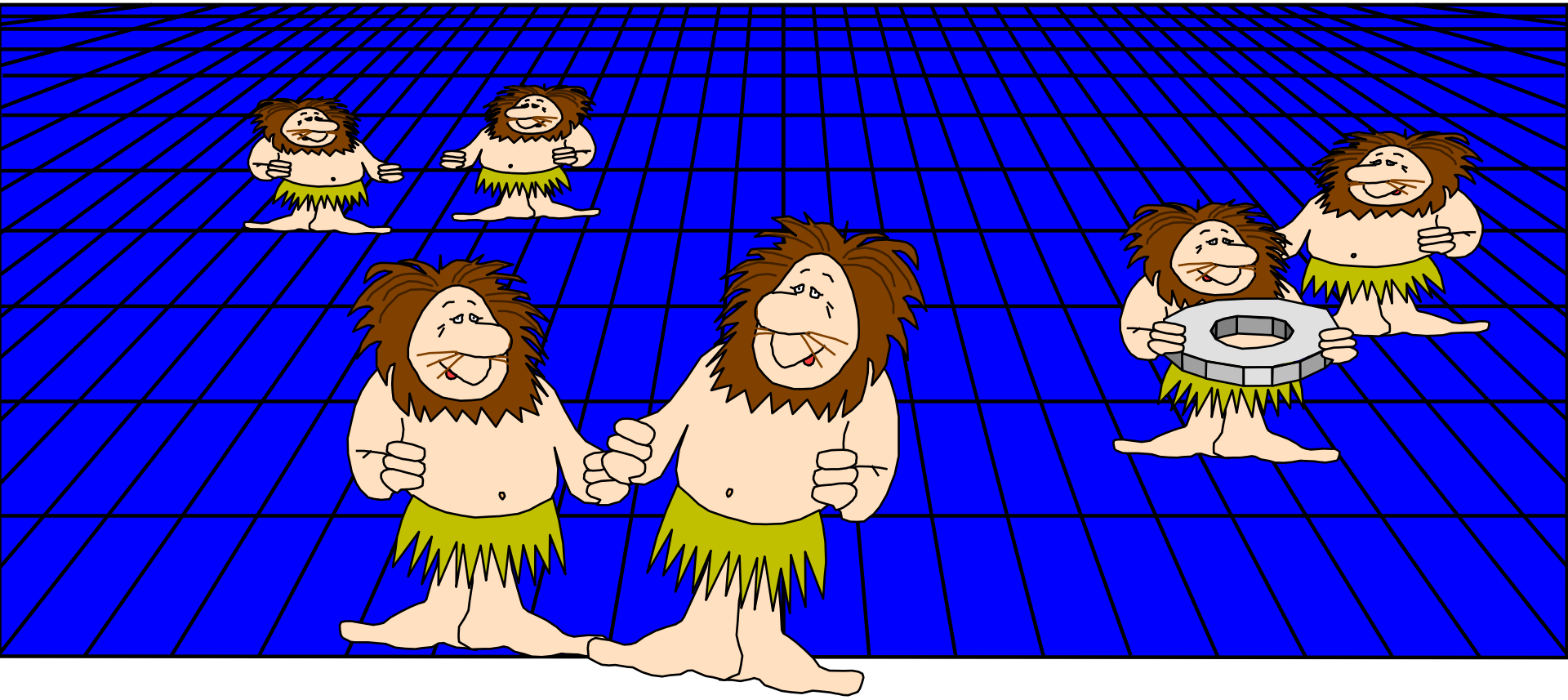
- Authentication
  - Brief history
  - Why cryptography is used
  - Cryptographic challenge/response
  - Authentication pitfalls and solutions
- Smartcards as a token for digital credentials
  - Smartcard basics
  - Authentication with Smartcards
- Java Card technology motivation and basics

Agenda With Section Highlights

- Java Card technology development with various vendor
  - APIs
  - Manufacturer/Industry
- Javacard customized development
  - Authentication
  - Locked wallet
- Lessons learned from university research
- Code Samples
- More Info

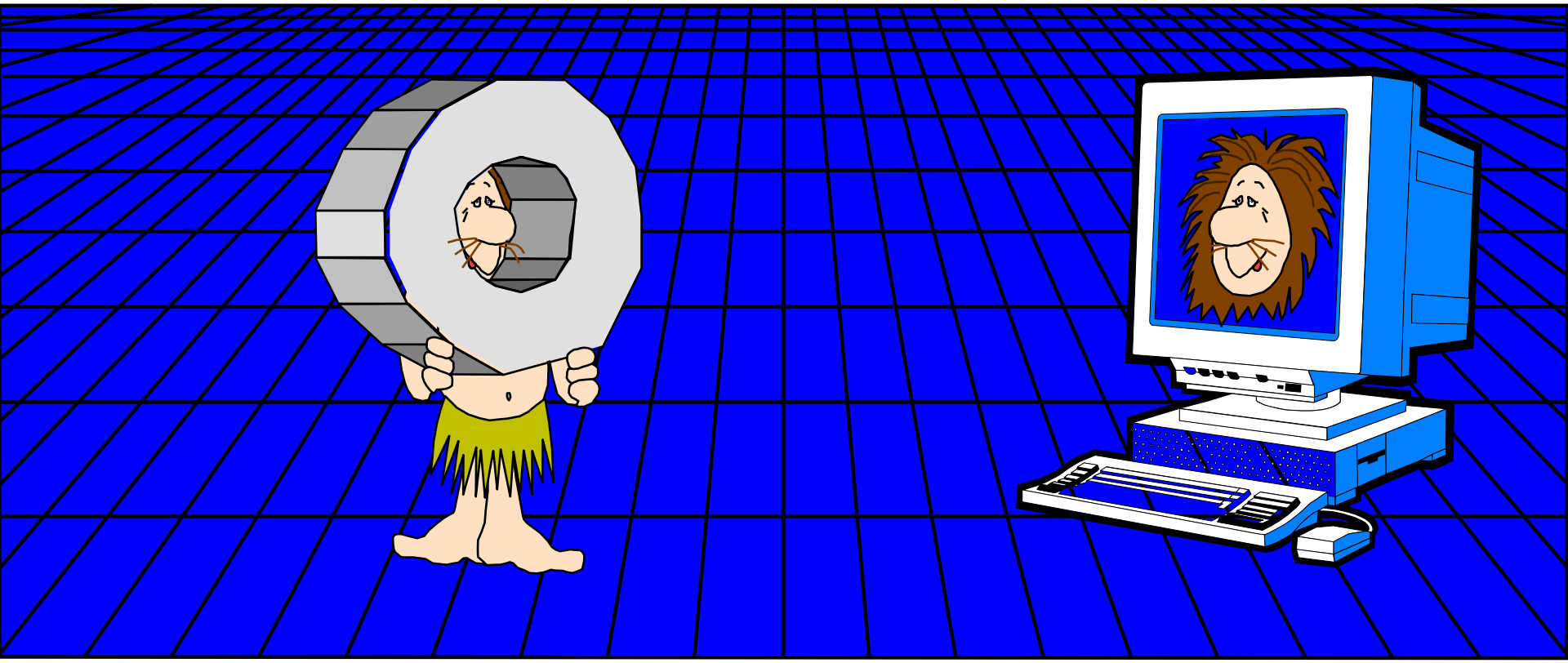
History of Authentication

A Long Time Ago



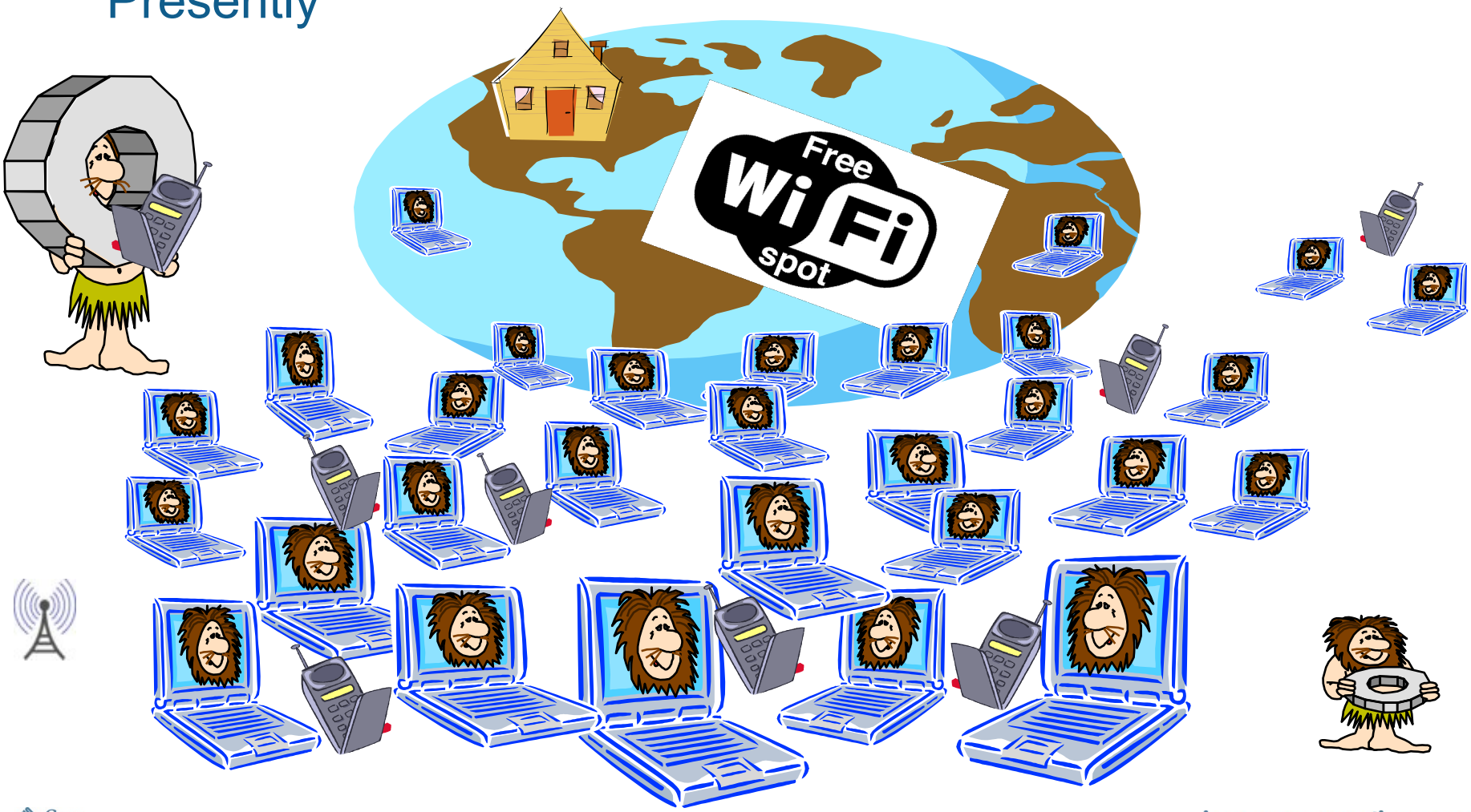
History of Authentication

A Short Time Ago



History of Authentication

Presently



Bio and IRS Headquarters' Research

Before Computers

Government Research Lab

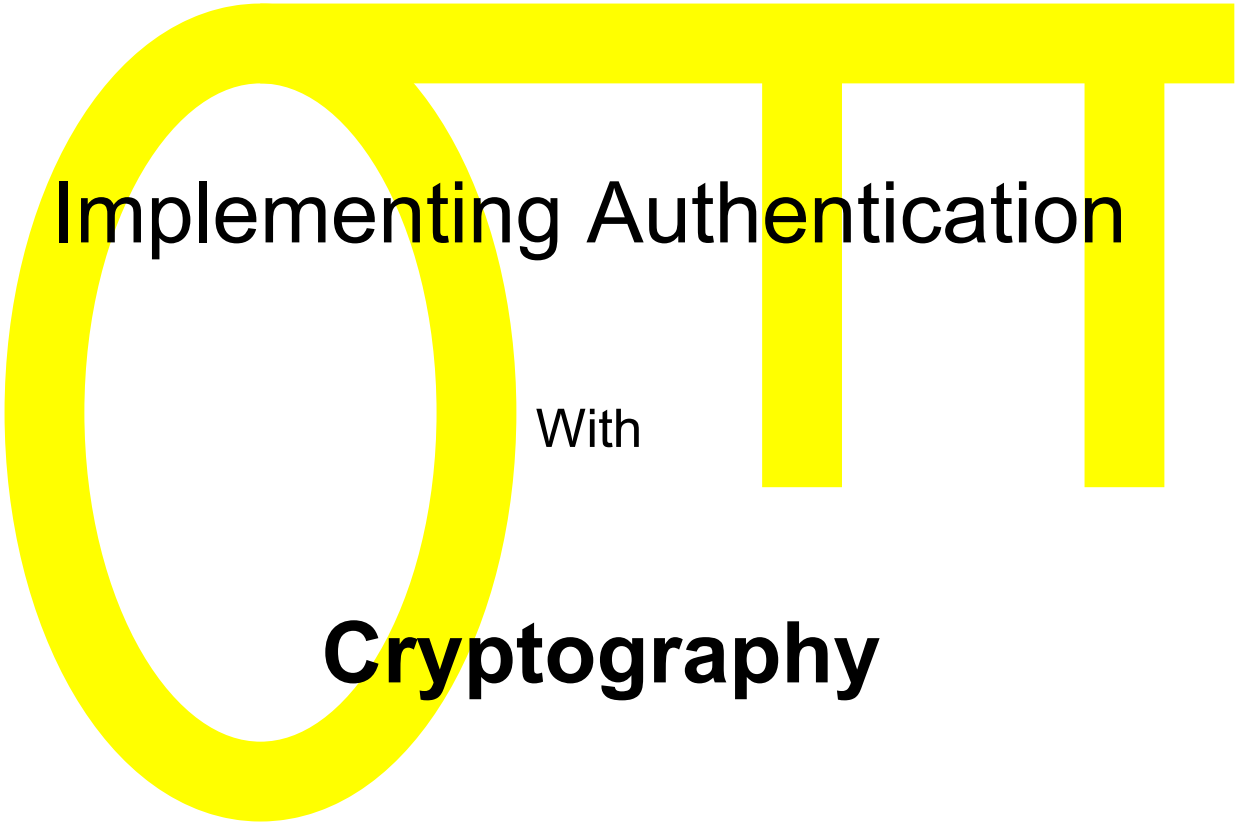
Advanced technology

- Data mining
- NLP
- Cryptography

Partnering

(gov't industry academia)

- Behavioral modeling
- Smartcard



Implementing Authentication

With

Cryptography

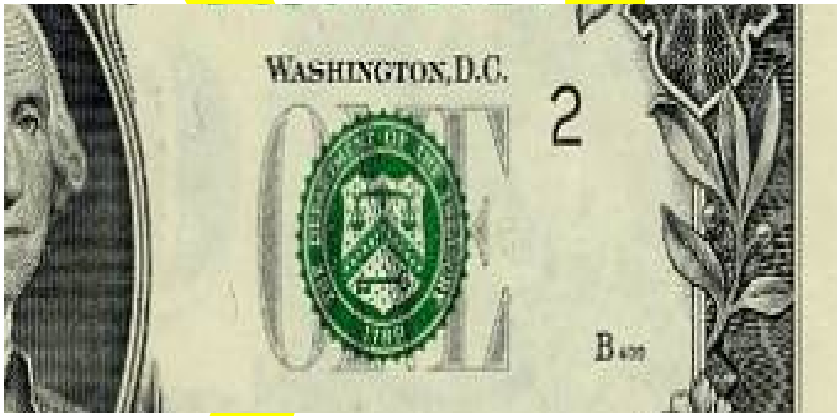
Key Centric

Key Man

Key Word

Key Data

Key to the City



Key Phrase

Key Play

Locks and Keys

Key to Your Heart

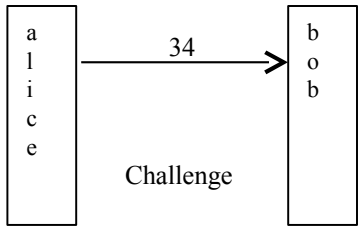
Key

Cryptography Is Key Centric

Only Need to Trust

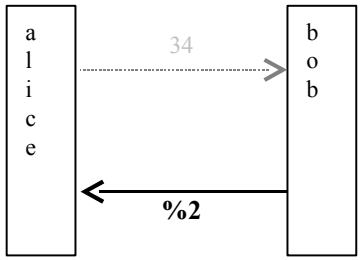
Security of Your/Their Keys

Authentication With Challenge/Response



Alice Challenges

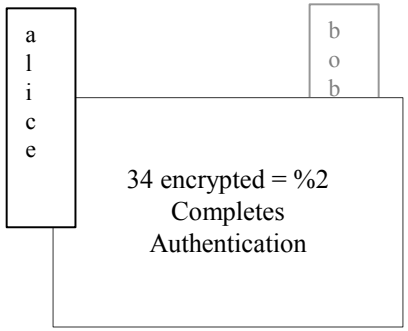
2. Picks a number between 1 and 100 (e.g. 34)
3. Encrypts 34 → %2 (see bottom picture)
4. Challenges the requesting computer to encrypt 34



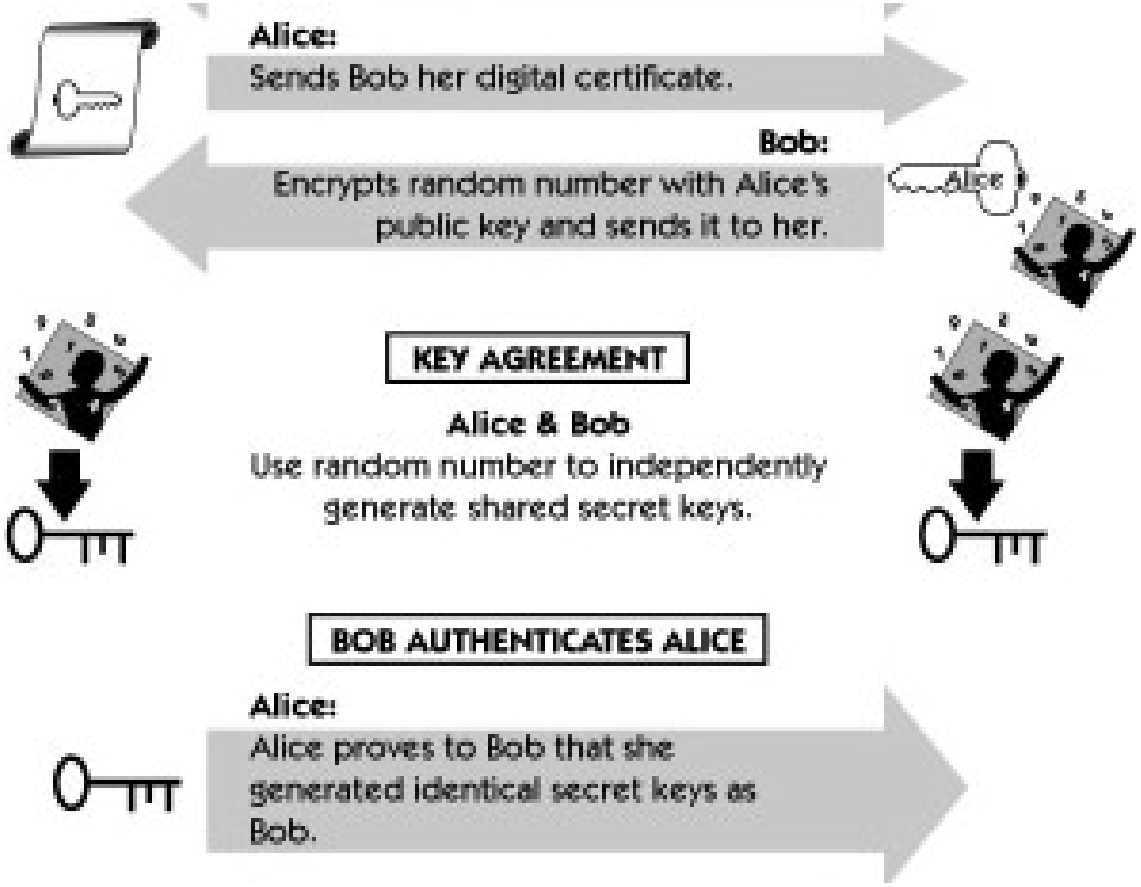
Bob Responds

He encrypts 34. Say 34 encrypts to '%2'
He sends **%2** back to Alice

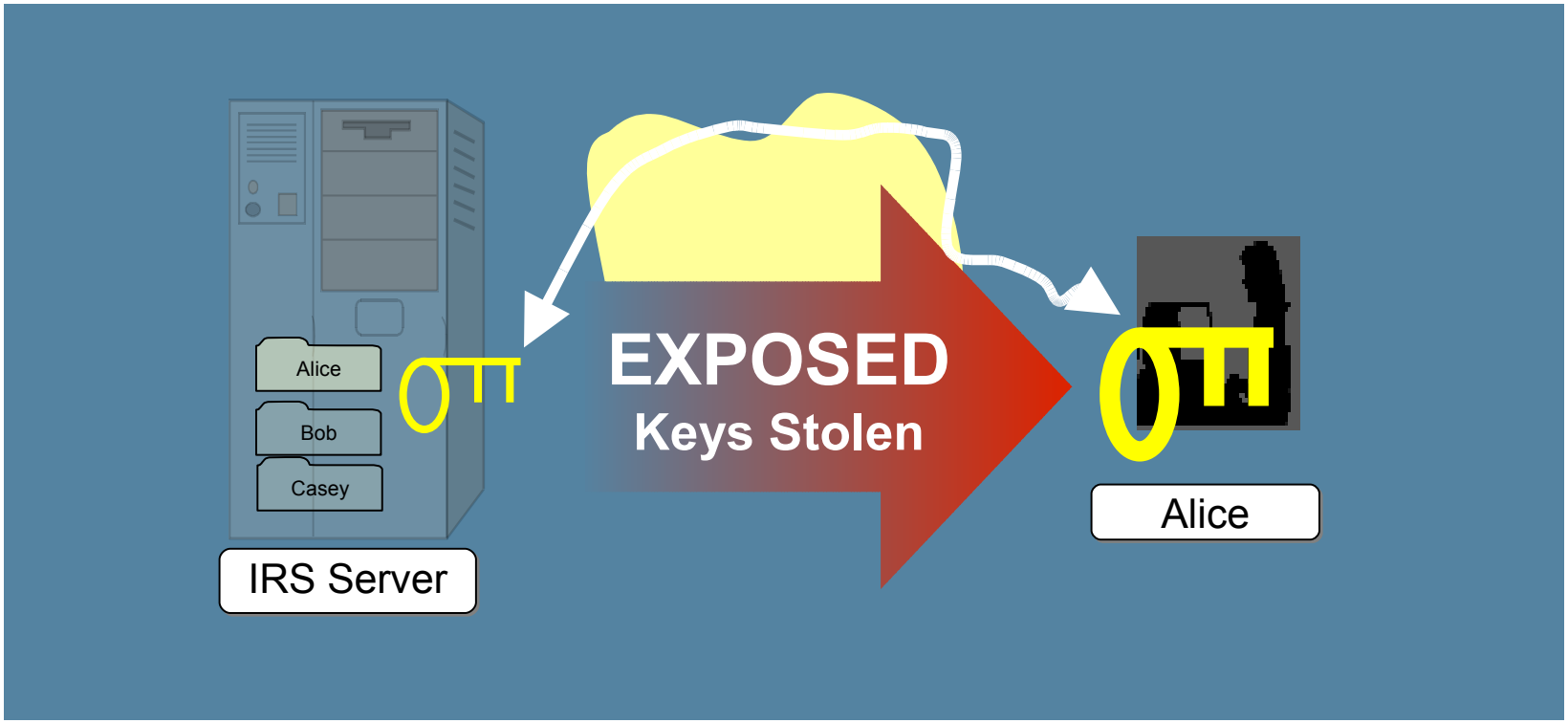
Alice has authenticated Bob



SSL Authentication



Authentication Pitfall



Alice assured Correspondent (IRS Server)
Knows (Shares) identical secret key
Without divulging what key is!

Internet Communications

Authenticating who's there?

know (password)

have (smartcard)

are (biometric)

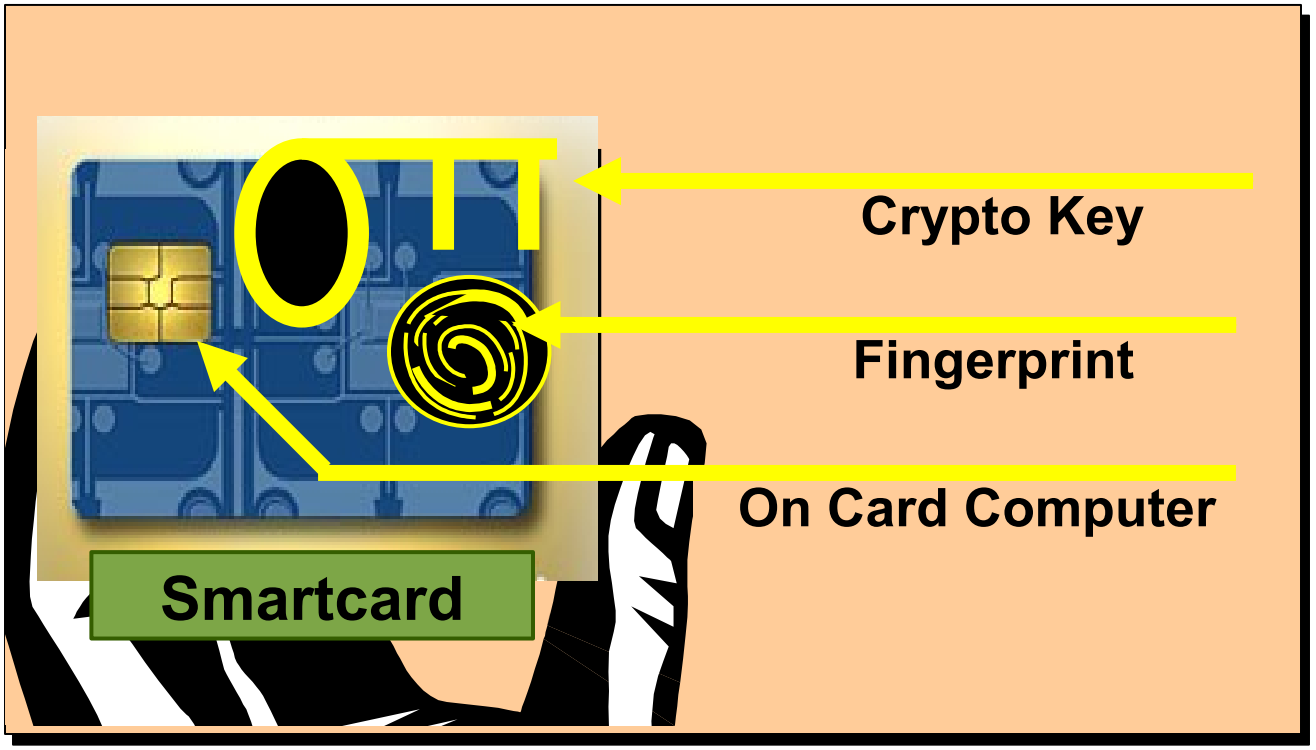
Smartcard Motivation

1. Authentication

2. Off-load processing (protected purse)

Source: Please add the source of your data here

SmartCard With Crypt Key and Biometric Fingerprint



Smartcard Computer

Motivation:

Keep crypt keys on ‘relatively’ safe computer

Mindset change: smartcards are a computer

New nomenclature

sc ←→ server ←···· ···· ····→ client ←→ sc

(host) (host)

Two Types of Smartcards

Memory Cards

- No onboard microprocessor
- Limited functionality
- Advantage is simple technology
  - e.g. Prepaid phone cards

Microprocessor (Smart) Cards

- CPU: 16, 32, even 64
- Often have coprocessor for crypt math
- Memory: (see later slide)

Source: Please add the source of your data here

Many, (Too) Many Standards

Global Platform (GP) (formerly Open Platform)

Common Criteria (CC)

International Airline and Transportation Association (IATA)

Global System for Mobile Communication (GSM) Standards

EMV 2000 Specifications. (EMV v4.0 consists of 4 books)

Book 1, *Application-Independent ICC to Terminal Interface Requirements*
Book 2, *Security and Key Management*
Book 3, *Application Specification*
Book 4, *Cardholder, Attendant, and Acquirer Interface Requirements*

Personal Computer/Smart Card (PC/SC) Workgroup Open Specifications

OpenCard Framework

The Health Insurance Portability and Accountability Act (HIPAA) of 1996 (Public Law 104-191)

International Civil Aviation Organization (ICAO), Passport Guidelines

Card/Host Smartcard Software

Card

- Small subset
  - Java card Java VM (Java Card VM) is less than 1% of JDKTM 1.5 release
- Doesn't trust anything
  - Assumes working in hostile environment
  - Unless authenticated cards don't trust host (& vice-versa)
  - Usually cards act in slave mode to host master

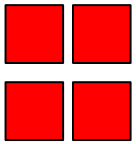
Host (aka off-card or terminal software)

- Runs on desktop or server that sc is attached to
- Written in high level software (C/C++, Java technology, etc.)

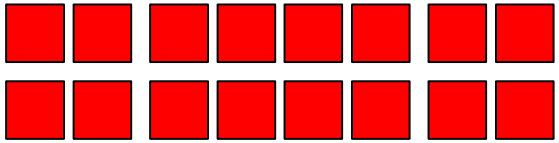
Three Kinds of SmartCard Memory

ROM

EEPROM



RAM



ROM Memory

Persistent data (no power necessary)

Can’t be changed after manufacture

~ 100 Kb

EEPROM Memory

Persistent—modifiable data

Limit: 100,000 Erasers/~ 10 years

Slow

~ 100 Kb

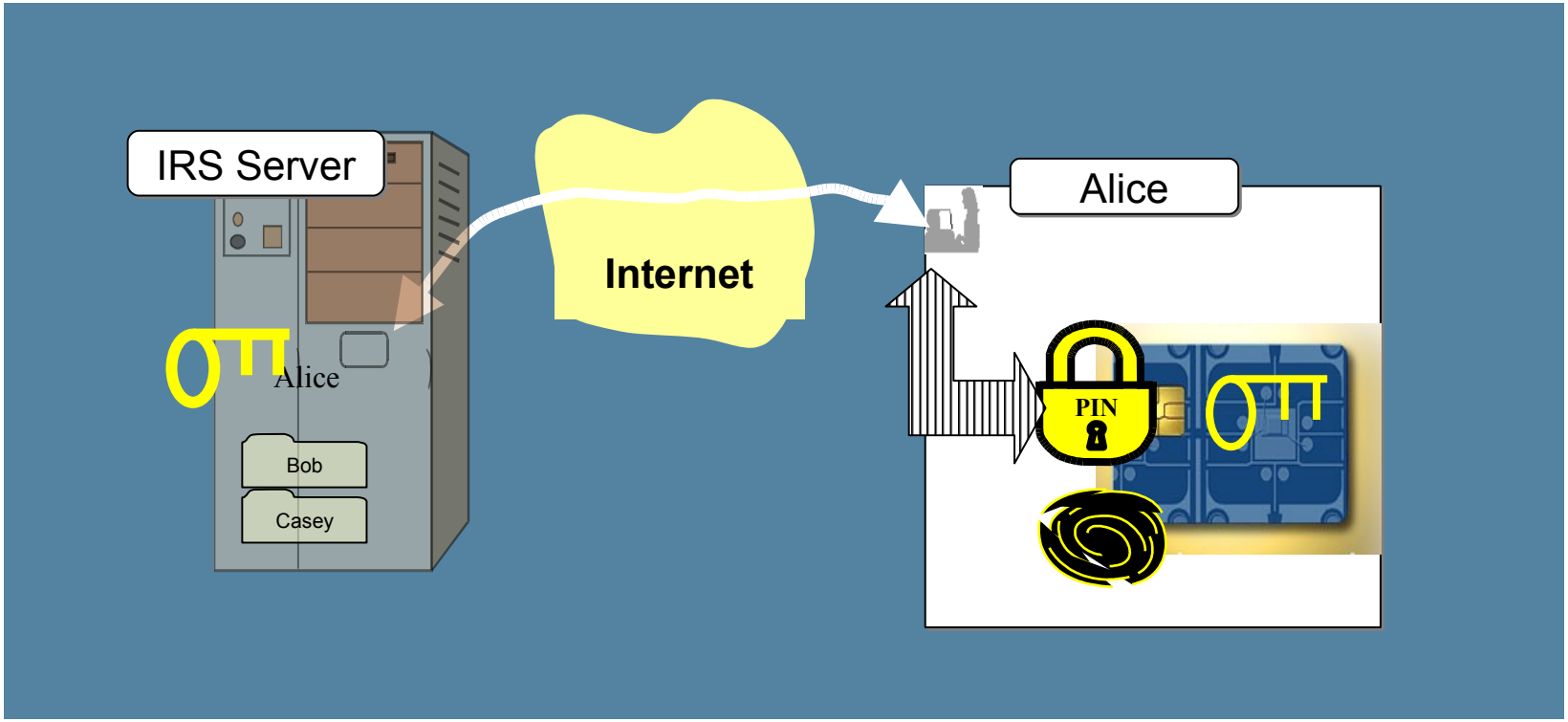
RAM Memory

Not persistent (volatile) data

Fast

~ 5 (or even less) Kb

Authentication With Smartcard



Alice assured Correspondent (IRS Server)
Has smartcard and Knows PIN
Is biometric identical

Java Card Technology

Previously

- All smartcard software was burned onto card by manufacturer

1997

- Schlumberger (now Axalto) enables dynamically loaded Java technology based pgrms (Java Card based applets)

Note: Java Card based applets ≠ browser applets

Java Card Technology Development

Initial Attempt

Java Card Technology Vendor API

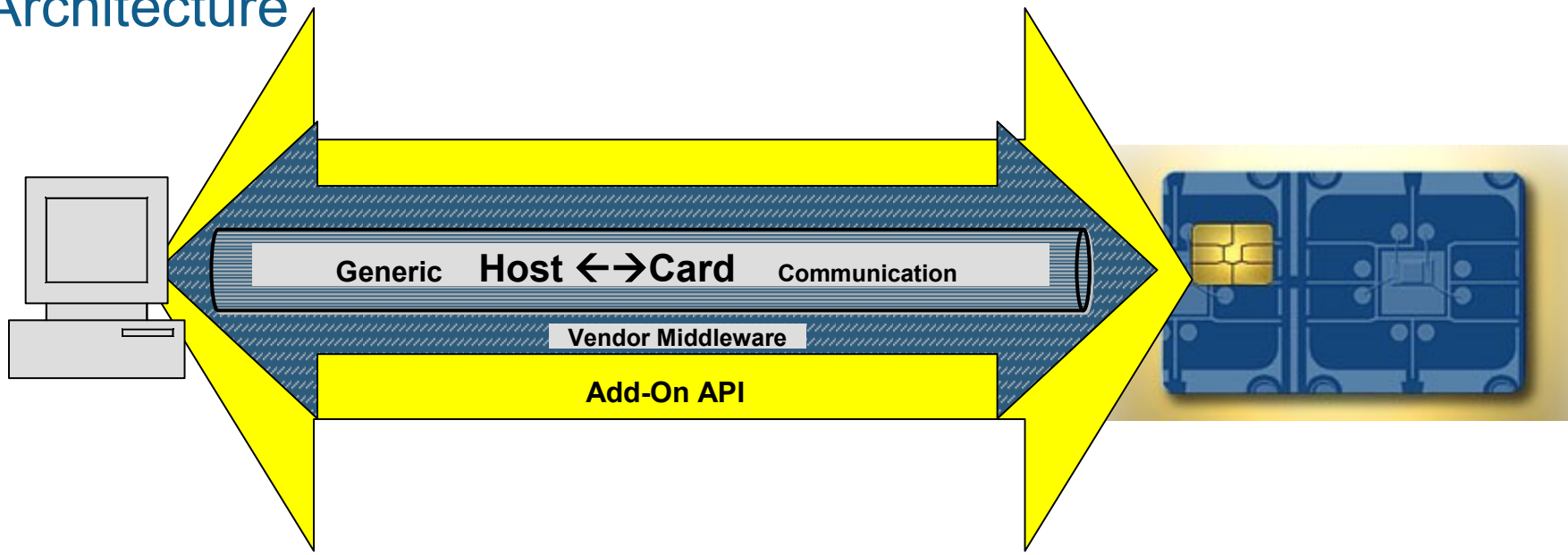
- C/C++ method wrapped in Java programming language call
- Limited
- Proprietary

3rd Party Add-ons

- Limited set of cards
  - Really a product—Or a Market Test

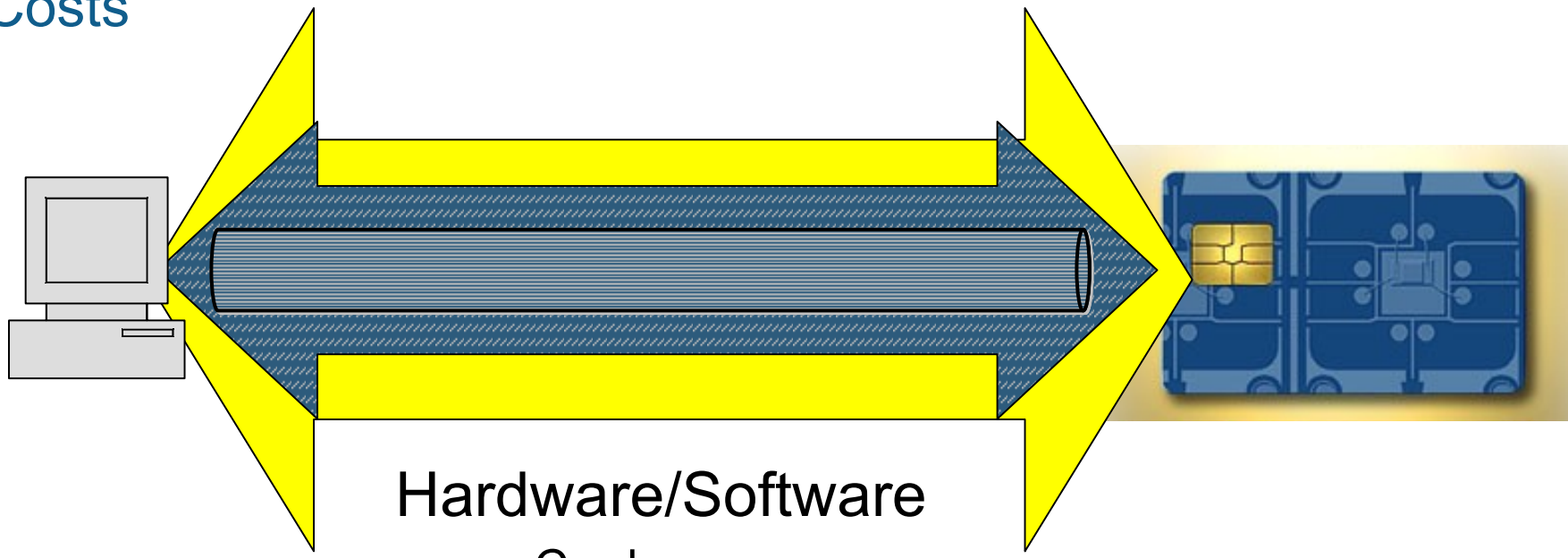
Host ←→ Java Card Technology—Vendor API

Architecture



Host ←→ Java Card Technology—Vendor API

Costs



Hardware/Software

- Cards
- Vendor API

Wetware (people)

Experience: ActivCard, Phaos,

Java Card Technology Development

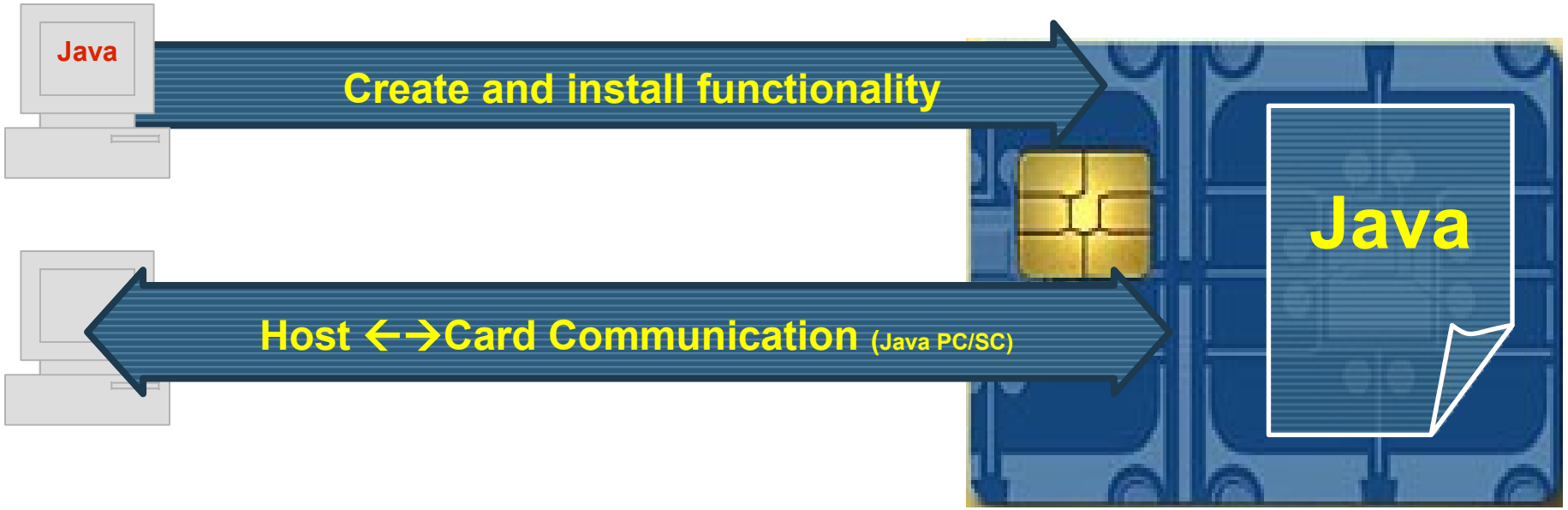
Final Prototype

Java Card API/Development kit

- Limited
- Not Proprietary
- Substantial effort

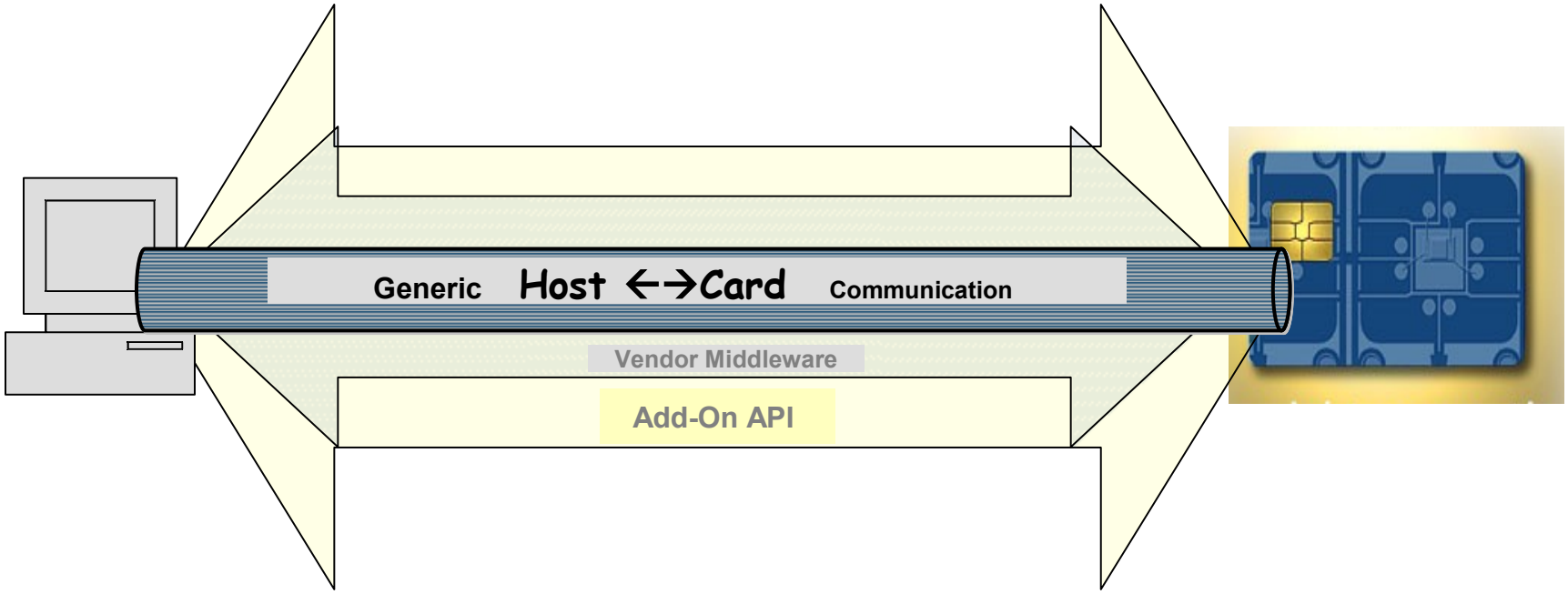
Experience: Card: Jcop, Axalto, Gemplus, Aspects, Host: J-PC/SC

JavaCard Custom Development

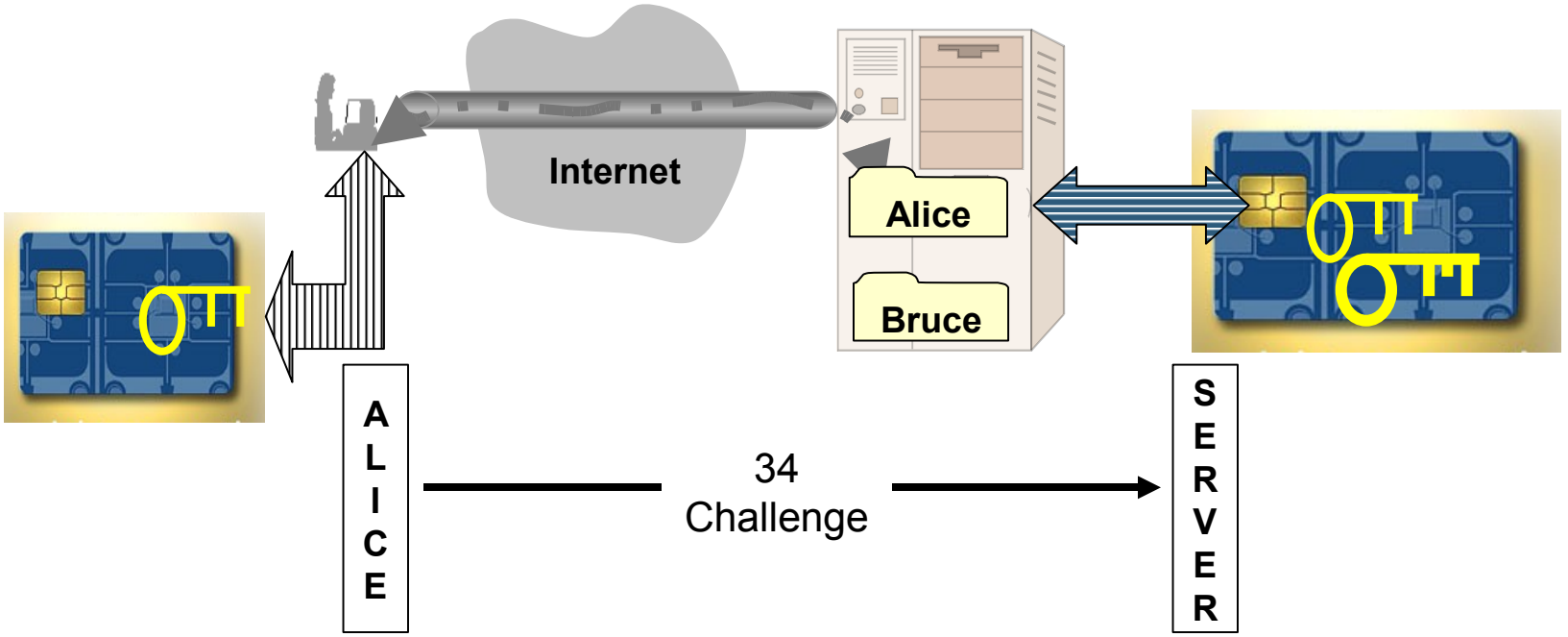


Two faceted development:

Host ←→ Java Card Technology—Customized

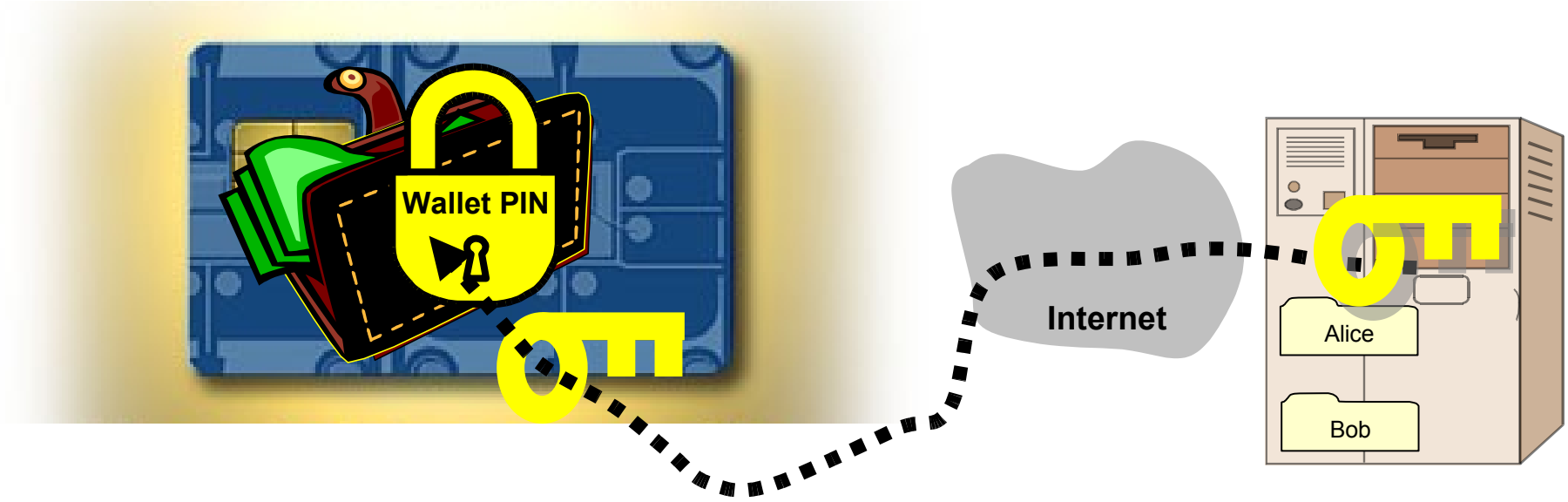


JavaCard-Based Authentication



Secret crypto keys are never exposed
All **private** crypto key processing uses smartcard computer

Locked Wallet Controls Accessibility



Working With University Grad Students

- Can be very productive (but not necessarily)
- “Show me (them) the… ”
  - Using corporate/government goodwill
- Students
- Faculty
- Time tables
- NSF/DARPA

Java Card API Code Sample

Actual Code

Yes	No
Primitive types boolean, byte, short	long, double, float, characters! Strings
One-dimensional array	Threads
Inheritance, virtual functions	G/C,
	Dynamic class loading

JavaCard API Code Sample

```
package gov.irs.sfa.strauss.card;

import javacard.framework.APDU;
import javacard.framework.Applet
import javacard.framework.ISO7816;
import javacard.framework.ISOException;
import javacard.framework.OwnerPIN;
import javacard.framework.Util;
import javacard.security.KeyBuilder;
import javacard.security.RSAPrivateCrtKey;
import javacardx.crypto.Cipher;

import gov.irs.sfa.strauss.Proto;
```

Packages OK

Packages are javacard

Custom Packages

JavaCard API Code Sample

Not for dilettante

```
private
NamedKeyPair()  {
    short S64 = 64;
    privCrt.setP( dd.primeP, ZERO, S64 );
    privCrt.setQ( dd.primeQ, ZERO, S64 );
    privCrt.setDP1( dd.expP, ZERO, S64 );
    privCrt.setDQ1( dd.expQ, ZERO, S64 );
    privCrt.setPQ( dd.crtCoeff, ZERO, S64 );
    if( !privCrt.isInitialized() )
ISOException.throwIt(. . .);
}
```

JavaCard API Code Sample


```
public void process( APDU apdu ) {
    cmdLen = apdu.setIncomingAndReceive();
    short error = transformBuffer( apdu.getBuffer() );
    if( error!=ISO7816.SW_NO_ERROR )     // note: not
        ISOException.throwIt(error);              // throw new
ISOException
    apdu.setOutgoingAndSend(  . . .  );
}
```

JavaCard API Code Sample

```
private short transformBuffer( byte[] apdu ) {
    . . .
  byte ins = apdu[ISO7816.OFFSET_INS];

  switch( ins ) {
       case Proto.GET_ID:           return copyPayload( apdu,
dd.identity );
       case Proto.VERIFY_PIN:       return verifyPIN( apdu );
  }
  . . .
    switch( ins ) {
       case Proto.ENCRYPT:      return encDec( apdu, . . .
       case Proto.DECRYPT:      return encDec( apdu,
Cipher.MODE_DECRYPT . . .     case Proto.GET_MOD:     return
copyPayload( apdu, dd.ownMod );
  . . .
 }
```

JavaCard API Code Sample

```
/** En/decrypt cmd payload to response payload. */

private
short encDec( byte[] apdu, byte cipherMode, short minLen,
short maxLen) {
    if( cmdLen<minLen || cmdLen>maxLen )
        return ISO7816.SW_WRONG_LENGTH;
    cipher.init( privCrt, cipherMode );
    respLen = cipher.doFinal( apdu, ISO7816.OFFSET_CDATA,  . .
.
}
```

Host

```
 /** Transmit a command to the card and receive a reply.
   *  @param hdr a 4-byte APDU header
   *  @param data non-null payload, may be empty
   *  @return success: a non-null, but possibly empty reply payload; failure: null.
  **/
private byte[] sendReceive(byte[] hdr, byte[] payload)  throws Exception {
        if( log!=null ) log.entering("StraussCard", "sendReceive "+name(hdr[1]));
        assert hdr.length==4 && payload.length<=APDU_PAYLOAD_MAX;

        byte[] apdu = ByteManip.cat(hdr, new byte[] { (byte)payload.length }, payload);
        byte[] resp = card.Transmit( apdu, 0, apdu.length );

        short code =  (short)(resp[resp.length-2]<<8 | resp[resp.length-1]);
        if( log!=null ) log.fine("code="+name(code));
        if( code != ISO7816.SW_NO_ERROR )     return null;

        byte[] stripped = new byte[resp.length-2];
        System.arraycopy(resp, 0, stripped, 0, stripped.length);
        return stripped;
}
```

Summary

- Authentication
- Authentication Confidentiality
  Integrity Non-repudiation
- Keep your keys safe

For More Information

Books:

SmartCards—Developers Toolkit—Jurgensen, Guthery

Java Card Technology for SmartCards—Chen

Cryptography Decrypted—Mel, Baker

HxMEL.com

Web:

GSA SmartCard Handbook etc. http://smart.gov

NIST PIV, FIPS . . . http://csrc.nist.gov/piv-program/

SUN: http://developers.sun.com/techtopics/mobility/javacard/articles/javacard1/

Q&A

Seth Meltzer

Doris Baker

A Raconteur's Java Card™ Technology Overview and How to Work the University Research Bureaucracy

Seth Meltzer, Doris Baker

Headquarters' Research
IRS

TS-9764