Smart Cards

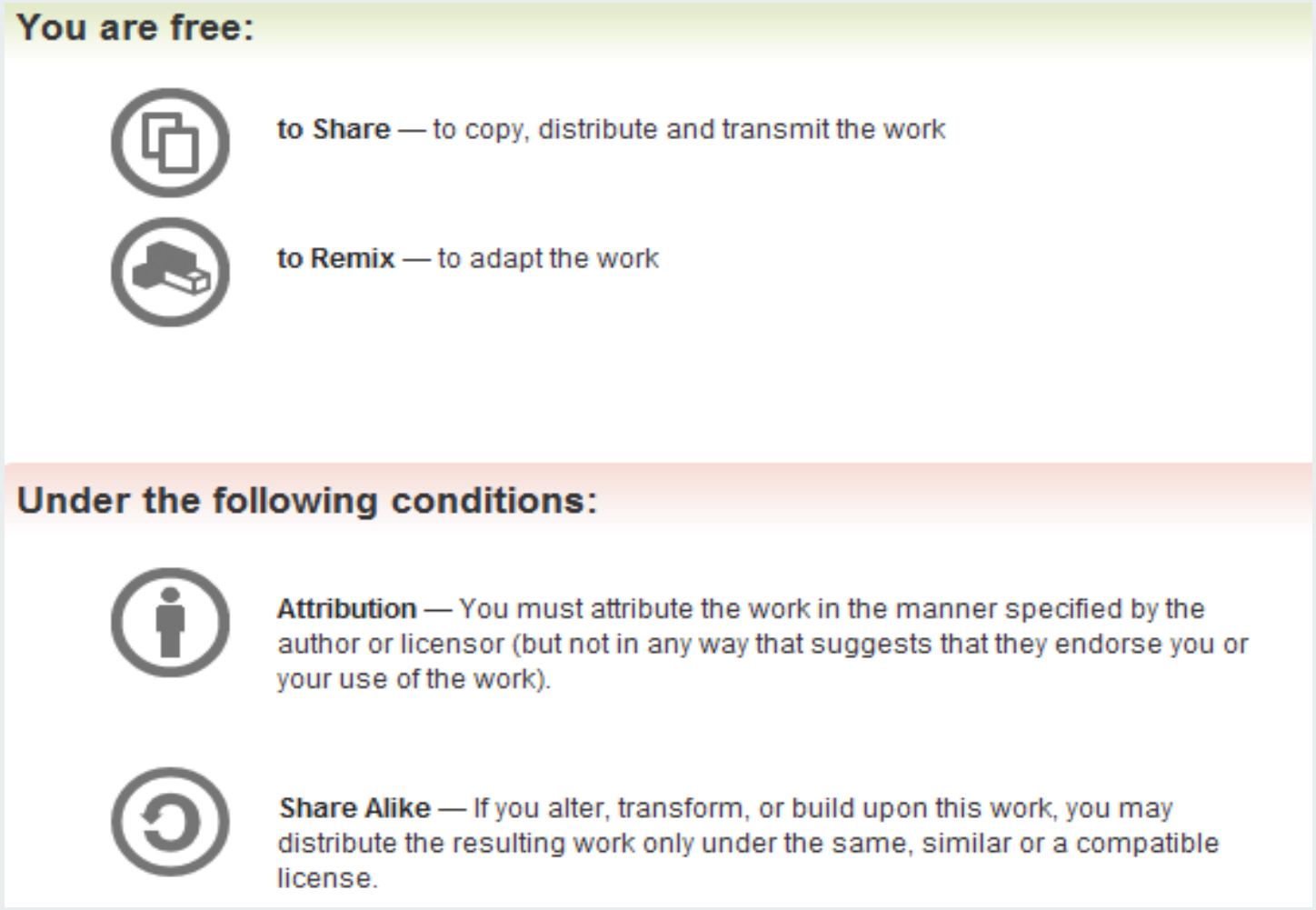
Material last updated in 2007

Ronald van der Knijff

knijff at holmes dot nl

All materials is licensed under a Creative Commons “Share Alike” license.

- http://creativecommons.org/licenses/by-sa/3.0/



Attribution condition: You must indicate that derivative work "Is derived from Ronald van der Knijff's 'Smart Cards' class, available at http://OpenSecurityTraining.info/SmartCards.html"

Smart Cards

Ronald van der Knijff :

- HTS (secondary technical school) electrical engineering::telecom 1987-1991
- Defense: Signals 1992-1993
- University: Computerization 1993-1996
- Netherlands Forensic Institute 1996-
  - Section Digital Technology
  - Specialization Embedded Systems
    - *digital evidence*
    - *fraud*
    - *government expertise*

Program

- Smart Cards
  - Physical and electrical characteristics
  - Operating Systems
  - Physical Attacks
  - Security Mechanisms
  - Security Evaluation
  - Logical Attacks

Study Targets

- Why using a smart card
- What's a smart card and what's not
- How's a smart card structured
- What kind of interface equipment is available
- Working of a smart card OS
- Key developments
- Physical attacks

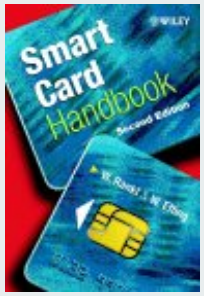
Study Targets

- Purpose and operation of security mechanisms
  - hardware authentication
  - individual authentication
    - identification
    - verification
  - data authentication
    - one-way hashing
    - MAC's
    - signing
    - certificates

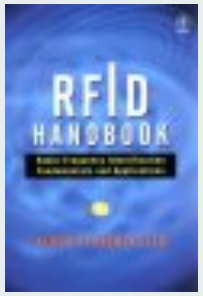
Study Targets

- Purpose and operation of security mechanisms
  - authorization
  - confidentiality
  - symmetric versus asymmetric
- *Attacks*
- Why and how security evaluation
- Why and how risk analysis

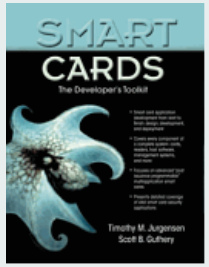
Literature



Smart Card Handbook - W. Rankl & W. Effing



RFID Handbook: Radio-Frequency Identification Fundamentals and Applications - Klaus Finkenzeller
"Fundamentals and Applications in Contactless Smart Cards and Identification"
(2nd English edition, April 2003)



Smart Cards: The Developer's Toolkit -
Timothy M. Jurgensen, Scott B. Guthery
old version: unix.be.eu.org/docs/smart-card-developer-kit

Literature

Smart Card Security and Applications - Mike Hendry

Smart Cards : A Guide to Building and Managing Smart Card Applications - J. Thomas Monk, Henry N. Dreifus

Java Card™ Technology for Smart Cards: Architecture and Programmer's Guide - Zhiqun Chen

Smart Card Application Development Using Java - Uwe Hansmann, Martin S. Nicklous, Thomas Schaeck, Frank Seliger

Why smart cards?

Information society → Information represents value

Digital representation of information → Easy storage, transportation, manipulation and reproduction

Most information systems are open and free accessible → Not always desirable (passwords, credit card data, personal data ...)

Guarantee of confidentiality and integrity demands a closed, inaccessible (mini)system

Why smart cards?

Information society →Services become location and time independent

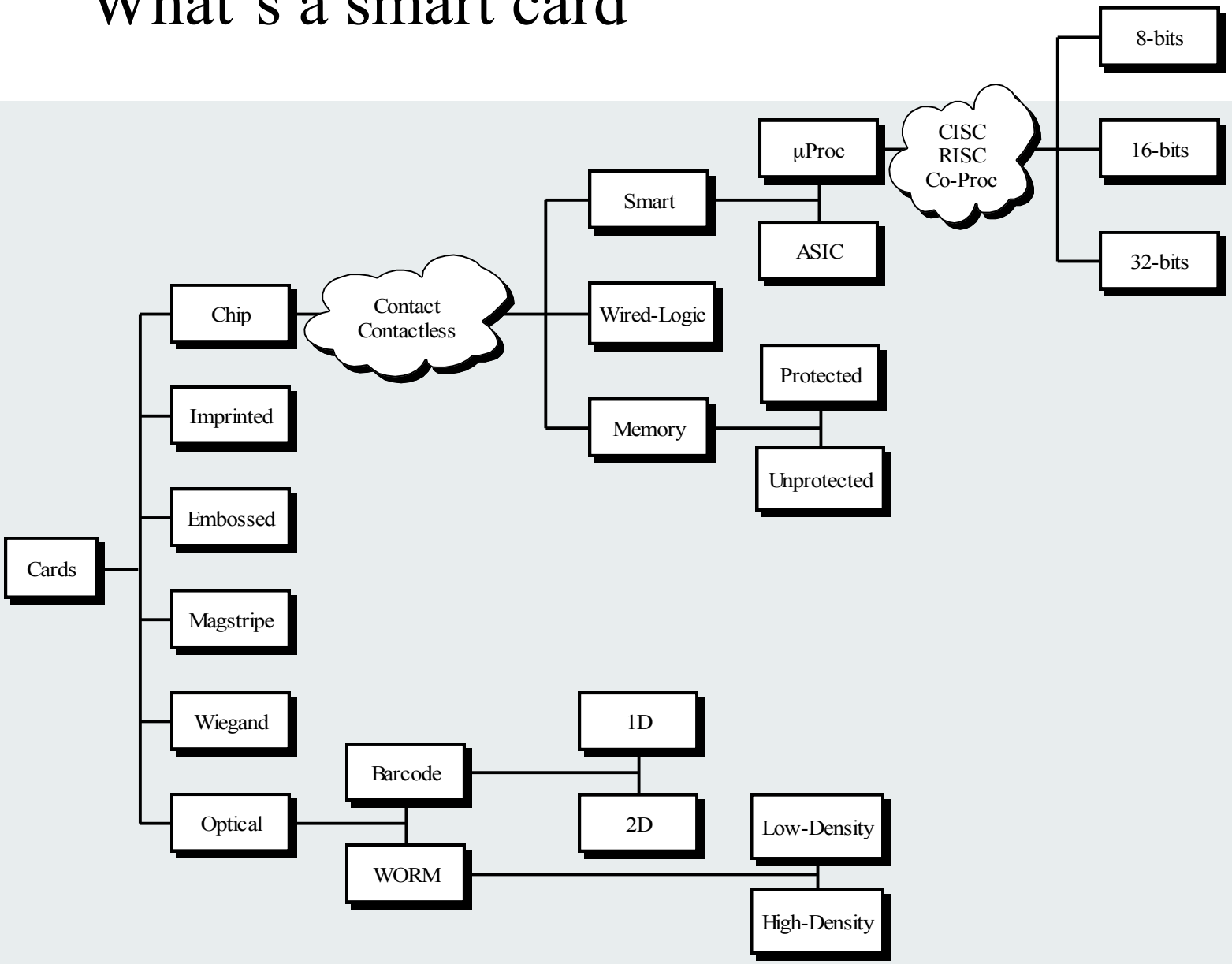
Services require verification of (pseudo) identities

On-line communication → Expensive and vulnerable(because of central storage and processing) → Off-line with local distributed storage seems attractive alternative

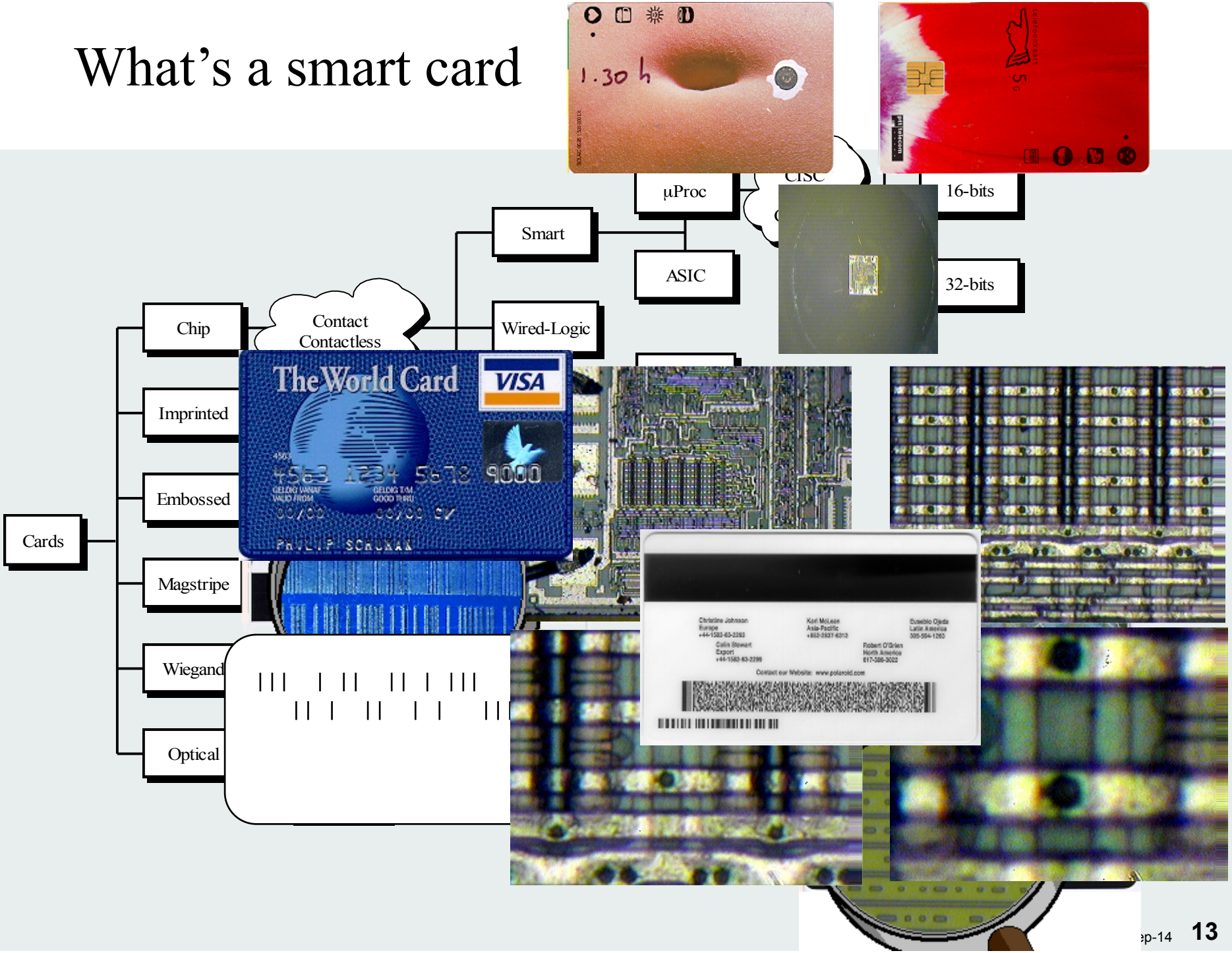
Central storage of personal information raises privacy concerns → With local storage the user protects his/her privacy

<u>Smart card is a portable, protected mini-archive</u>

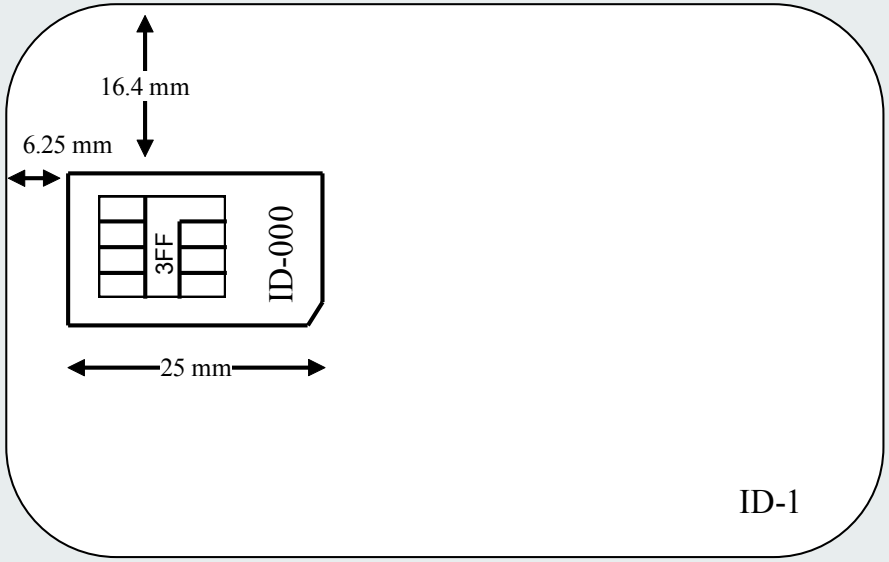
What's a smart card



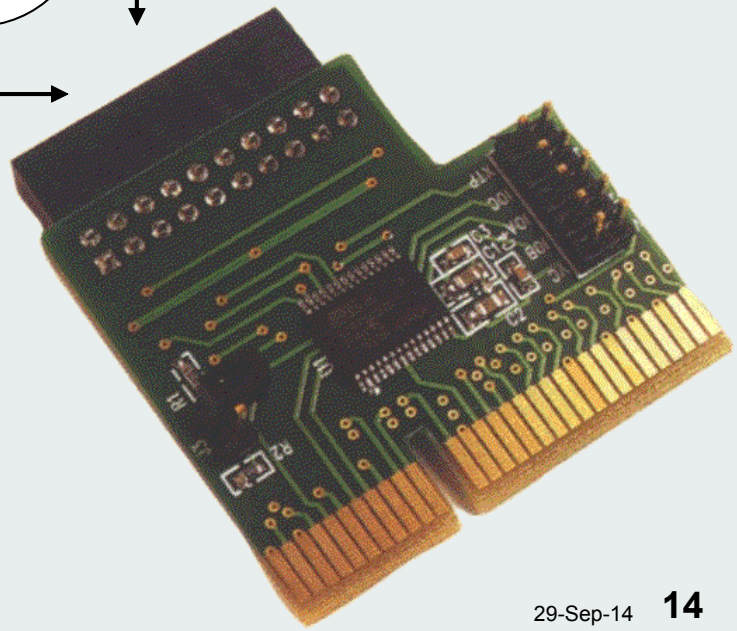
What's a smart card



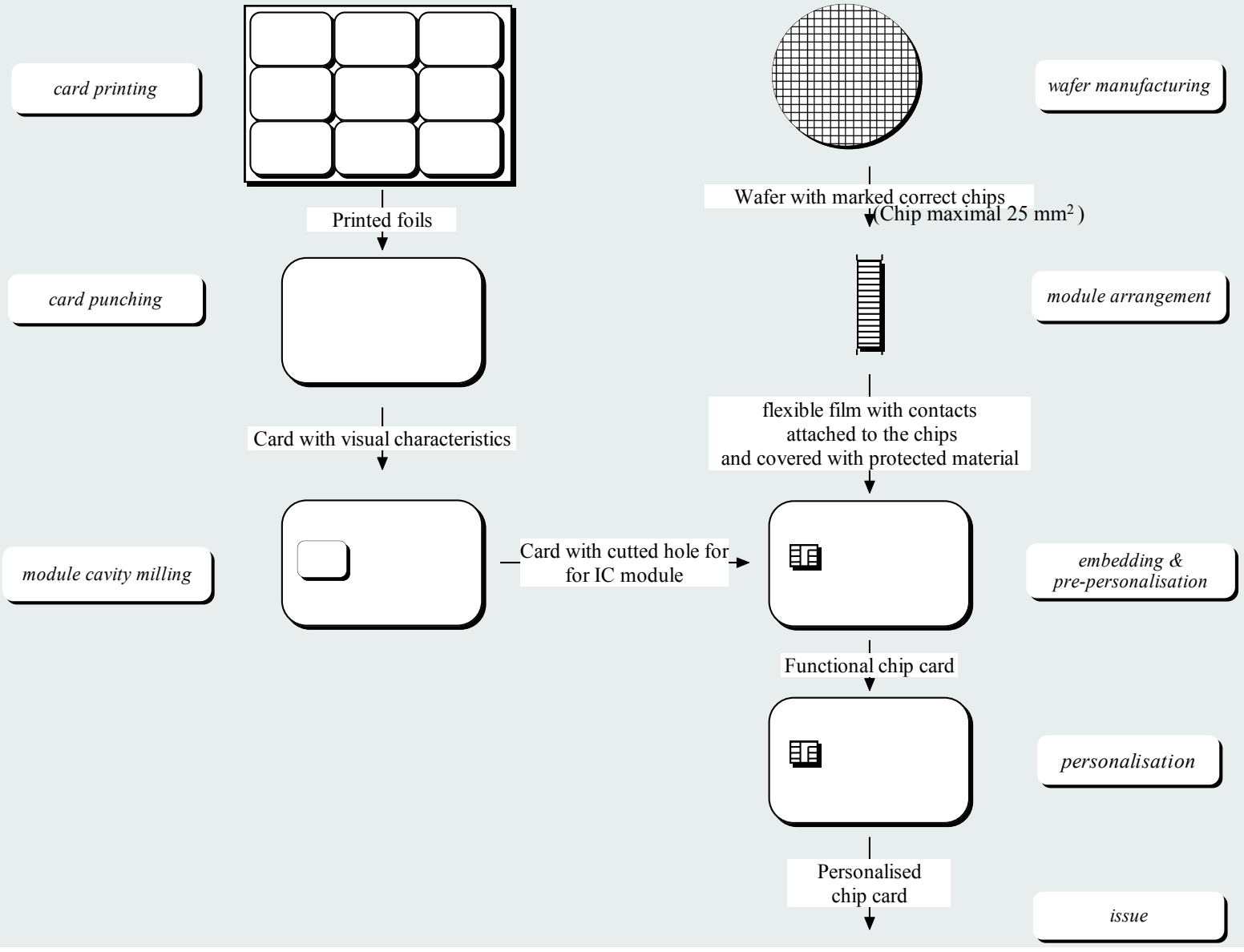
(Smart) Card dimensions



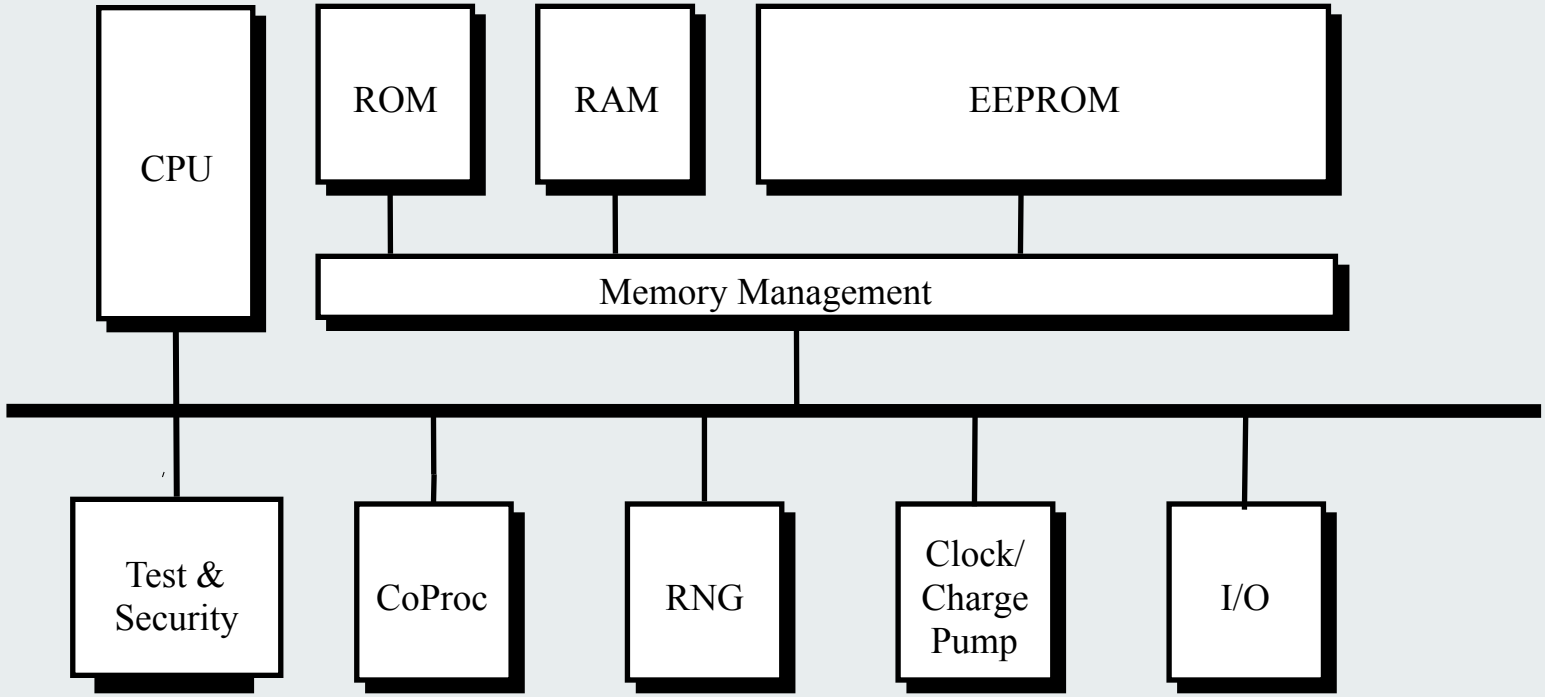
C1=Vcc	C5=GND
C2=RST	C6=Vpp
C3=CLK	C7=I/O
C4=RFU	C8=RFU



Production (single layer body)



Smart card chip



CPU (Central Processing Unit)

- **8-bit CISC / RISC**
  - **6805 / 8051 / Z80 / H8 / AVR**
- 16-bit CICS / RISC
  - 8051XA / H8 / ARM
- 32-bit RISC
  - ARM / MIPS

- Need for more processing power (virtual machines, cryptography)

ROM (Read Only Memory)

- Permanent storage
- “Filled” during production (mask)
- Contains static part OS + test & security
- Capacity 8- 512 KB
- Optical readable after removal of top layers
- Flash is coming up for prototyping, small quantities and as ROM/EEPROM replacement

EEPROM (Electrically Erasable ROM)

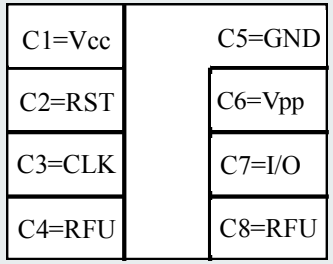
- Non volatile, re-writable memory
- Write=Erase + Program
  - non-atomic
  - slow
  - requires high programming voltage (≈15 Volt)
  - limited (10.000 - 100.000 times)
  - data retention (10 - 100 years)
- Contains file system and dynamic part of OS
- Capacity 1- 512 KB and increasing (1 MB FLASH from Sharp)
- Readable via micro-probing, SEM en FIB
- FRAM in future?

RAM (Random Access Memory)

- Volatile re-writable memory (SRAM)
- For storage of temporary data
  - stack
  - heap
  - I/O buffer
- Capacity 128- 16384 Bytes and increasing
- Volatile but still small permanent storage characteristics

I/O (Input/Output)

- Contact Interface

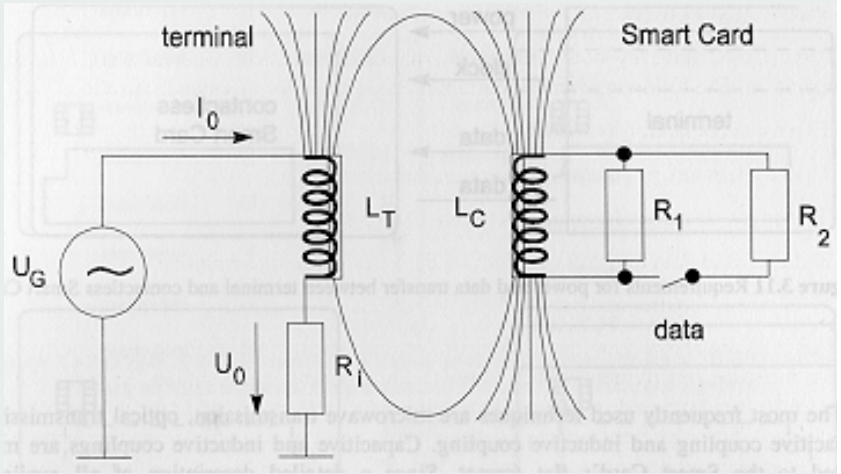


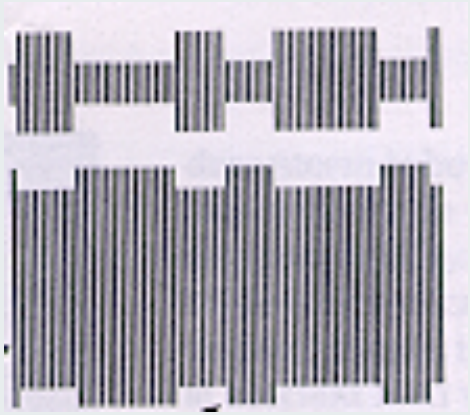
  - » Vcc = 5 Volt (3 Volt)
  - » Vpp not used anymore
  - » CLK (3.5712, 4.9152, 10 MHz.)
  - » UART voor I/O

- Contactless Interface (125 kHz & 13.56 MHz)
  - *Close coupled*, a few millimeters
  - *Proximity*, less than 10 centimeter
  - *Vicinity*, more than 10 centimeter

Contactless Interface

- Power from CAD
- Modulation:
  - CAD →Card : AM, FM, PM
  - Card → CAD: AM
- Anti collision





CoProcessor

- Some cryptographic algorithms cannot be implemented fast enough on smart cards CPU's
- Coprocessor is a specially developed arithmetic unit (with own RAM) for computations relating to cryptographic algorithms
  - Symmetric algorithms: DES computation unit (takes roughly the same chip space as ROM program code)
  - Asymmetric algorithms: Exponentiation and modulo arithmetic on large numbers
- Low level calling: CPU prepares input data, *Escapes* to CoProc and processes output data
- High level calling: CryptoAPI

CoProcessor

Implementation	Mode	512 bits	768 bits	1024 bits
Smart Card without NPU, 3.5 MHz	signing	20 min	---	---
Smart Card without NPU, 3.5 MHz (with Chinese remainder theorem)	signing	6 min	---	---
Smart Card with NPU, 3.5 MHz	signing	308 ms	910 ms	2000 ms
Smart Card with NPU, 3.5 MHz (with Chinese remainder theorem)	signing	84 ms	259 ms	560 ms
Smart Card with NPU, 4.9 MHz	signing	220 ms	650 ms	1400 ms
Smart Card with NPU, 4.9 MHz (with Chinese remainder theorem)	signing	60 ms	185 ms	400 ms
PC (Pentium, 200 MHz)	signing	12 ms	46 ms	60 ms
PC (Pentium, 200 MHz)	verification	2 ms	4 ms	6 ms
RSA integrated circuit	signing	8 ms	---	---

CoProcessor

Fumihiko Sano* Masanobu Koike* Shinichi Kawamura† Masue Shiba*

Cipher	RAM (bytes)		ROM (bytes)		Time (clock) Encrypt		Schedule		Encrypt + Schedule		
MARS	572	5	5,468		45,588	4	21,742	2	67,330	3	*
RC6	156	3	1,060	2	34,736	3	138,851	4	173,587	4	
Rijndael	66	1	980	1	25,494	1	10,318	1	35,812	1	only encryption
Serpent	164	4	3,937	4	71,924	5	147,972	5	219,896	5	
Twofish	90	2	2,808	3	31,877	2	28,512	3	60,389	2	
DES	17		772						25,398		
Triple DES	17		849						72,341		
MISTY	44		1,598						25,486		

*: omit to check "weak" in the key schedule.

CoProcessor

CryptoBytes RSA Volume 4, Number 1
http://www.rsasecurity.com/rsalabs/cryptobytes

Chip		P83W854/-858	P83W8516/-8532	SLE44CR80S	SLE66CX160S	µPD789828
Internal Clock Frequency		independent	independent	5 Mhz	5 Mhz	40 Mhz
DES	64 bits	10 ms @ 5 Mhz	10 ms @ 5 Mhz	3.7 ms*	3.7 ms*	4 ms
SHA	512 bits	10 ms	5 ms	5.6 ms*	5.6 ms*	< 2 ms
MD5	512 bits	N/A	N/A	9 ms*	9 ms*	N/A
RSA 512	Sign with CRT	45 ms	37 ms	60 ms	37 ms	16 ms
RSA 512	Sign without CRT	140 ms	93 ms	220 ms	110 ms	52 ms
RSA 512	Verify ($e = F_4$)	22 ms	10 ms	20 ms*	10.3 ms*	2 ms
RSA 768	Sign with CRT	182.5 ms	88 ms	250 ms*	124 ms*	52 ms
RSA 768	Sign without CRT	385 ms	220 ms	N/A	437 ms*	164 ms
RSA 768	Verify ($e = F_4$)	36 ms	18 ms	N/A	18.4 ms*	4 ms
RSA 1024	Sign with CRT	250 ms	160 ms	450 ms	230 ms	100 ms
RSA 1024	Sign without CRT	800 ms	400 ms	N/A	880 ms	360 ms
RSA 1024	Verify ($e = F_4$)	50 ms	25 ms	N/A	24 ms*	7 ms
RSA 2048	Sign with CRT	2180 ms	1100 ms	N/A	1475 ms*	750 ms
RSA 2048	Sign without CRT	21 s	6.4 s	N/A	44 s*	N/A
RSA 2048	Verify ($e = F_4$)	156 ms	54 ms	N/A	268 ms*	45 ms
DSA 512**	Sign	75 ms	58 ms	95 ms	50 ms	31 ms
DSA 512**	Verify	115 ms	82 ms	175 ms	90 ms	70 ms
DSA 768**	Sign	145 ms	100 ms	N/A	N/A	57 ms
DSA 768**	Verify	230 ms	145 ms	N/A	N/A	150 ms
DSA 1024**	Sign	215 ms	150 ms	N/A	143 ms*	N/A
DSA 1024**	Verify	355 ms	225 ms	N/A	271 ms*	N/A
ECDSA 135/131	Sign	N/A	N/A	185 ms	185 ms	N/A
ECDSA 135/131	Verify	N/A	N/A	360 ms	360 ms	N/A
ECDSA 255	Sign	N/A	N/A	N/A	N/A	81 ms
ECDSA 255	Verify	N/A	N/A	N/A	N/A	380 ms

CoProcessor

CryptoBytes RSA Volume 4, Number 1
http://www.rsasecurity.com/rsalabs/cryptobytes

Name	Manufacturer	µC-core Name	CCP Modulus	Max	RAM	ROM	EEPROM	Voltage	Max Ext. Clock	Max Int. Clock	Tech-nology	Sensors
H8/3111	Hitachi	H8/300	Coprocessor	576 bit	800 B	14 KB	8 KB	3V & 5V	10 Mhz	10 Mhz	0.8 µm	LV, LF
H8/3112	Hitachi	H8/300	Coprocessor	576 bit	1312 B	24 KB	8 KB	3V & 5V	10 Mhz	10 Mhz	0.8 µm	LV, LF, HF
H8/3113*	Hitachi	H8/300	Coprocessor	1024 bit	1.5 KB	32 KB	16 KB	3V & 5V	10 Mhz	14.32 Mhz	0.5 µm	LV, HV, LF, HF
T6N29	Toshiba	Z80	1024B	1024 bit	512 B	20 KB	8 KB	3V & 5V	—	—	0.6 µm	V
T6N37*	Toshiba	Z80	1024B	1024 bit	512 B	20 KB	8 KB	3V & 5V	—	—	—	V/T/F
T6N39*	Toshiba	Z80	1024B	1024 bit	512 B	20 KB	8 KB	3V & 5V	—	—	—	V/T/F
T6N42*	Toshiba	Z80	2048B	2048 bit	512 B	20 KB	8 KB	3V & 5V	—	—	—	V/T/F
ST16CF54B	SGS-Thomson	8 bit MCU	MAP	512 bit	512 B	16 KB	4 KB	5V +/- 10%	5 Mhz	5 Mhz	—	—
ST19CF68	SGS-Thomson	8 bit CPU	MAP	512 bit	960 B	23 KB	8 KB	3V,5V +/- 10%	10 Mhz	10 Mhz	0.6 µm	—
ST19KF16	SGS-Thomson	8 bit CPU	MAP	1088 bit	960 B	32 KB	16 KB	3V,5V +/- 10%	10 Mhz	10 Mhz	0.6 µm	—
P83W854	Philips	80C51	FameX	2048 bit	800 B	20 KB	4 KB	2.7V to 5.5V	8 Mhz	—	—	V/T/F
P83W858	Philips	80C51	FameX	2048 bit	800 B	20 KB	8 KB	2.7V to 5.5V	8 Mhz	—	—	V/T/F
P83W8516	Philips	80C51	FameX	2048 bit	2304 B	32 KB	16 KB	2.7V to 5.5V	8 Mhz	—	—	V/T/F
P83W8532	Philips	80C51	FameX	2048 bit	2304 B	32 KB	32 KB	2.7V to 5.5V	8 Mhz	—	—	V/T/F
SmartXA	Philips	16 bit CPU	FameX	2048 bit	1.5/2 KB	32 KB	8/16/32 KB					
SLE44CR80S	Siemens	80C51	CCP	540 bit	256 B	17 KB	8 KB	3V to 5V	7.5 Mhz	7.5 Mhz	0.7 µm	—
SLE66CX160S	Siemens	80C51	ACE	1100 bit	1280 B	32 KB	16 KB	2.7V to 5.5V	7.5 Mhz	7.5 Mhz	0.6 µm	—
µPD789828*	NEC	78K0S	SuperMAP	2048 bit	1 KB	24 KB	8 KB	1.8V to 5.5V	5 Mhz	40 Mhz	0.35 µm	—

* expected in forthcoming months

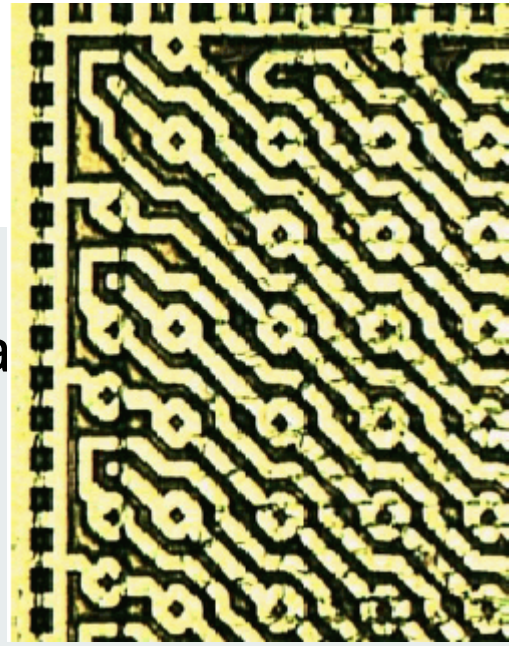
MMU (Memory Managing Unit)

- Important for non-verified (post-issuance) executable code
- Each data object has attributes specifying the physical address space
- Hardware registers check if the object stays in the specified address area
- Exchange of data between applications only via OS routines with authorization mechanisms

Test & Security

- (Self)test hardware and software
  - ROM checksum
  - EEPROM static data checksum
  - RAM dump
  - EEPROM read/write functions
  - Most test hardware and software made useless before personalization
    - fuse blowing not sufficient!
    - hidden instructions will be found!

Test & Security

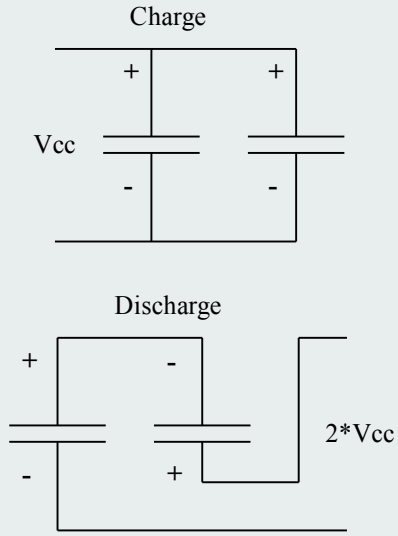


- Smart Card is a tamper resistant device (gra
  resistant, responsive, more in ISd2)
- Penetration:
  - Protective epoxy cover
  - Top-layer sensor mesh
  - Layout scrambling
- Monitoring:
  - Supply voltage
  - Clock frequency and slope
  - Temperature
  - Amount of light
  - Condition of protective layers
  - *Leakage* prevention
- Control only possible with attached power supply!

Markus Kuhn: http://www.cl.cam.ac.uk/~mgk25/

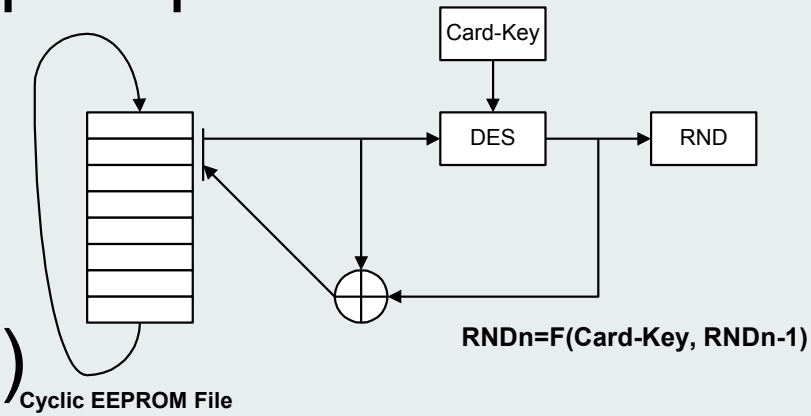
Clock Circuit & Charge Pump

- External clock signal
- Internal clock multiplication possible
- I/O clock divisor of external clock
- Charge pump generates EEPROM programming voltage
  - via external clock and capacitors
  - via local oscillator and capacitors
- Autonomous internal clock is more secure



RNG (Random Number Generator)

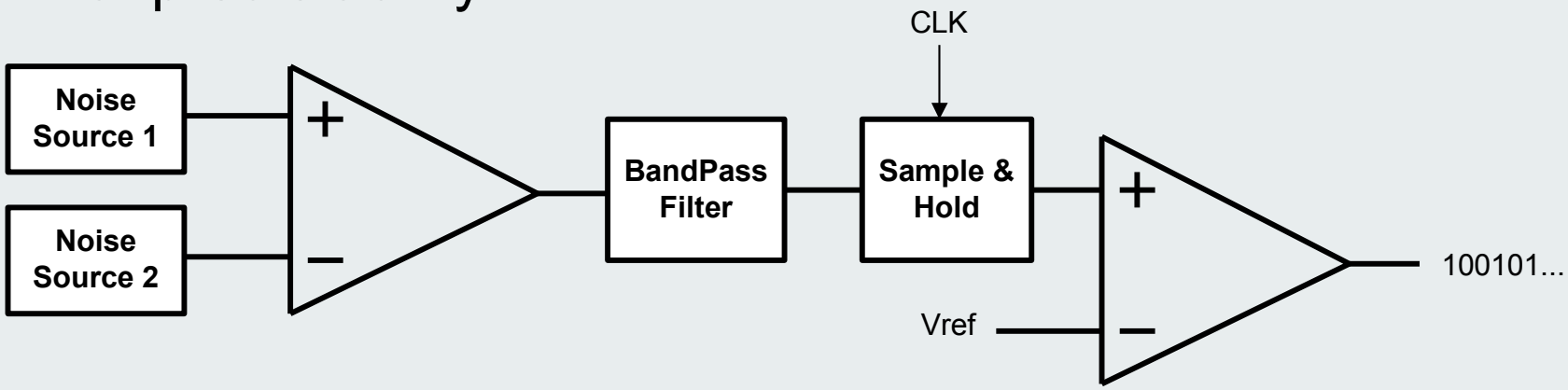
- Needed for cryptographic procedures:
  - Key generation
  - Authentication
  - Freshness
- PRNG (Pseudo RNG)



  - uses an algorithm with input, internal state and output which are "as unpredictable as possible"
  - knowledge of internal algorithm state and/or input might make the output predictable

RNG (Random Number Generator)

- TRNG (True RNG)
  - uses physical processes (frequency instability of an oscillator, semiconductor noise, particle-decay ...)
  - unpredictable output
  - influence of physical process must not affect the unpredictability



Amaury Neve, Dennis Flandre and Hean-Jacques Quisquater

RNG – Testing Random Numbers

Five basic tests (not more than failure to reject hypothesis):

1. Frequency test (monobit test)

$$X_1 = \frac{(n_0 - n_1)^2}{n} \qquad \chi^2 \mid DoF(1) \wedge n > 11$$

2. Serial test (two-bit test)

$$X_2 = \frac{4}{n-1}\left(n_{00}^2 + n_{01}^2 + n_{10}^2 + n_{11}^2\right) - \frac{2}{n}\left(n_0^2 + n_1^2\right) + 1 \qquad \chi^2 \mid DoF(2) \wedge n > 21$$

3. Poker test (*k* non-overlapping parts of length *m*)

$$X_3 = \frac{2^m}{k}\left(\sum_{i=1}^{2^m} n_i^2\right) - k \qquad \chi^2 \mid DoF(2^m - 1)$$

4. Runs test (*G*ap/*B*lock)

$$X_4 = \sum_{i=1}^{k} \frac{(B_i - e_i)^2}{e_i} + \sum_{i=1}^{k} \frac{(G_i - e_i)^2}{e_i} \qquad \chi^2 \mid DoF(2k - 2)$$

5. Autocorrelation test

$$X_5 = 2\left(A(d) - \frac{n-d}{2}\right) / \sqrt{n-d}, with: A(d) = \sum_{i=0}^{n-d-1} s_i \oplus s_{i+d}, and: 1 \le d \le n/2 \qquad N(0,1) \mid n - d \ge 10$$

Menezes, van Oorschot, Vanstone - Handbook of Applied Cryptography

RNG – Testing Random Numbers

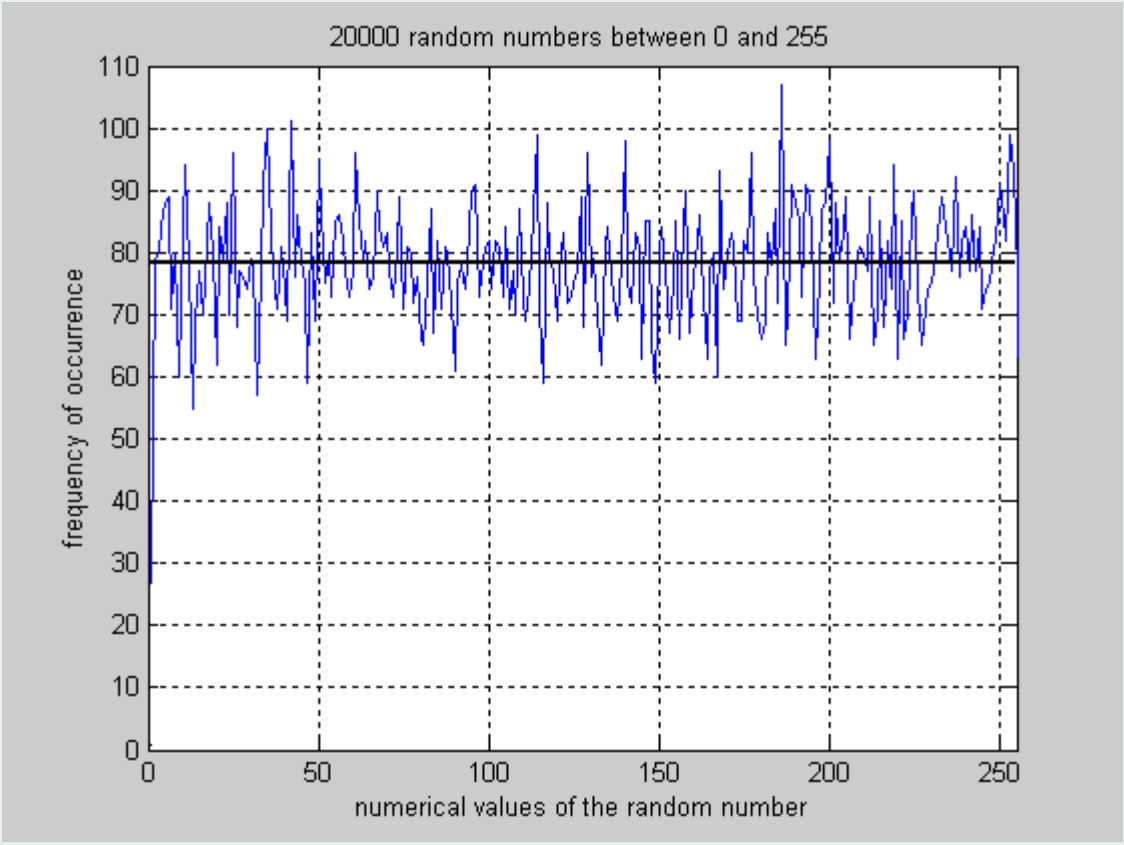
FIPS 140-1: Single bit stream of 20,000 consecutive bits must pass following tests:

1. Monobit Test – 9,654 < X < 10,346 (X = the number of ones)
2. Poker Test –
    1. Divide the 20,000 bit stream into 5,000 contiguous 4 bit segments. Count and store the number of occurrences of each of the 16 possible 4 bit values. Denote f(i) as the number of each 4 bit value i
    2. Evaluate the following:
       X = (16/5000) * [SUM of f(i)^2, for i = 0 to 15] – 5000
    3. The test is passed if 1.03 < X < 57.4.
3. Run Test –

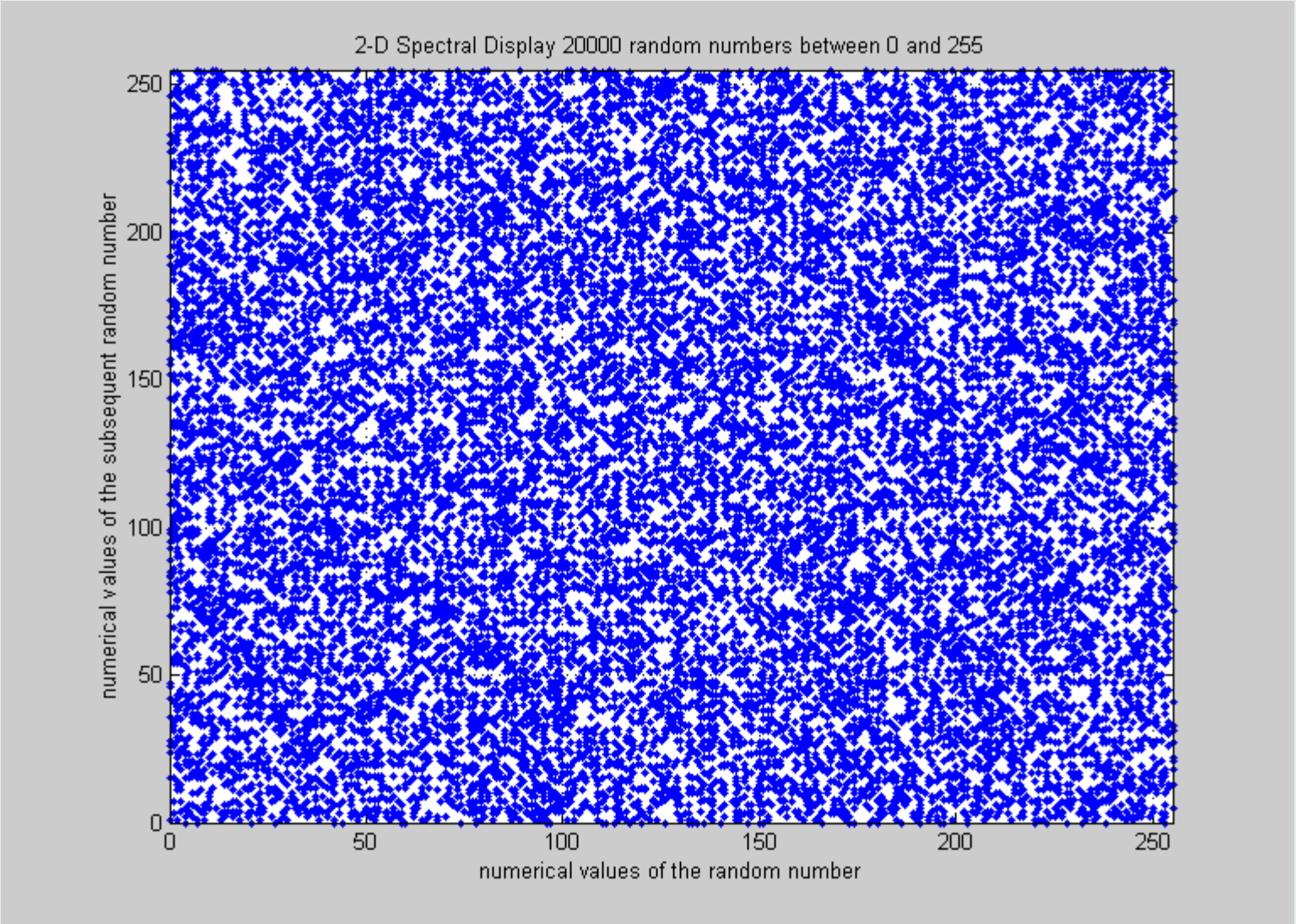
Length of Run	Required Interval
1	2,267 - 2,733
2	1,079 - 1,421
3	502 - 748
4	223 - 402
5	90 - 223
6+	90 - 223

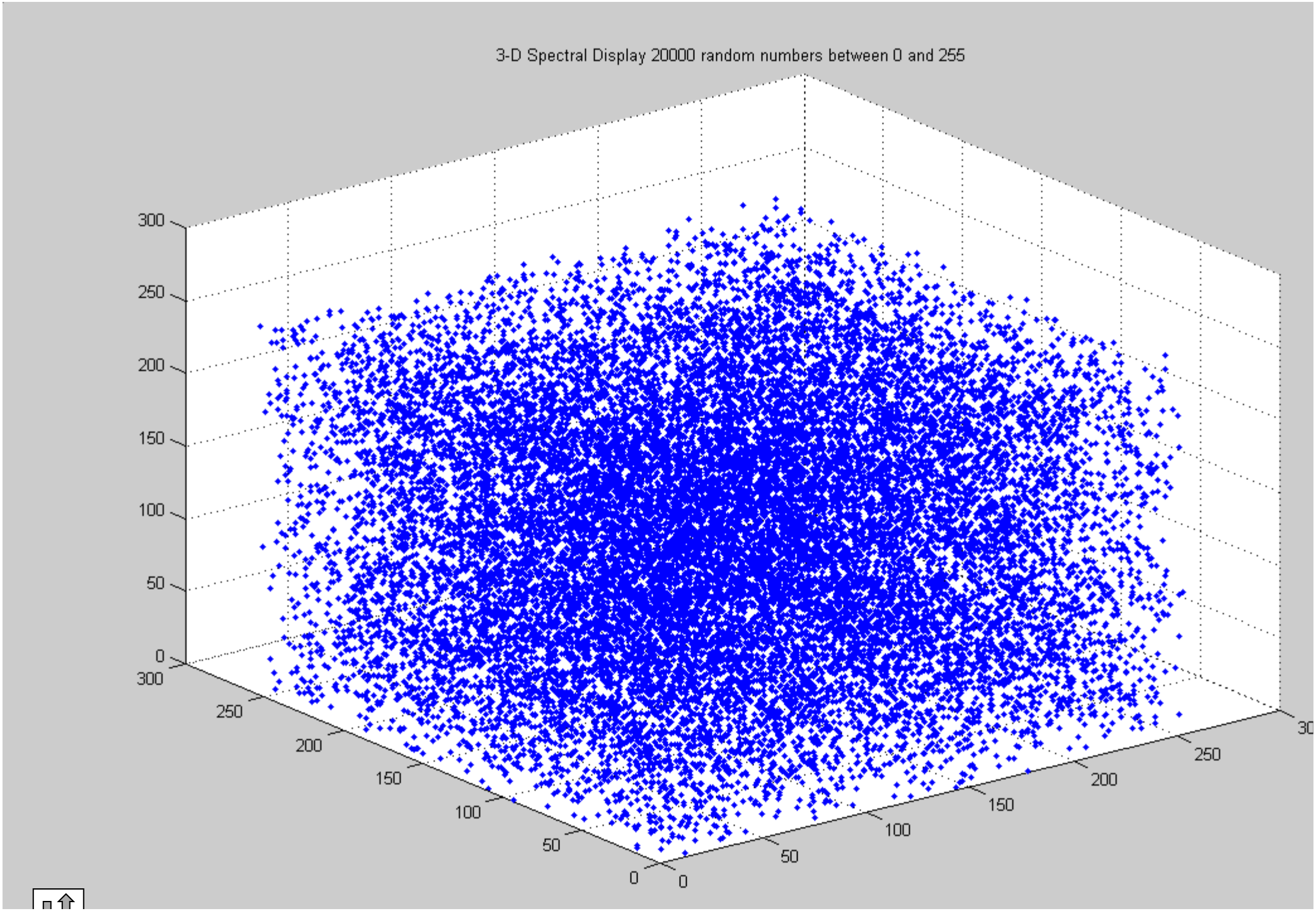
4. Long Run Test – No Runs > 33

RNG – Testing Random Numbers



RNG – Testing Random Numbers




3-D Spectral Display 20000 random numbers between 0 and 255
300
250
200
150
100
50
0
300
250
200
150
100
50
0
0
50
100
150
200
250
30

Card Accepting Devices (CAD/IFD)

Dumb - PC signal conversion and clock gen.
(Litronic 210, Dumbmouse, Dr Chip ...)

μProc - remote controlled microprocessor
(Gemplus GemPC 410 , Towitoko Chipdrive ...)

Keypad - PIN's stay local
(ORGA ICCR, ThuisChipper ...)

Embedded - Part of other electronic device
(POS-SAM, GSM-SIM ...)

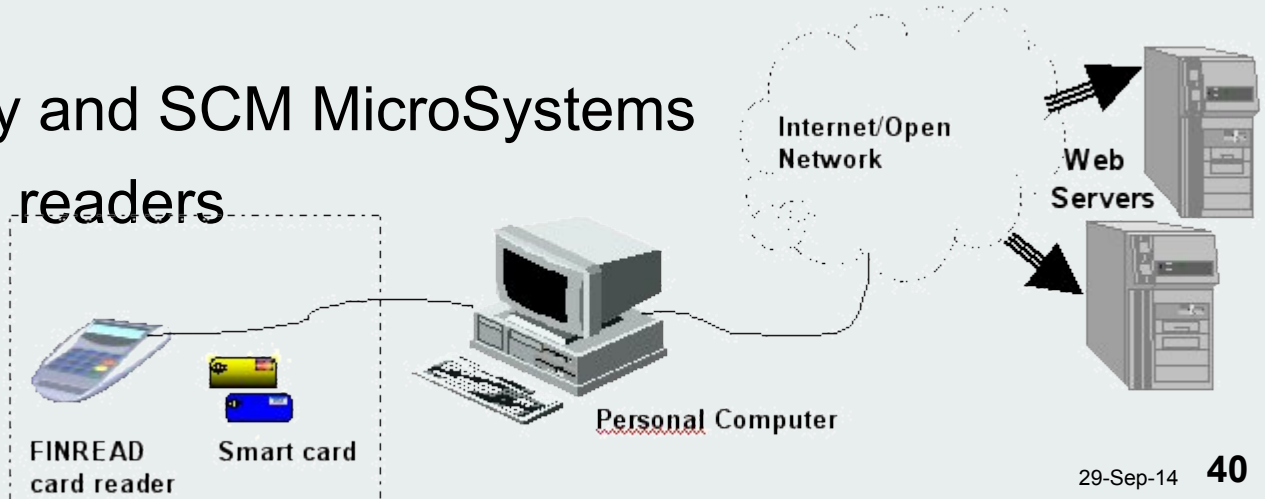
Wallets, Balance checkers …

USB-slots

Card Accepting Devices – FINREAD

www.finread.com

- Specifications for secure:
  - CAD architecture
  - downloading and management of CAD applets
  - data exchange between CAD and Smart Card
- Interoperability of secure transactions on open networks by:
  - common format for the downloading of applications,
  - common set of APIs between the card reader system software and the card reader downloaded applications
- Achieve a high security level through the definition of **security requirements**
- 2003: Ingenico, Omnikey and SCM MicroSystems first approved FINREAD readers



Card Operating Systems

Smart card OS tasks:

- life-cycle management
- instruction processing
- data management
- data transmission
- (hardware) error handling
- control of co-processor

Life-cycle management

Security not stronger than weakest link→
authorization depends on life cycle phase:

Test

- full access to all memories
- unique read-only serial number in EEPROM
- termination through writing EEPROM + (fuse blowing) + (hardware removal)

Life-cycle management

Completion

- key needed
- writing of EEPROM OS jump table + fixes
- file system initialization with root + serial number + transport key

Pre-personalization

- transport key needed
- placing of file structure
- writing of static data (PIN's, keys, application code)
- finishing by invalidation of transport key

Life-cycle management

Personalization

- writing of personal data
- files contain authorization attributes

Usage

- Security via file access control data

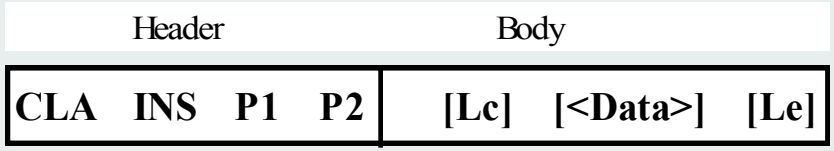
End

- All card functionality is made inaccessible in a an irreversible way

Real-life Card Life Cycle model: GlobalPlatform Card Specifications: OP_READY, INITIALIZED, SECURED, CARD_LOCKED, TERMINATED)

Instruction processing

- CAD master, smart card slave
- Instructions coded in APDU (OSI 7)
  - command APDU:

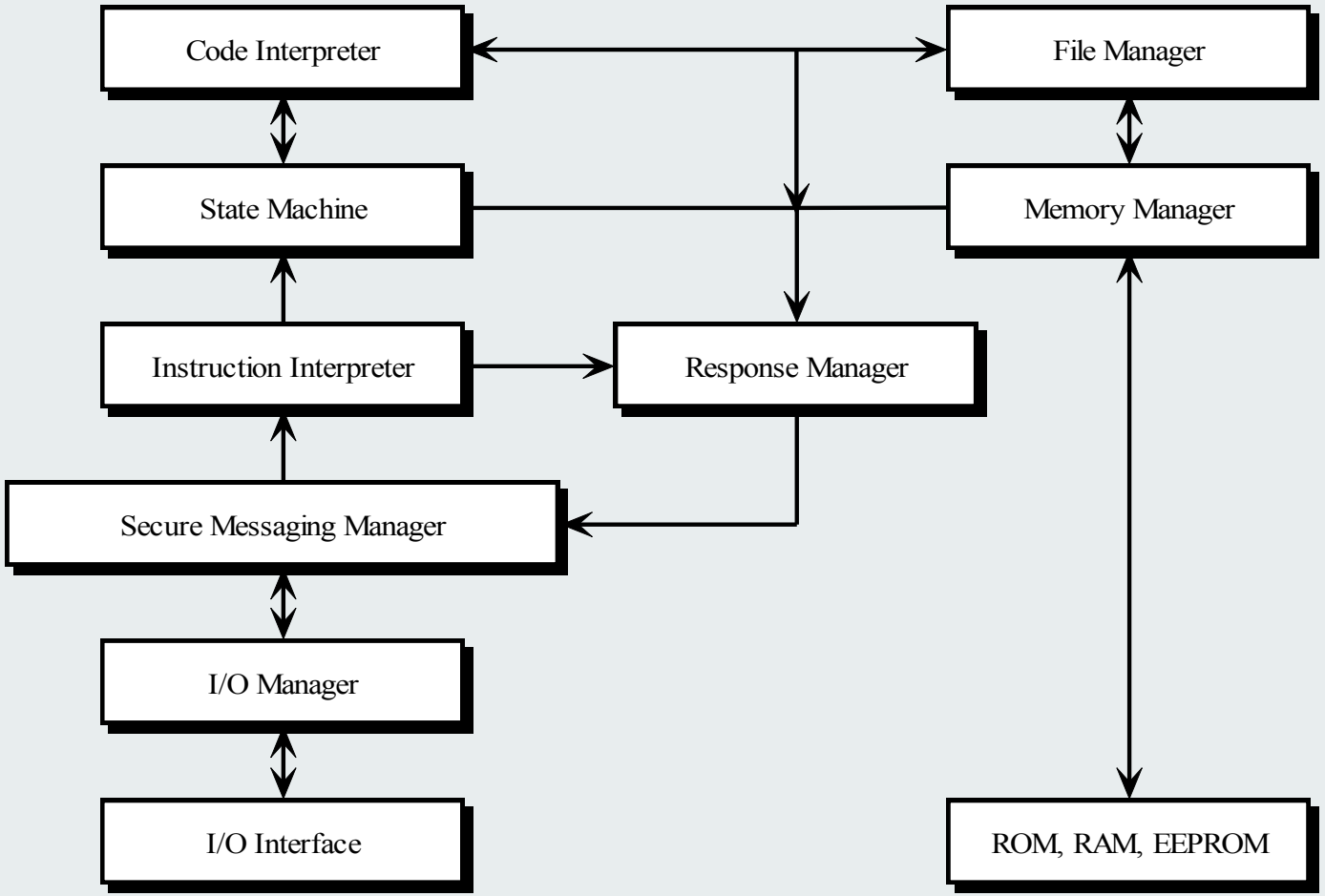


  - response APDU:

Instruction processing

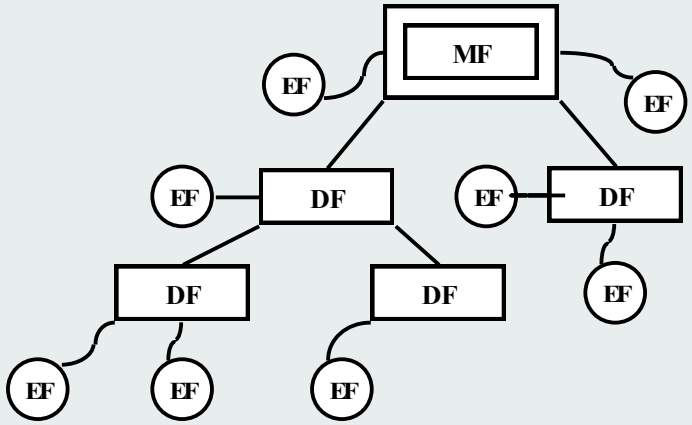
Instruction	Function	INS	Standard
EXTERNAL AUTHENTICATE	Card authenticates CAD	82	ISO 7816-4
INTERNAL AUTHENTICATE	CAD authenticates card	88	ISO 7816-4
ASK RANDOM	Receive random number from card	84	EN 726-3
GIVE RANDOM	Send random number to card	86	EN 726-3
SELECT FILE	Select card file	A4	ISO 7816-4
GET RESPONSE	Receive response from card	C0	ISO 7816-4
VERIFY CHV	Verify PIN	20	EN 726-3
CHANGE CHV	Change PIN	24	EN 726-3
DISABLE CHV	Switch-Off PIN	26	EN 726-3
ENABLE CHV	Switch-On PIN	28	EN 726-3
UNBLOCK CHV	Unblock PIN with PUK	2C	EN 726-3
READ BINARY	Read from transparant EF	B0	ISO 7816-4
READ BINARY STAMPED	Read from cryptographic secured transparant EF	B4	EN 726-3
READ RECORD	Read from lineair or cyclic EF	B2	ISO 7816-4
READ RECORD STAMPED	Read from cryptographic secured lineair or cyclic EF	B6	EN 726-3
SEEK	Search in lineair/cyclic EF	A2	EN 726-3
WRITE BINARY	Write to transparant EF (secure->non-secure)	D0	ISO 7816-4
UPDATE BINARY	Overwrite data in transparant EF (erase+write)	D6	ISO 7816-4
UPDATE RECORD	Write to record of lineair/cyclic EF	DC	ISO 7816-4
APPEND RECORD	Inser record to end of lineair fixed EF or overwrite less recent record of cyclic EF	E2	ISO 7816-4
INCREASE	Increase value of file counter	32	EN 726-3
INCREASE STAMPED	Increase value of file counter in cryptographic secure way	36	EN 726-3
DECREASE	Decrease value of file counter	30	EN 726-3
DEREASE STAMPED	Decrease value of file counter in cryptographic secure way	24	EN 726-3
ENVELOPE	Embed instruction with cryptographic protection	C2	ISO 7816-4
ERASE BINARY	Erase (part) of transparant EF	0E	ISO 7816-4
EXECUTE	Execute a file	AE	EN 726-3
CREATE FILE	Create a new file	E0	EN 726-3
DELETE FILE	Delete file	E4	EN 726-3
INVALIDATE	Reversibly block a file	04	EN 726-3
REHBILITATE	Unblock a file	44	EN 726-3

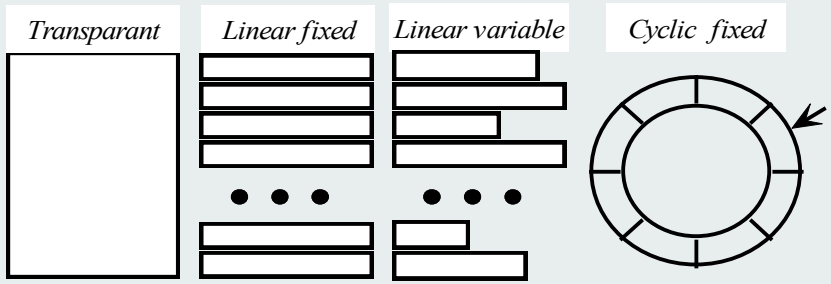
Instruction processing



Data Management

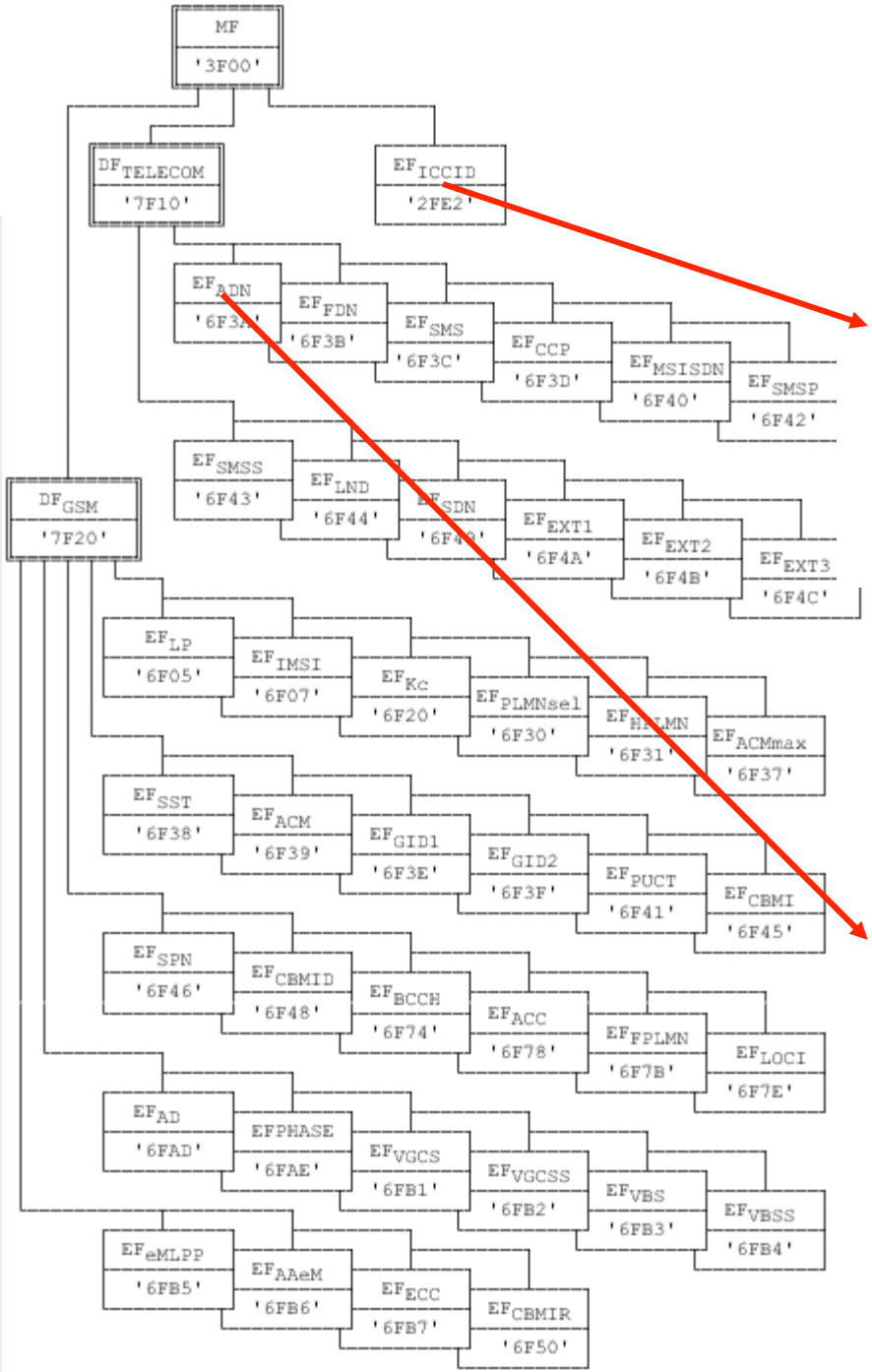
- File system:
  - Master File (root)
  - Directory Files
  - Elementary Files
    - transparant
    - linear fixed
    - linear variable
    - cyclic
    - purse
    - executable
    - …





Data Management

- File consists of:
  - Header (ID, type, size, access conditions, status)
  - Data



Identifier: '2FE2'	Structure: transparent	Mandatory
File size: 10 bytes	Update activity: low	

Access Conditions:

READ	ALWAYS
UPDATE	NEVER
INVALIDATE	ADM
REHABILITATE	ADM

Bytes	Description	M/O	Length
1 - 10	Identification number	M	10 bytes

Identifier: '6F3A'	Structure: linear fixed	Optional
Record length: X+14 bytes	Update activity: low	

Access Conditions:

READ	CHV1
UPDATE	CHV1
INVALIDATE	CHV2
REHABILITATE	CHV2

Bytes	Description	M/O	Length
1 to X	Alpha Identifier	O	X bytes
X+1	Length of BCD number/SSC contents	M	1 byte
X+2	TON and NPI	M	1 byte
X+3 to X+12	Dialling Number/SSC String	M	10 bytes
X+13	Capability/Configuration Identifier	M	1 byte
X+14	Extension1 Record Identifier	M	1 byte

Data Management

- ASN.1 BER - TLV coding
  - Abstract Syntax Notation One
  - Basic Encoding Rules
  - Tag Length Value
  - ISO 7816-6 annex contains general TLV definitions

T	L	V	T	L	V	T	L	V	T	L	V	T	L	V	T	L	V	T	L	V
'85'	'06'	"Ronald"	'86'	'07'	"van der"	'87'	'06'	"Knijff"	'88'	'10'	"Volmerlaan"	'89'	'02'	"17"	'90'	'06'	"3024AE"	'91'	'11'	"Rijswijk ZH"
'87'	'06'	"Knijff"	'86'	'07'	"van der"	'85'	'06'	"Ronald"	'90'	'06'	"3024AE"	'91'	'11'	"Rijswijk ZH"	'88'	'10'	"Volmerlaan"	'89'	'02'	"17"

Advantage:

- *self-describing*
- extendable/backwards compatible

Disadvantage:

- T and L need storage space

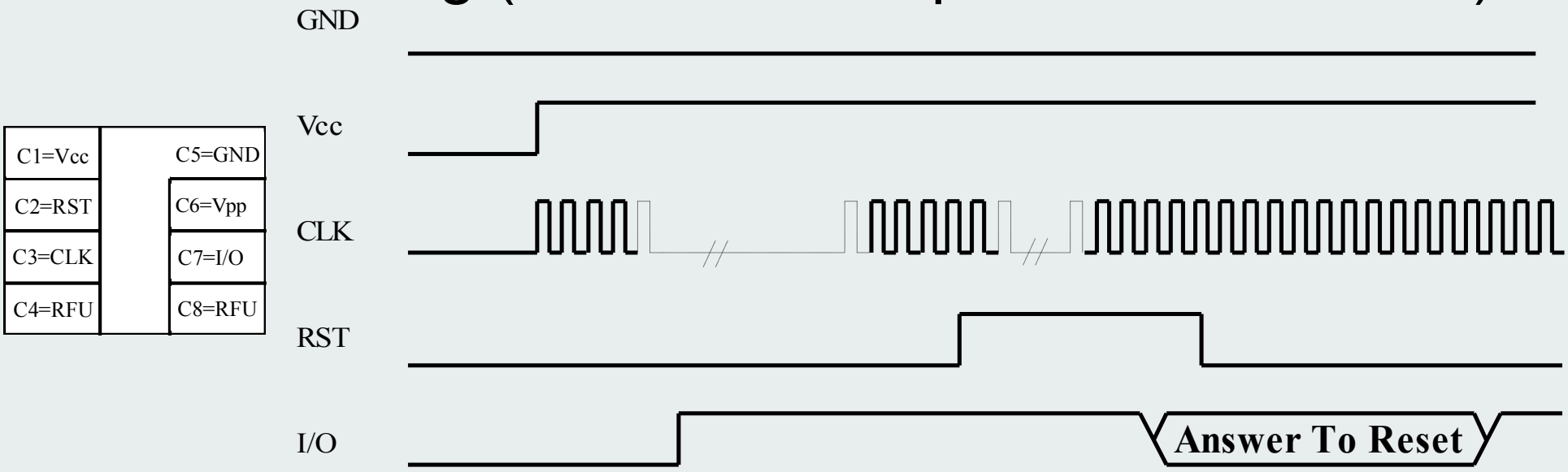
ASN.1 BER TLV coding

Table 8 - IDOs in alphabetic order

Tag	Name of Data Element	Description & ISO Reference	Length / Format	May be found within template
5F42	Address	Address of an individual	variable	65
5F25	Application effective date	Date from which the application can be used, under the responsibility of the Application Provider	n 6 YYMMDD	6E
5F24	Application expiration date	Date after which an application expires	n 6 YYMMDD	6E
4F	Application identifier	A DE which identifies an application in a card (see ISO/IEC 7816-5)	variable	61/6E
5F44	Application image	Image data for an icon or logo associated with an application (see ISO/IEC 10918-1)	variable	6D
6D	Application image template	Template containing at least an application image (see ISO/IEC 10918-1)	variable	6E
50	Application label	A DE for use at the man machine interface (see ISO/IEC 7816-5)	variable	61/6E
47	Card capabilities	As defined in ISO/IEC 7816-4	variable	66
5F26	Card effective date	Date, from which the card can be used, under the responsibility of the Card Issuer	n 6 YYMMDD	66
59	Card expiration date	Date after which the card expires	n 4 YYMM	66
45	Card issuer's data	As defined in ISO/IEC 7816-4	variable	66
5F34	Card sequence number	A number distinguishing between separate cards with the same Primary Account Number	n 2	66
43	Card services data	As defined in ISO/IEC 7816-4	1 byte	-
5F2E	Cardholder biometric data	Biometric data relating to the cardholder	variable	65
7F21	Cardholder certificate	A constructed DO containing the public key of the cardholder, further information, signature of certification authority	variable	65
5F43	Cardholder handwritten signature image	An image of the cardholder's signature (see ISO/IEC 11544)	variable	6C
6C	Cardholder image template	Cardholder related images stored within the ICC	variable	65
5F20	Cardholder name	To indicate the name of the cardholder (see ISO/IEC 7813)	n 2..26	65
5F2C	Cardholder nationality	To indicate the nationality of the cardholder. See ISO 3166 for coding	n 3	65
5F40	Cardholder portrait image	Encoded image data, used for the cardholder portrait image	n 1	6C
5F49	Cardholder public key	A DE containing the cardholder's public key for digital signature functionality using asymetric mechanisms	variable	65
5F48	Cardholder secret key	A DE containing the cardholder's secret key for digital signature functionality using asymetric mechanisms	variable	65
79	Coexistent Tag Allocation Authority	Used to identify a coexistent tag allocation scheme and the authority responsible for the scheme	variable	-
52	Command to perform	Command APDU (see ISO/IEC 7816-4)	variable	61
78	Compatible Tag Allocation Authority	Used to identify a compatible tag allocation scheme and the authority responsible for the scheme	variable	-
41	Country Authority	See 4.4.4	variable	-
5F28	Country code	Code for the representation of Name of Country (see ISO 3166)	n 3	66
5F2A	Currency code	Code for the representation of currencies and funds (see ISO 4217). Length will be 2 bytes if numeric format; 3 bytes if alphabetic format	a 3 or n 3	6E
5F36	Currency exponent	Codes a number by which an amount of the currency indicated in the card shall be multiplied (see ISO 4217)	n 1	6E
5F2B	Date of birth	Date of birth of related individual	n 8 YYYYMMDD	65
53	Discretionary data	Provides a standard way to denote a DE not defined in ISO/IEC 7816. Its use within the file control information and the application template is defined in parts 4 and 5 of ISO/IEC 7816. Clause 5 of this part ISO/IEC 7816 covers all the cases where this IDO can be retrieved	variable	all templates defined in Annex A

Data transmission

- Serial asynchronous master slave
- CLK 3.5712 MHz. or 4.9152 MHz.
- Booting (transmission speed I/O=CLK/372):



- After PTS communication through via negotiated protocol

Data Transmission T=0 protocol

- Byte oriented
- TPDU (Transmission Protocol Data Unit) ≈ APDU
  - CAD transmits CLA, INS, P1, P2, P3
  - Card transmits procedure byte ACK
  - Following communication depends on Command
  - Communications end with status bytes SW1, SW2
- Transmission errors detected via parity bit and corrected via second time transmission
- Poor separation application and data link layer

Data Transmission T=1 protocol

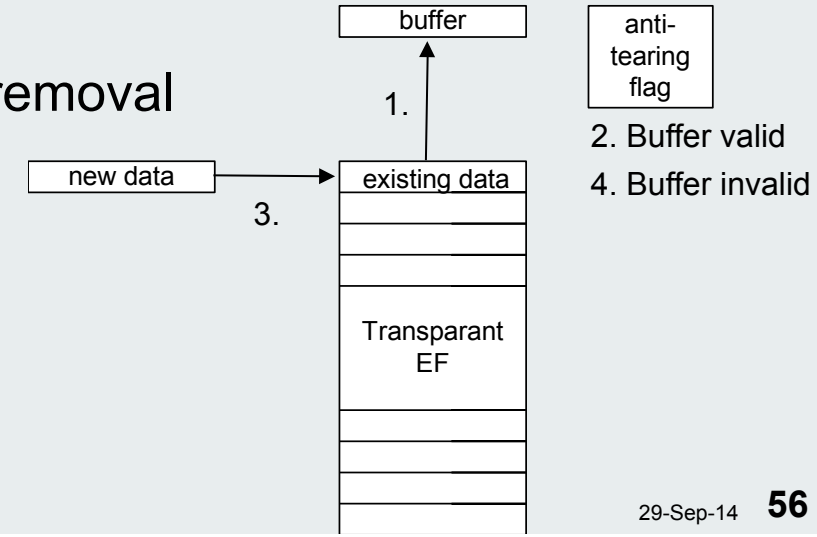
- Block oriented

Prologue			Information	Epilogue
NAD	PCB	LEN	APDU	EDC
1 Byte	1 Byte	1Byte	0 - 254 Bytes	1-2 Bytes

- Block types:
  - I - application data
  - R - receive confirmation
  - S - protocol control data
- Transmission errors detected with EDC: LRC (XOR byte) or CRC ($x^{16}+x^{12}+x^5+1$), correction via S-block + PCB

Hardware Error Handling

- EEPROM most error prone component → Write=Erase {1} + Program {0,1}
- EEPROM secure state: minimum energy state (0)
- Prevention and Control mechanisms
  - Map one logical EEPROM address to more physical addresses and alternate write actions
  - Implement one logical EEPROM address with several physical addresses and use *majority voting*
  - EDC's and ECC's
  - *anti tearing* against sudden card removal
    - buffer + anti-tearing flag
    - cyclic files



Developments - Architectures

- Major smart card chip suppliers:
  - Infineon, STMicroelectronics, Hitachi, Philips, Atmel, Samsung, NEC
- smaller structures (<0.18 μm) →more functionality on 25 mm^2
  - larger memories
  - co-processors (3DES, AES, RSA)
  - security (sensors, TRNG, MMU …)

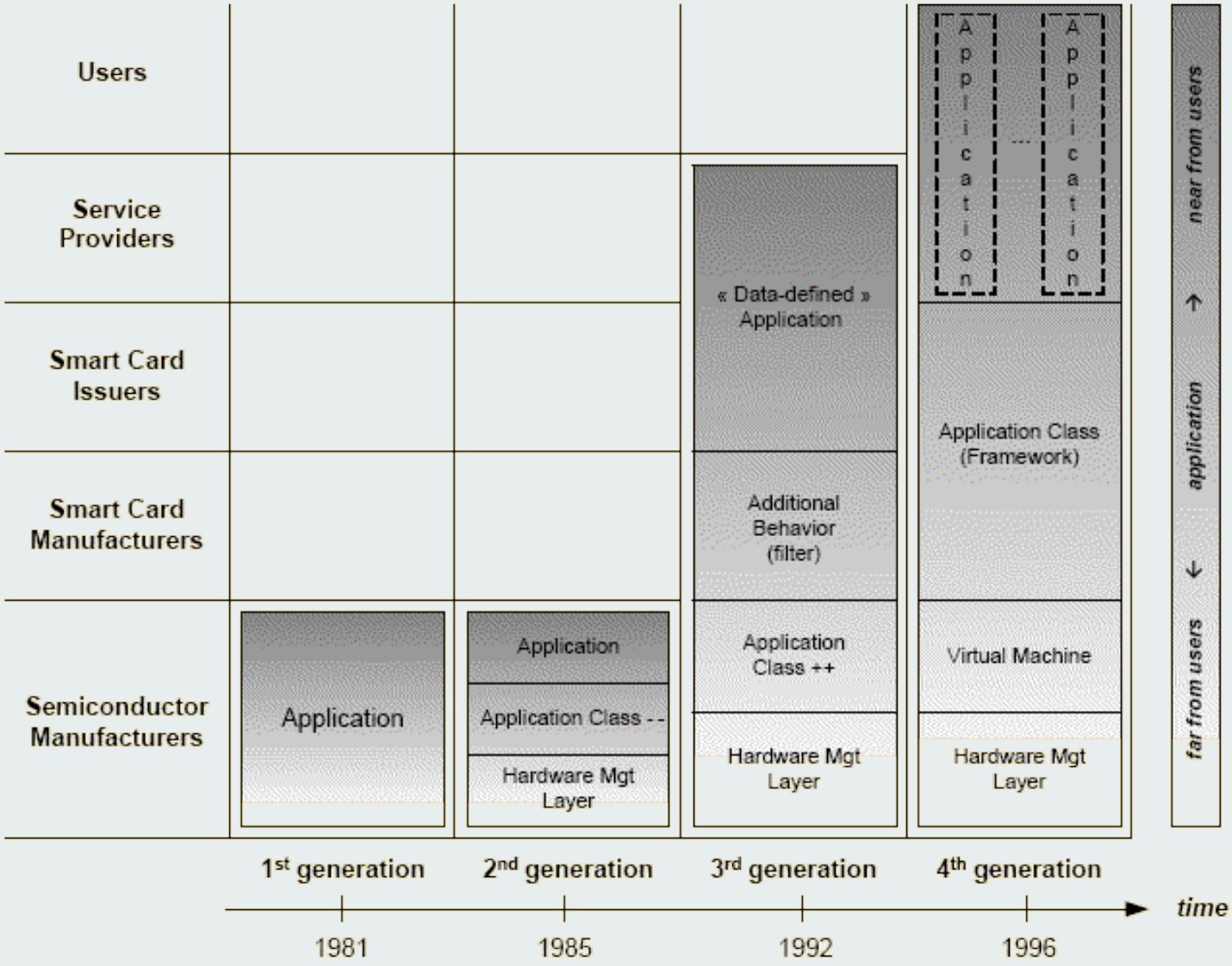
Developments - Architectures

- 16- and 32 bits architectures
- RISC (ARM & MIPS)
- Virtual Machine Optimizations
- Dual interface cards (e.g. MIFARE Pro)
  - contact interface
  - contactless interface
  - both connected to the same chip/OS/memory

Developments - Card OS'es

- Major smart card card/OS suppliers:
  - Axalto (SchlumbergerSema=Schlumberger + Bull CP8)
  - Gemplus
  - Giesecke & Devrient
  - Oberthur
  - Orga
  - Sagem
- Older COS generations:
  - too much and all different
  - CAD applications depend on card OS
  - low-level *on-card* application development
  - limited expansion possibilities during usage

www.gemplus.com/smart/r_d/publications/pdf/DGGJ03os.pdf



Developments - Card OS'es

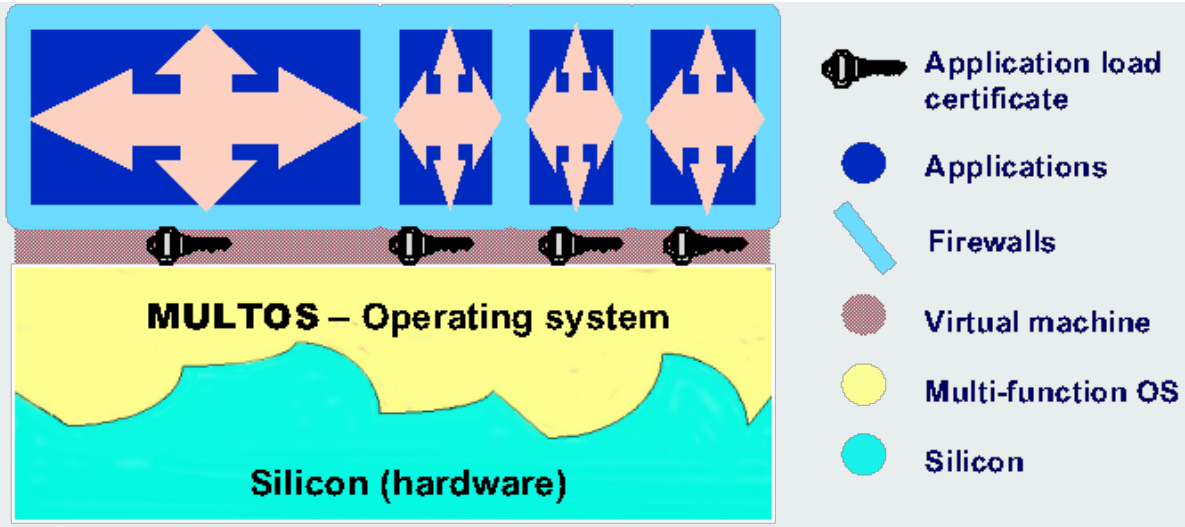
Solutions:

- Join best aspects into one standard Card OS API
- New systems
  - Java Card
  - MULTOS
  - BasicCard
  - Windows for Smart Cards/Smartcard.NET

Virtual Machine principle:

- abstract machine on top of OS
- interpretation of hardware independent byte code

Developments - MULTOS

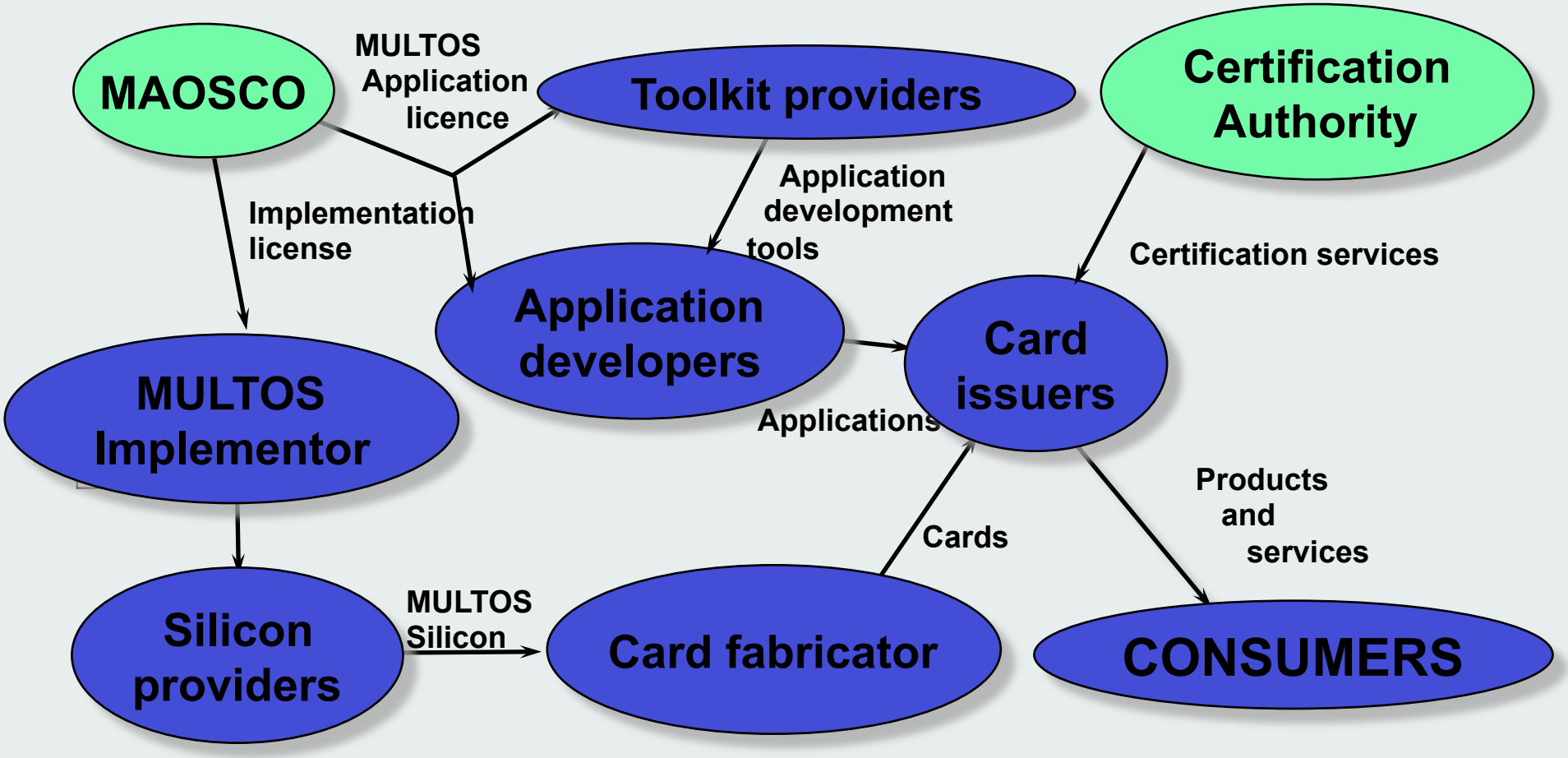


- Low level kernel with VM on top
- VM interprets MEL (MULTOS Executable Language)
- Multi-Functional
- designed for ITSEC E6-High

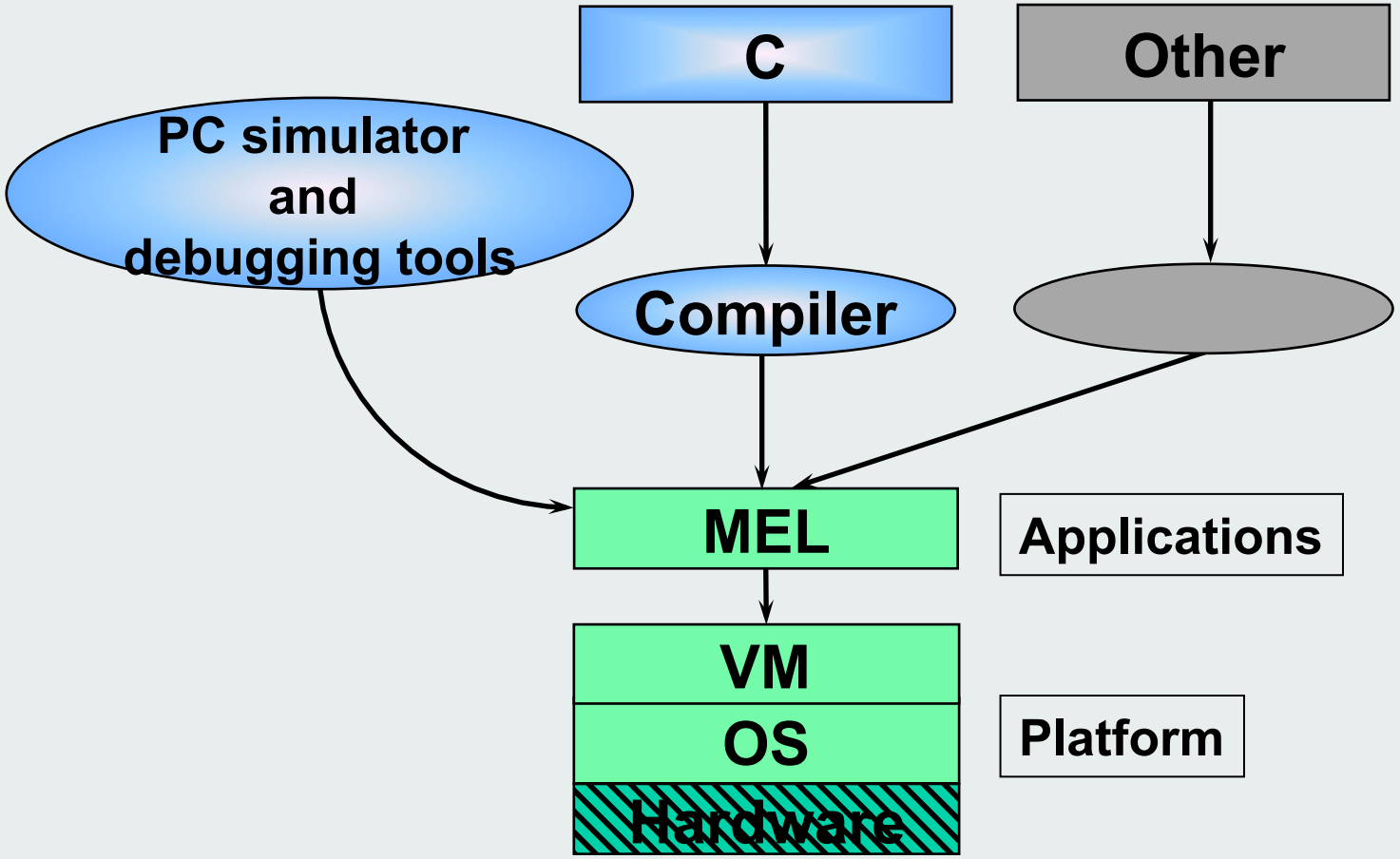
Developments - MULTOS

- Licenses needed from MAOSCO
- Applications certified by CA
- Crypted application loading with certificate
- Application controls interaction with other applications
- Implementations on Hitachi 3112/3113, Infineon SLE66CX160S, Motorola ...
- Used for MONDEX purse and EMV

Developments - MULTOS licensees

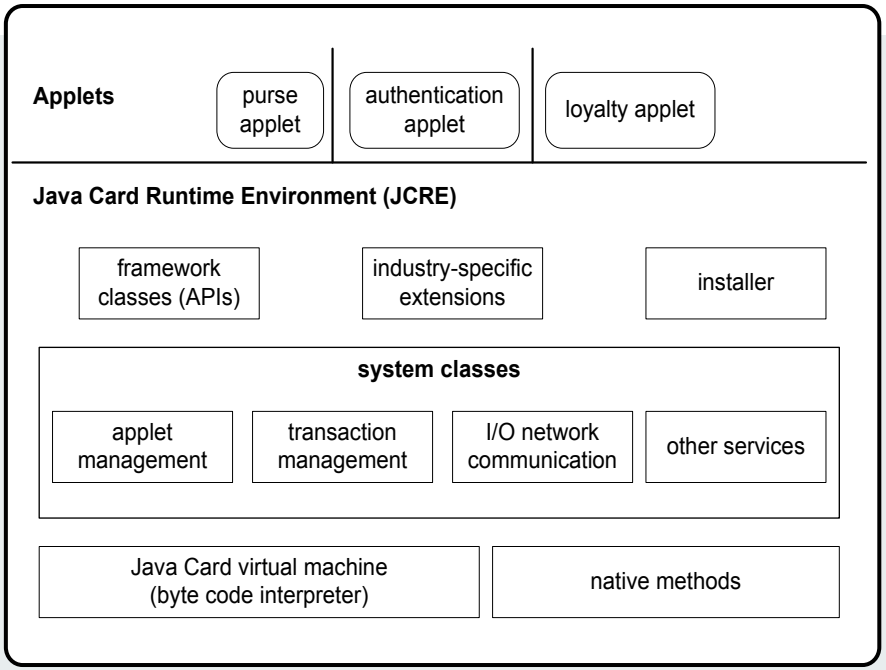


Developments - MULTOS



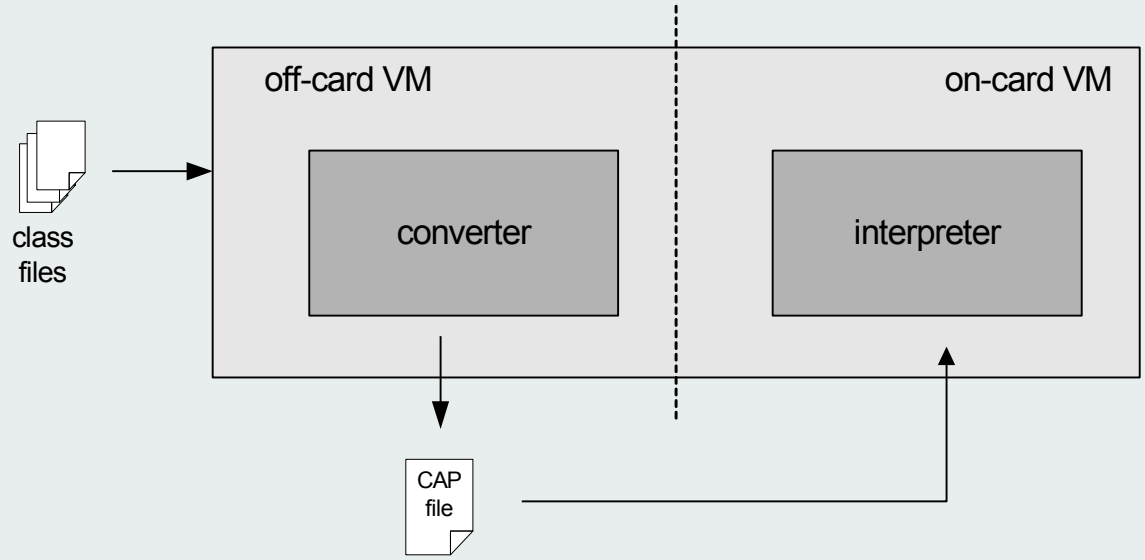
Developments - Java Card

- Java without:
  - Dynamic class loading
  - Security manager
  - Threads and synchronization
  - Object cloning
  - Finalization
  - Large primitive data types (32- en 64 bit, Unicode)
- Minimal architecture:
  - 8-bit core
  - 16 KB ROM, 256 Bytes RAM, 8 KB EEPROM



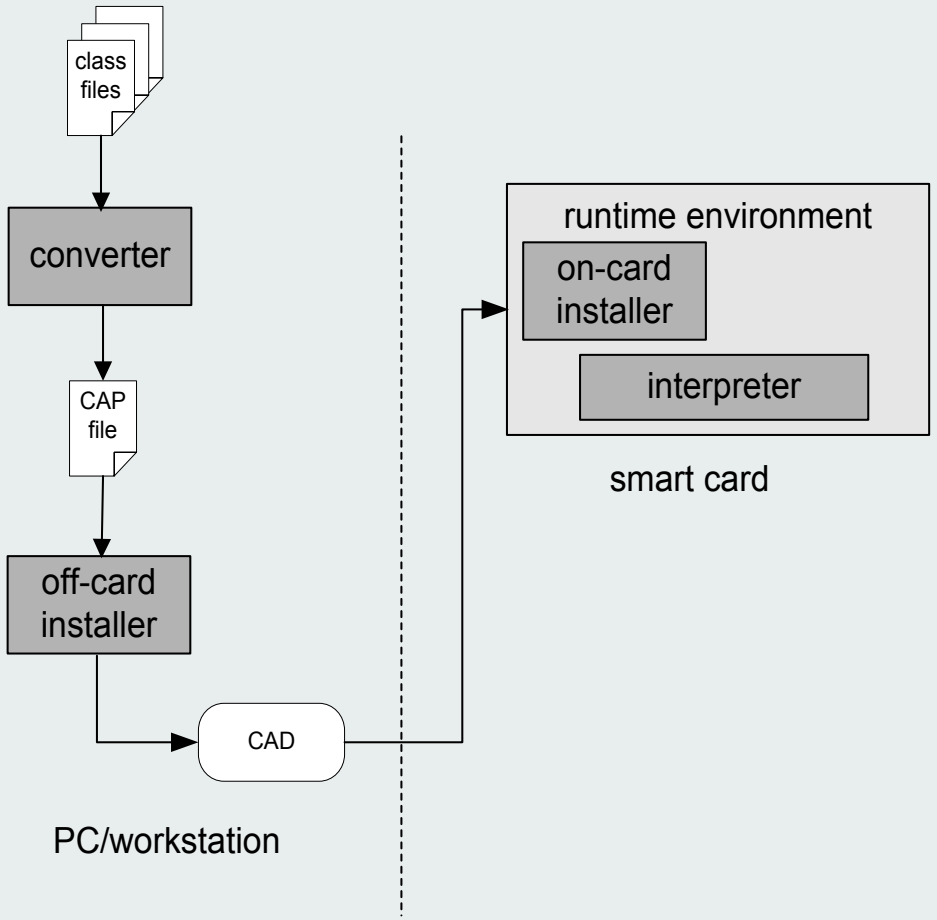
Developments - Java Card

- JCRE active until *card-end-of life (no signals =long clock cycle)*
- Applet active from registration until removal
- *transient* RAM objects for speed
- Each Applet unique AID (ISO 7816-5)
- JCRE controls security policy (applet firewalls/context switching)
- Objects owned by creator Applet
- Applet can share attributes and methods with specific applets or all applets



Developments - Java Card

- Applet development with common Java Tools
- Compiler generates *class file*
- Java Card converter checks and optimizes byte code
- Converter/Loader for each Java Card implementation
- Global Platform standard for secure applet addition, management and removal
- Security definable with *Java Card Protection Profile* *java.sun.com/products/javacard/JCSPPC-1.0b.pdf*
- A lot of implementations available (especially for GSM SIM development)



http://securingjava.com/chapter-eight/

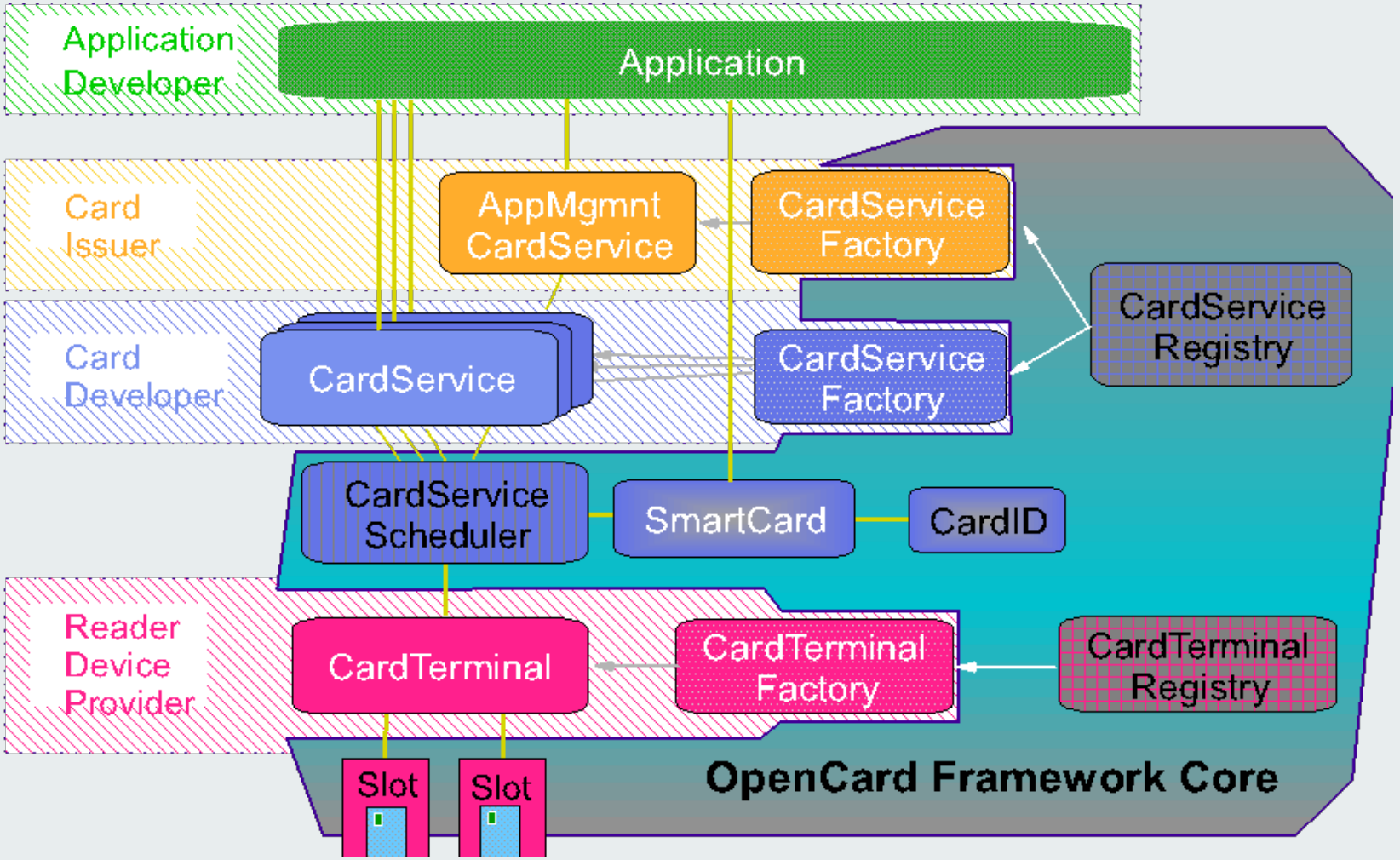
http://java.sun.com/products/javacard/JavaCardSecurityWhitePaper.pdf

http://java.sun.com/docs/books/javacard/

Developments - Open Card Framework (OCF)

- Standard framework for inter-operable smart cards solutions across different hardware and software platforms
- Two major function categories:
  - Application and Service Developers:
  - Card and Terminal Providers:
- Architecture + set of API's
- Java based
- Runs on any Java enabled platform

Developments - Open Card Framework (OCF)

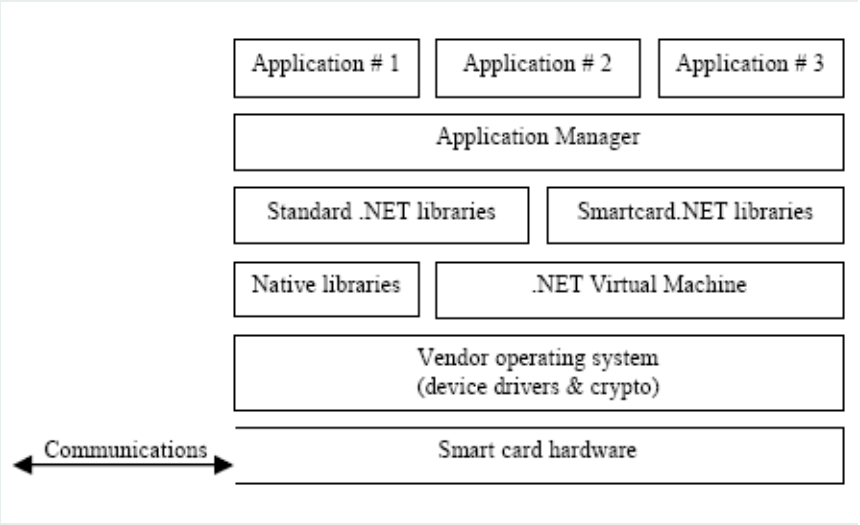


www.opencard.org
www.gemplus.com/techno/opencard/

Developments – Windows for Smart Cards

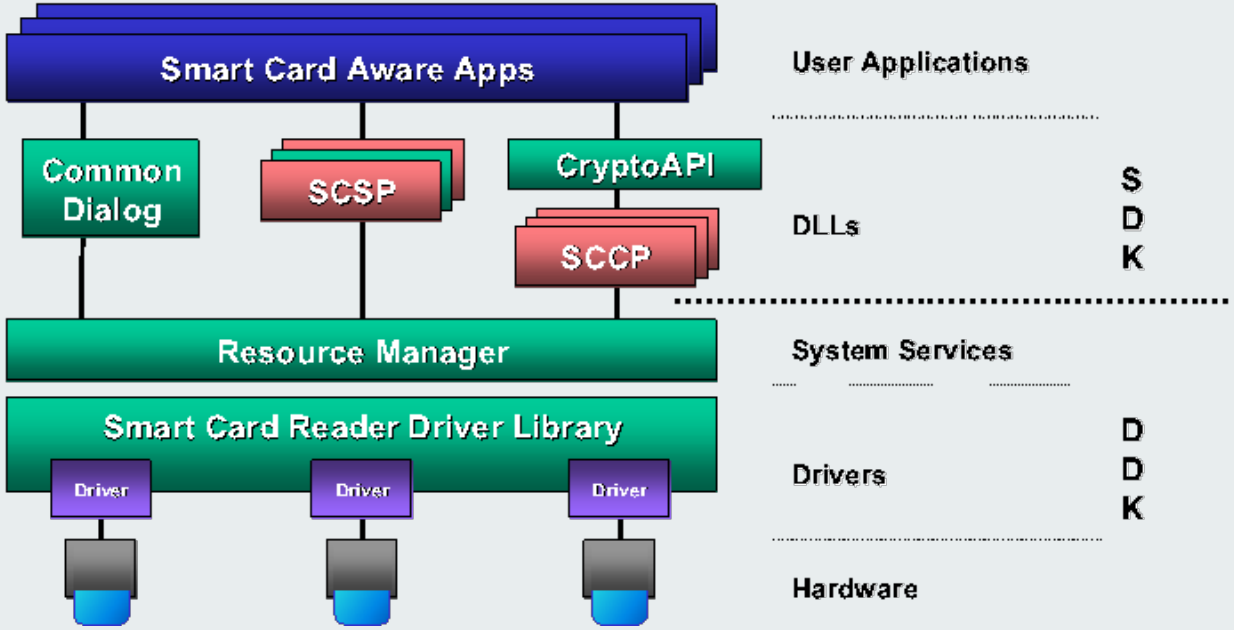
Microsoft Card OS:

- 8-bits, 8 KB ROM
- multi-application
- *applications loading in user phase*
- Windows developer environments
  - Visual Basic
  - Developer Studio
- May 2001 end of WfSC:
  - Current versions: 1.1 GSM & 2.0 Banking
  - Developer team discontinued
  - Source licensed and/or further developed by:
    - SCI (SCI-OS, s-Choice)
    - Sagem (W-OS)
- November 2002: *Hive Minded* announces *Smartcard.NET*
  - ECMA/ISO standard .NET platform for smart cards
  - Uses Microsoft Visual Studio .NET development tools
  - End 2004: licensed to Axalto to be used by Microsoft employees



Developments - PC/SC

Interoperability Specification for ICC's and Personal Computer Systems:

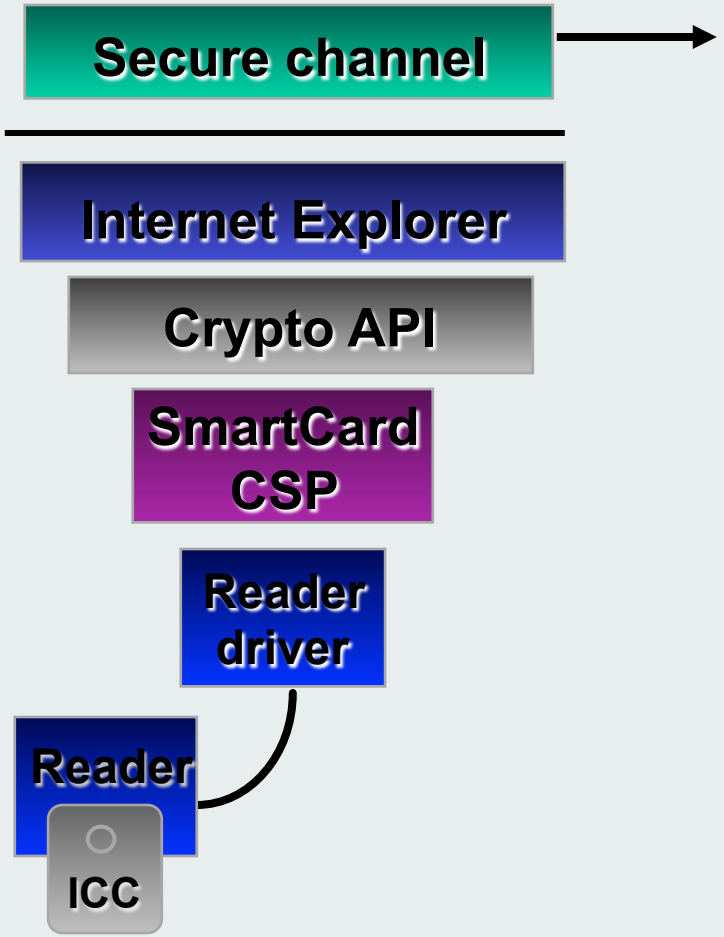


Developments - PC/SC

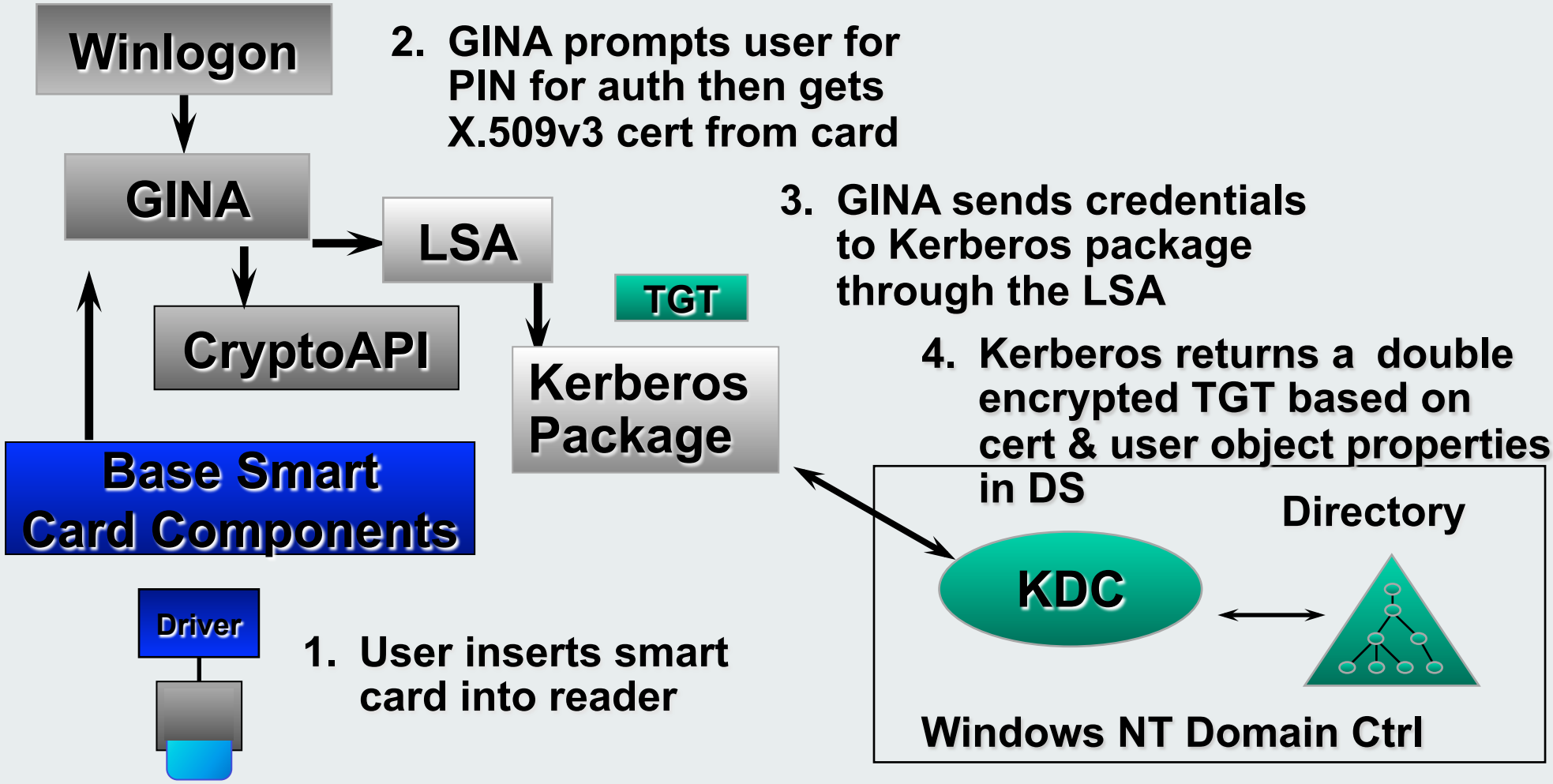
- Separation between applications, CAD's and smart cards
- SCSP on card , service or domain basis
- Availability:
  - Windows 95 and NT 4.0 as add-on
  - Windows 98 on CD
  - W2K/XP standard
  - Unix M.U.S.C.L.E (www.linuxnet.com)
- Used for W2K/XP with Crypto API, BIO API and Kerberos for GINA

W2K/XP PC/SC client authentication (SSL/TLS)

- Secure Channel between Internet Explorer and Internet Information Server
- Keys and certificates managed by CryptoAPI, stored in smart card
- Smart Card CSP retrieves certificate and protocol signature from card



W2K/XP PC/SC interactive logon



Final report smartcard PKI experiment TU/e:
http://www.gigaport.nl/netwerk/access/ta/pki/tue/eindrapportage-tue.pdf

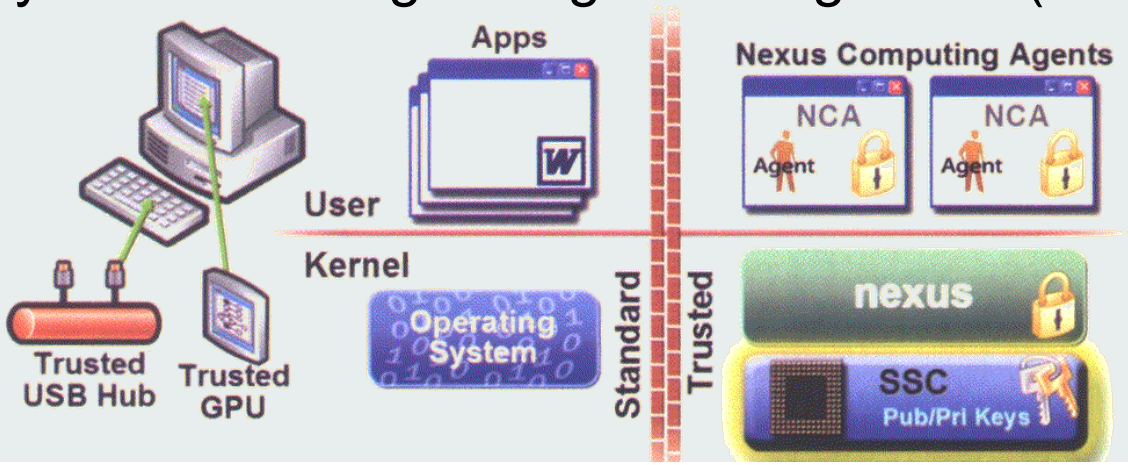
W2K/XP PC/SC remote logon

- RAS (Remote Access Service) supports EAP (Extensible Authentication Protocol)
- EAP provides standard mechanism for PPP additions
- Built-in EAP smart card module for strong authentication of remote users towards:
  1. RAS server
  2. Domain (EAP/TLS like *client authentication)*
- Also used for VPN connection authentication and data encryption (PPTP and L2TP/IPSec)

Trusted Computing Group

www.trustedcomputinggroup.org

- Mission: Standardize, license and promote new hard- and software to protect PC's, PDA's, mobile phones and other devices from hackers, viruses and other security threats
- Promoters: AMD, Hewlett-Packard, IBM, Intel, Microsoft and Sony
- Some of the TCG standards are incorporated in Microsoft's *Next-Generation Secure Computing Base (NGSCB):*
  - new security technology for MS Windows OS
  - SSC (Security Support Component) is a TCG TPM (Trusted Platform Module) implementation
- Not designed for but very usable for Digital Right Management (DRM)



Card Technology August 2003
www.cardtechnology.com

Developments - Standards

- **ISO** International Standards Organization, 7816 (http://www.iso.ch/)
- **ETSI** European Telecommunications Standards Institute, SIM (http://www.etsi.org/getastandard/home.htm)
- **EMV** Europay, Mastercard, and VISA, debit and credit cards (http://www.emvco.com/)
- **Global Platform** standards for smart card infrastructure (www.globalplatform.org)
- **PKCS #11 & #15** RSA Public-Key Cryptography Standards (http://www.rsasecurity.com/rsalabs/pkcs/)

Physical Attacks

- Visual Inspection
- Micro Probing
- Electron Beam
- Focused Ion Beam
- Scanning
- Signal Distortion

Physical Attacks - Visual Inspection

- Chip module protected on front with protective coating (epoxy)
- Removable with chemicals
- OS ROM code in general not visible from the top layer → removing top layers with *wet/dry etching*
- ROM content can be re-constructed with image processing
- Interesting technique: *backside inspection*
- optical microscope: magnification ≈ 1500; SEM ≈ 100000

The Chip

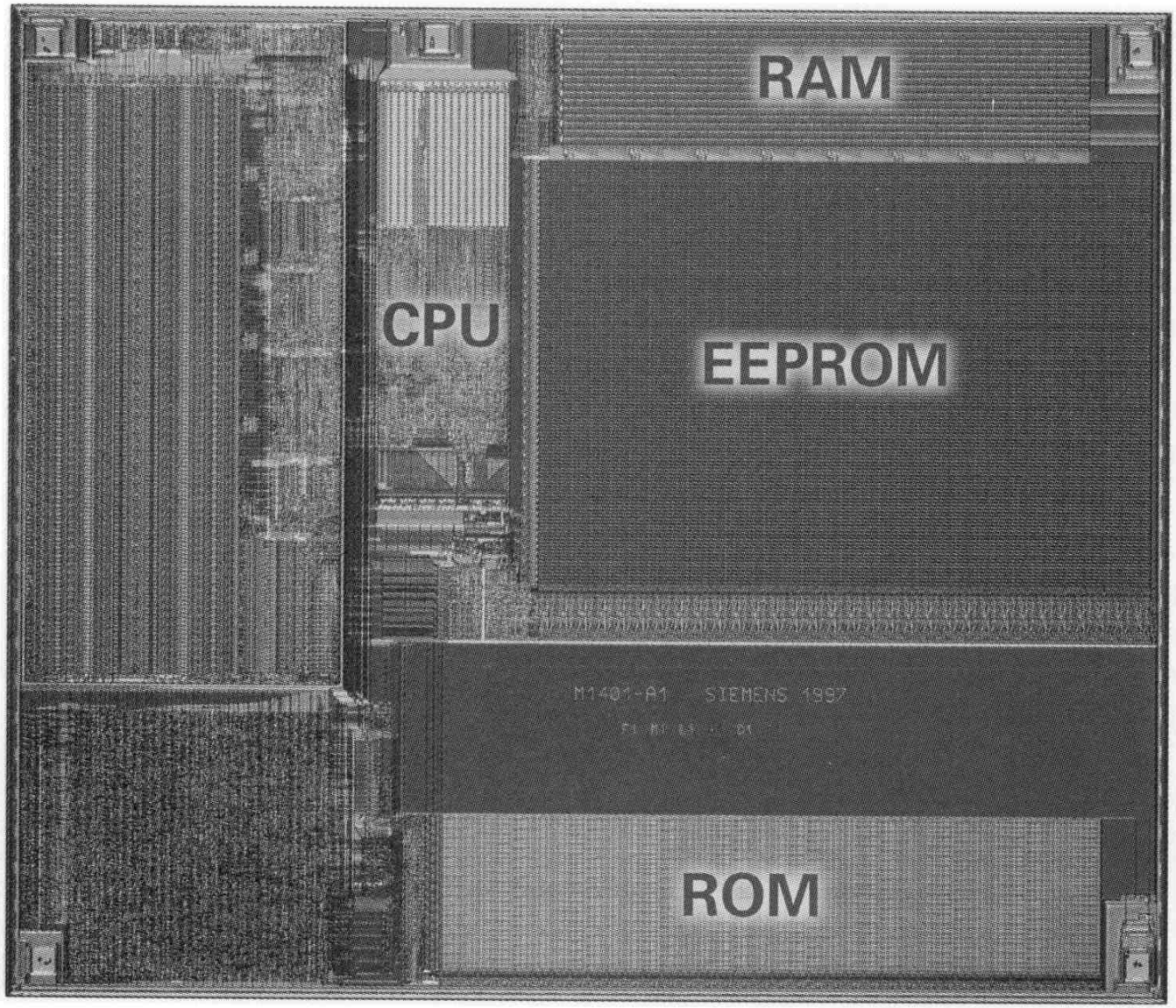
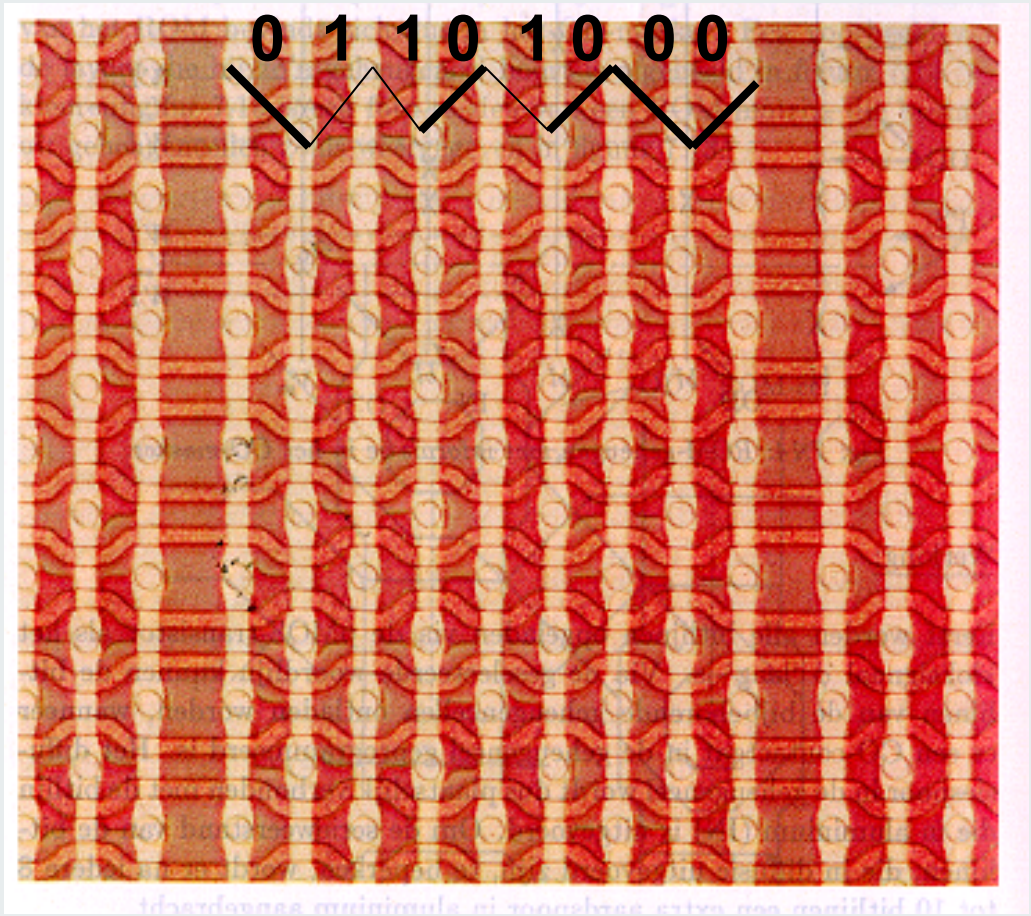


Figure 3.38 Photo of an SLE 66CX160S Smart Card microcontroller with an area of 21 mm^2. This chip was made using 0.6-µm technology and has 32 kB of ROM, a 16-kB EEPROM and 1280 bytes of RAM. The two unlabeled regions on the left-hand side of the chip are the numeric coprocessor and the peripheral elements (timer, random-number generator and CRC arithmetic processor). The five bonding pads for the electrical connections to the module contacts can be clearly seen in the photo. (Source: Infineon)

ROM mask

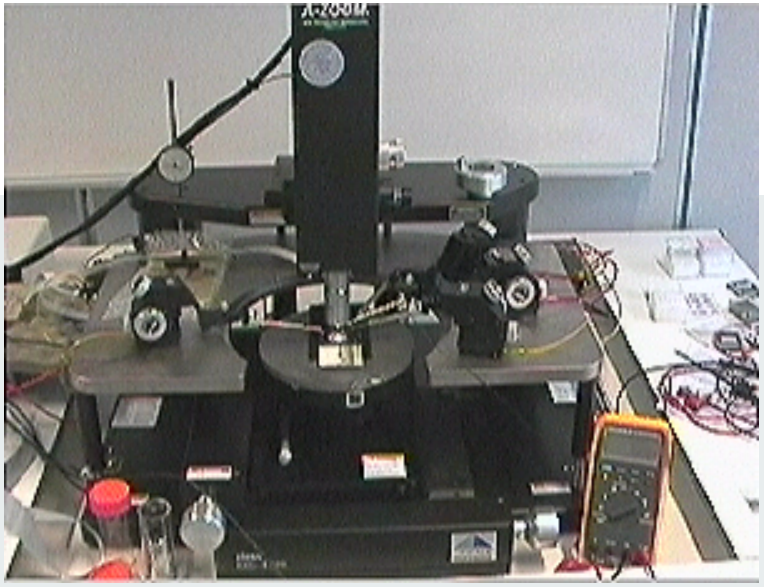


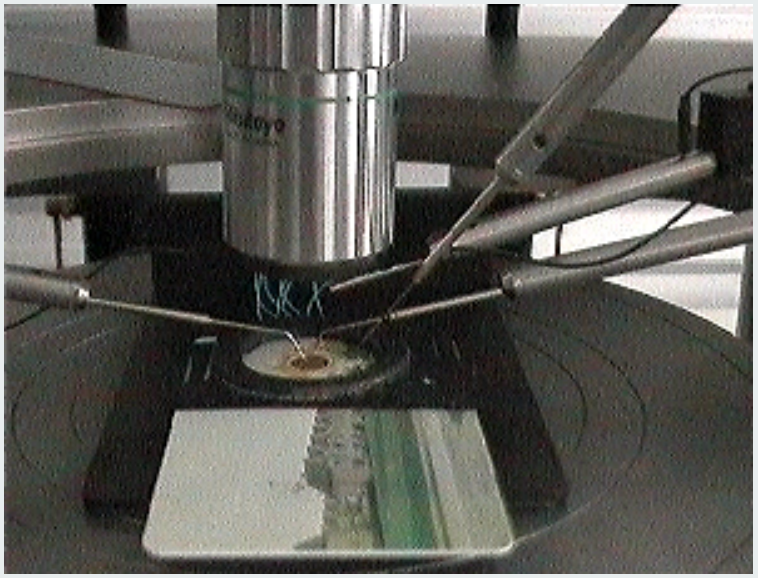
Physical Attacks - Micro Probing

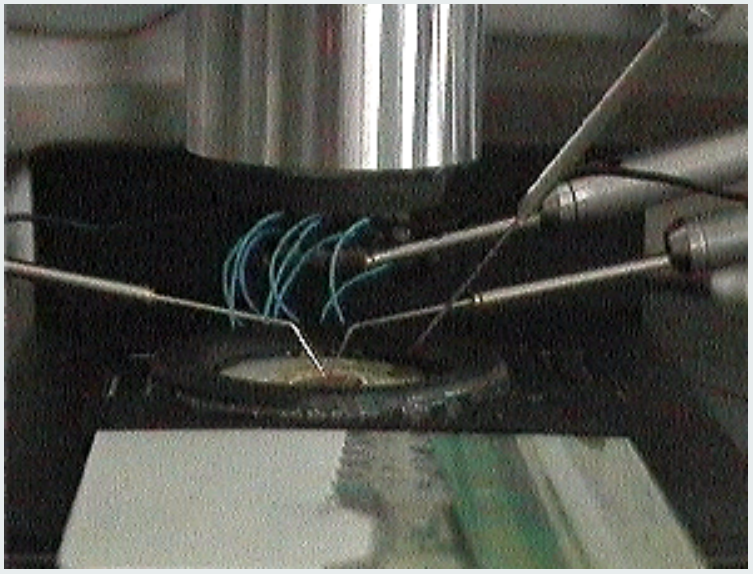
Needles ≈ 1 μm placed on internal chip structures:

- connect and/or disconnect tracks (MONDEX fuse)
- combination with laser cutter
- local signal detection and injection
- EEPROM could be read in this way
- make observations during normal operations
- labor-intensive work
- becomes less practical with smaller structures

Micro Probing







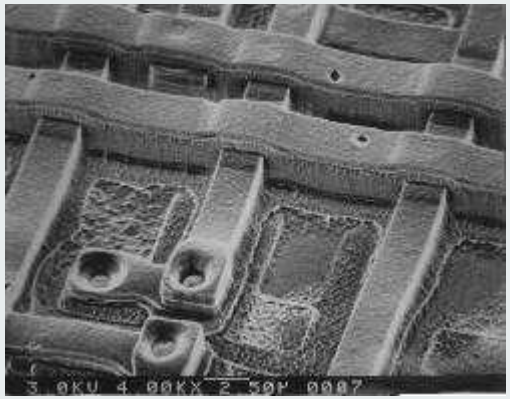
Physical Attacks - E-Beam

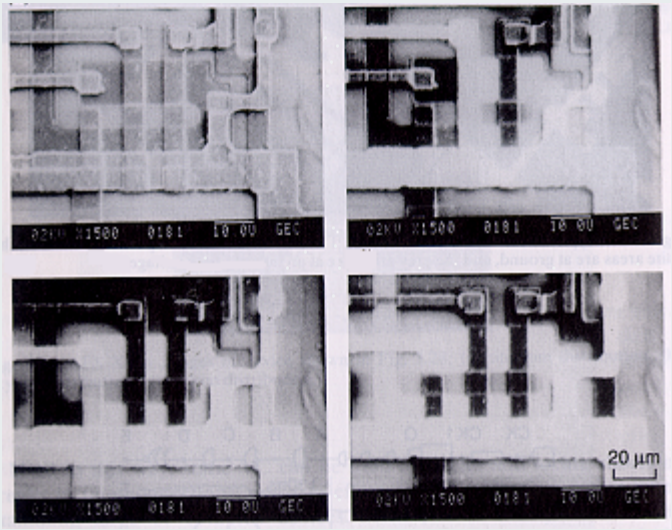
Scanning Electron Microscope for:

- magnification to 100000

Voltage Contrast:

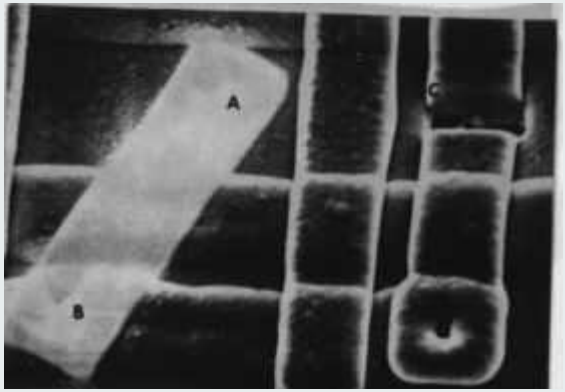
- EEPROM Read-out (data destructive)
- visualize local voltage levels





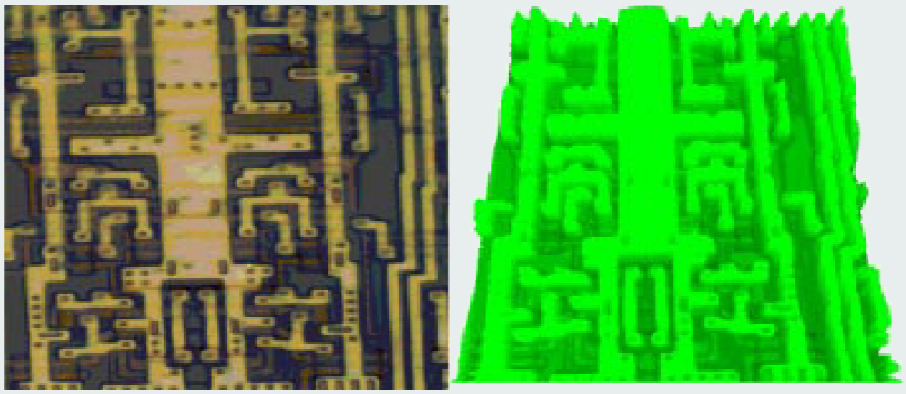
Physical Attacks - Focused Ion Beam

- Like SEM but with charged ions
- cutting and removal of material
- adding conductive material
- adding probing pads

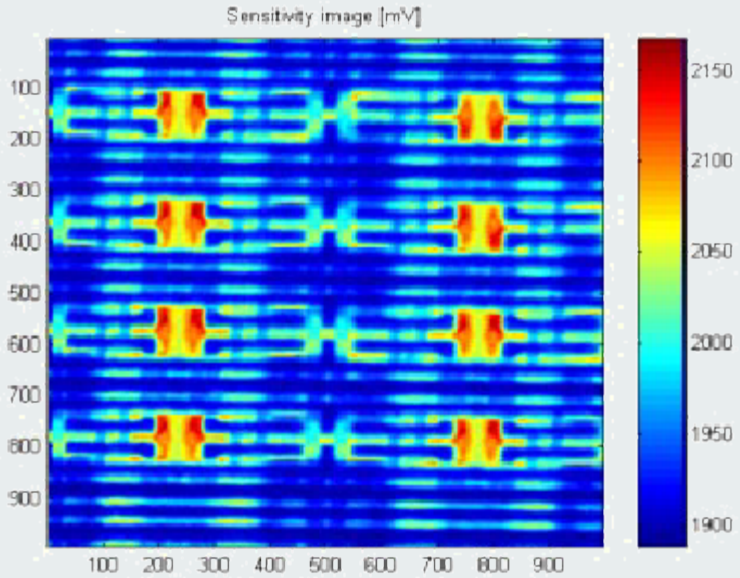


Physical Attacks -Scanning

Magnetic Scanning with Eddy Currents:



Laser Scanning:



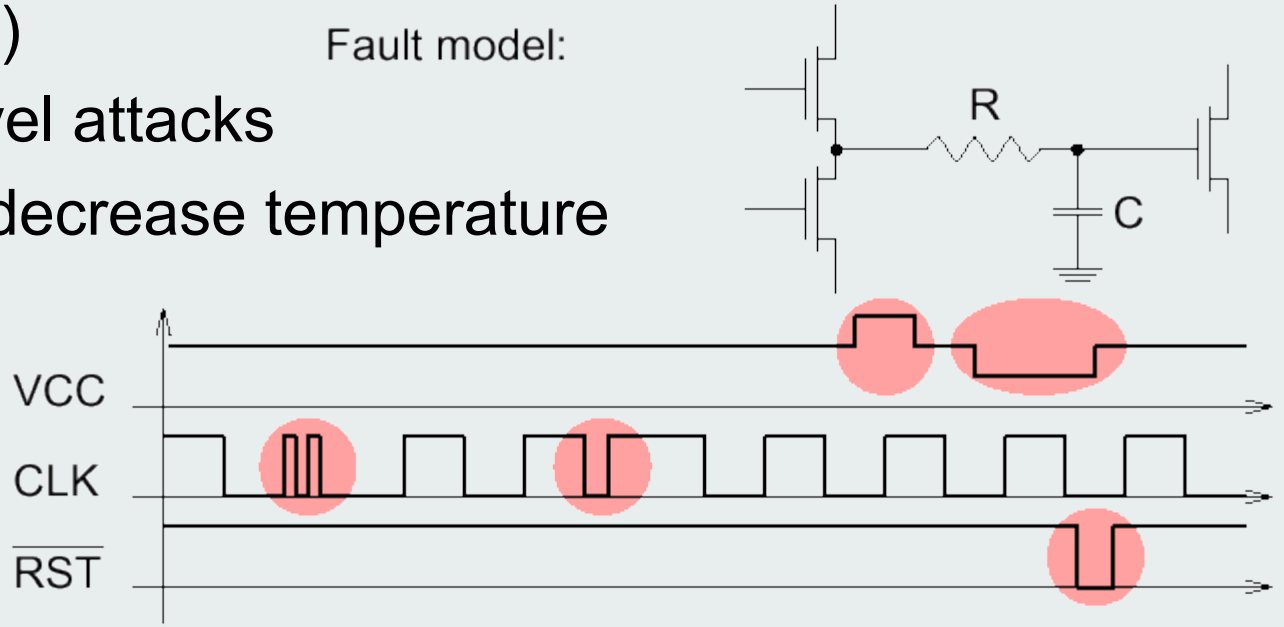
1	1	0	0
1	1	1	0
1	1	1	1
1	1	1	1

www.cl.cam.ac.uk/~sps32/ ww.cl.cam.ac.uk/~rja14/

Physical Attacks - Signal Distortion

deformation of signals to force chip in other state:

- decrease Vcc to block EEPROM write (PIN attack)
- distort CLK to cause a program counter jump (read EEPROM)
- logical level attacks
- increase/decrease temperature
- light
- radiation



Markus Kuhn: http://www.cl.cam.ac.uk/~mgk25/

Purpose of security

Protect information against:

- unallowed disclosure
- alteration
- unavailability

With security functions

- confidentiality
- integrity
- availability

Realization of security functions with, mostly on cryptography, based mechanisms

Authentication

Ascertain authenticity of

- hardware
- individuals
- data

Smart Card hardware authentication:

- internal - CAD ascertains authenticity of card (application)
- external - card ascertains authenticity of CAD
- mutual

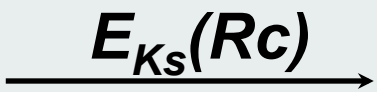
OS functions: GIVE/ASK RANDOM, GET CHALLENGE, INTERNAL/EXTERNAL AUTHENTICATE

A Survey of Authentication Protocol Literature - John Clark and Jeremy Jacob: www-users.cs.york.ac.uk/~jac/papers/drareviewps.ps

Internal Authentication (challenge response)

Smart Card with Ks: **CAD with Kc:**

transmits random nr. *Rc*



transmits Rc enciphered under Ks

OK iff $D_{Kc}(E_{Ks}(Rc)) = Rc$

Symmetric cryptography: **Ks=Kc=secret key**

Asymmetric cryptography: **Ks=smart card private key**

Kc=smart card public key

Replay Attack

Asymmetric, static internal authentication without smart card cryptographic processor:

- during personalization a hash is calculated over static smart card data, signed with a private smart card (issuer) key and stored in the smart card
- at each internal authentication:
  - CAD asks public key certificate from smart card together with the static smart card data and the signed hash
  - CAD verifies public key,calculates the hash and compares this with the checked smart card hash

After first authentication *sniffed* data can be used for authentication without smart card

Static data authentication can only guarantee that the data comes from the private key owner

Mutual authentication

Smart Card with *Ks*: **CAD with *Kc*:**

Rc ←

transmits random nr. *Rc*

transmits concatenation of Rc and random nr. Rs enciphered under Ks

$E_{Ks}(Rc+Rs)$ →

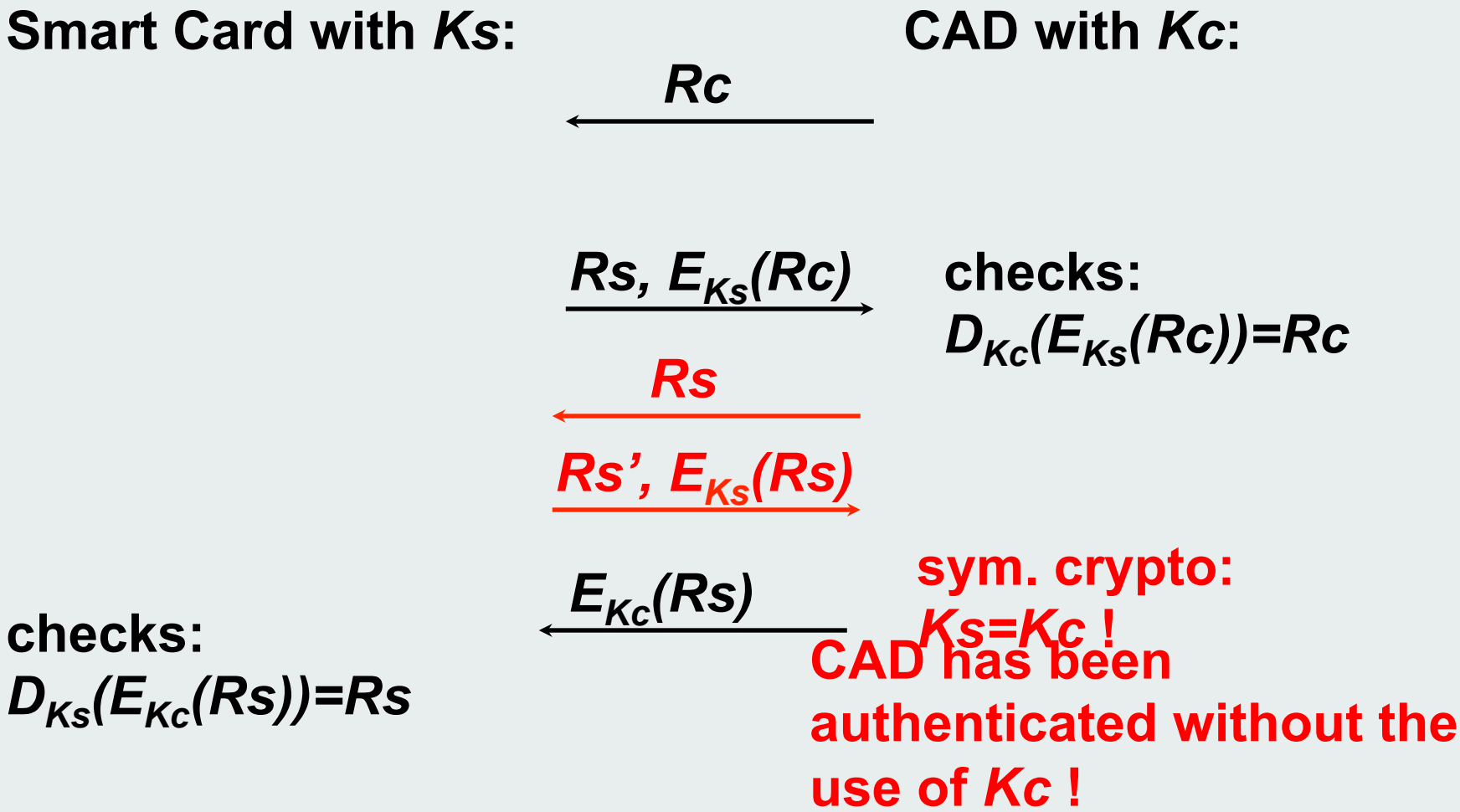
deciphers $E_{Ks}(Rc+Rs)$:
$D_{Kc}(E_{Ks}(Rc+Rs))=Rc+Rs;$

checks Rc and transmits $E_{Kc}(Rs+Rc)$

deciphers $E_{Kc}(Rs+Rc)$:
$D_{Ks}(E_{Kc}(Rs+Rc))=Rs+Rc;$
checks Rc and Rs

$E_{Kc}(Rs+Rc)$ ←

Reflection Attack



GSM SIM internal authentication

SIM with Ki_{16} **HPLMN Ki_{16}**

$TIMSI_8$ →

← RND_{16}

$A3A8_{Ki16}(RND_{16})=(SRES_4, Kc_8)$

$SRES_4$ → **Checks:**

$SRES_4=A3_{Ki16}(RND_{16})$

$Kc_8=A8_{Ki16}(RND_{16})$

GSM SIM internal authentication

ETSI GSM standard specifies A3A8 input & output, not the implementation. MOU example: COMP128

April 1998: COMP128 published and appears to be cripple → *Ki* can be recovered which makes card cloning possible

Software on the internet in weeks including SIM simulation software

Non COMP128: Libertel, KPN, Deutsche Telekom, E-Plus, Vodafone ….

Current fixes:

- COMP128-2/3
- Counter limits number of internal authentications

COMP128

- Based on FFT structure
- Lack of diffusion → Output bytes i, i+8, i+16 and i+24 internal round 2 depend only on input bytes i, i+8 of the challenge *RND*
- Rounds not bijective → different inputs with same outputs, *collisions,* can be found as 2 different RND's with identical (*SRES, Kc*)
- For each collision pair i 2 Ki bytes i, i+8 can be found by a 2-R attack
- Marc Briceno, David Wagner, Ian Goldberg, http://www.scard.org/gsm/

COMP128 SIM attack

```
// find collisions (≈8 hours for 1 SIM)
for (i=0;i<8;i++)
   for RND[i,i+8]=0; RND[i,i+8]≤0xffff)
   {
        (RND,SRES,Kc)Table[i].Add(SIM.RunGSM(RND));
        if ((RND,SRES,Kc)Table[i].FindCollision())
              {collisionmap[i]=(RND, RND'); break}
   }
// 2-R brute force attack on Ki
for (i=0;i<8;i++)
   for Ki[i,i+8]=0; Ki[i,i+8]≤0xffff)
        if (A3A8(ki,collisionmap[i,0])==A3A8(ki,collisionmap[i,1]))
              /* found Ki[i,i+8] */
              break;
```

Authentication of Individuals

Determine who someone really is:

- identification : one-to-many
- verification : one-to-one

With:

- something someone possesses (token=smart card)
- something someone knows (PIN, password)
- personal characteristics (finger-print)

OS functions: VERIFY/CHANGE/UNBLOCK/ DISABLE/ENABLE CHV

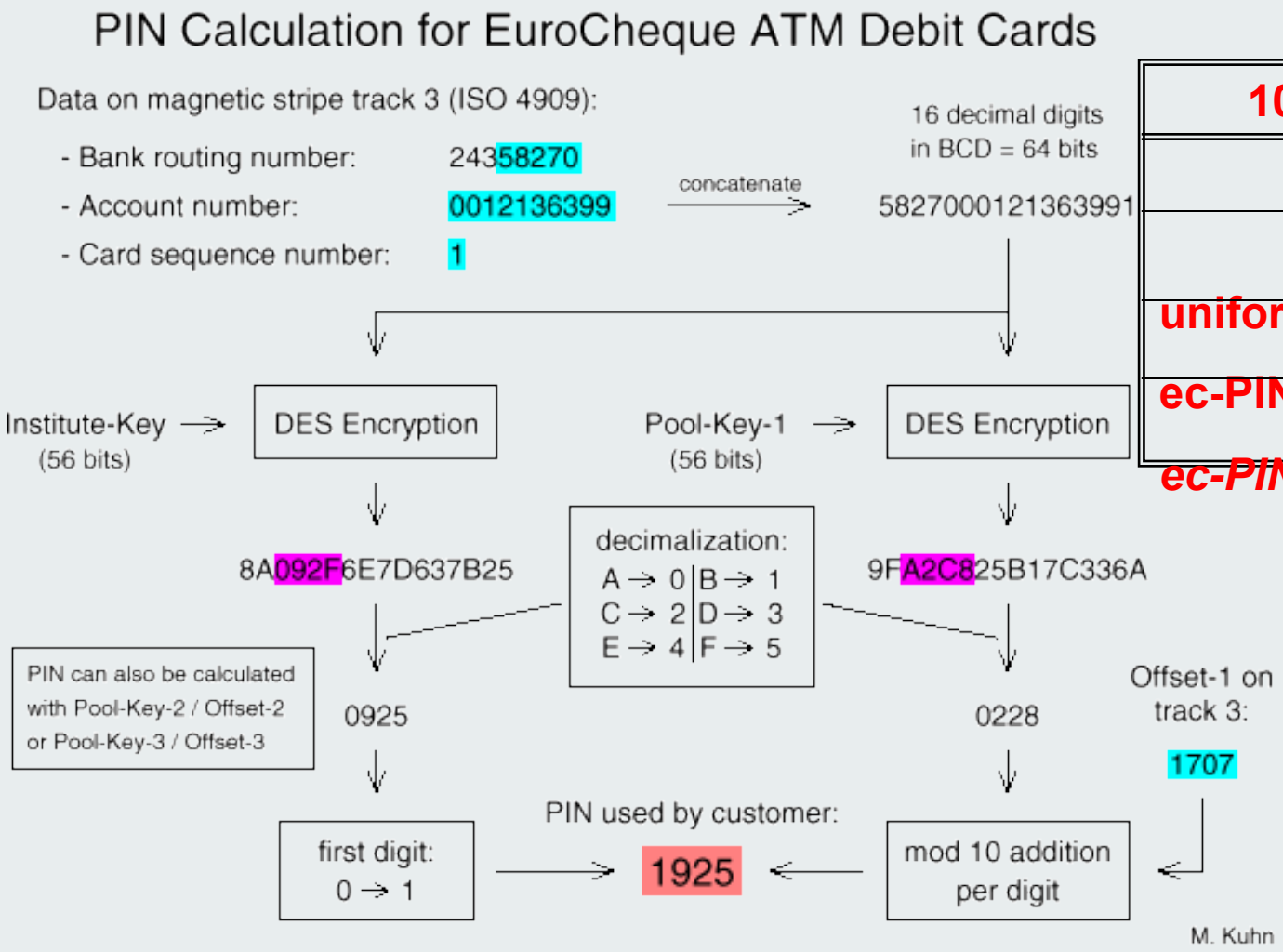
Smart Card PIN verification

- [4 digits] PIN stored in smart card with:
  - <PINmax> [3] maximum number of consecutive false verifications
  - <PINcur> present number of consecutive false verifications
- Authentication impossible iff: <PINcur> ≥ <PINmax>
- New PIN with [8 digits] PUK:
  - authentication permanently impossible iff <PUKcur> ≥ <PUKmax> [10]
- 4-digit PIN → 10.000 possibilities
- Prevention of EEPROM write operations to <PINcur>, <PUKcur> makes brute-force possible → write <P??cur>=<P??cur>+1 before verification !
- Use of 1 PIN for hybrid cards (magstripe/chip) → *sniffed* PIN during chip usage makes magstripe cloning possible
- Digit guessing → verification implementation must be time invariant

PIN verification

- PIN sometimes not random but result of cryptographic operation with card data and secret key.
- Use of hexadecimal representation during PIN generation causes non uniform distribution → increasing guess probability

(Old fashioned) PIN verification



1000 ≤ PIN ≤ 9999	
	P(PIN correct in 3 attempts)
uniform	= 1/3000
ec-PIN	≈ 1/150
ec-PIN+O1…O3	*≤ 1/105*

Probability Theory for Pickpockets -- ec-PIN Guessing, Markus G. Kuhn

PIN verification

September 22, 1998

German Court Ruling Another Blow to U.S. Encryption Standard

By Mary Lisbeth D'Amico

MUNICH – A German district court has ordered a bank in Frankfurt to repay a customer 4,543 marks (US$2,699) for money withdrawn from her bank account after her bank card was stolen.

The decision, made public Monday, again points to the holes in the 56-bit encryption technology used in Eurocheque cards, called EC Cards, according to the Chaos Computer Club, a German hackers group.

Calling the encryption technology for the EC bank cards "out-of-date and not safe enough," a Frankfurt District Court held the bank responsible for the amount stolen from the 72-year old plaintiff in February 1997. Neither the bank's name or that of the plaintiff were revealed……

http://www.cnn.com/TECH/computing/9809/23/germancrypt.idg/

Data authentication

Ascertain data integrity with:

One-way hash function *H*:

- input arbitrary amount of data *D,* output *h* with fixed length so that:
  - given *D*, calculation of *h* is easy
  - given *h*, difficult to find *D* with *H(D)=h*
  - given *D,* difficult to find *D'* with *D'≠D* and *H(D)=H(D')*
- Common hash functions: SHA-1 and MD5
- Hashing for data integrity requires protection of the hash

Data authentication

Message Authentication Code (MAC):

- symmetric enciphered hash added to the authenticated data (e.g. DES in CBC mode or RFC2104 HMAC)
- hash is protected (if key is protected)
- source known, key shared

Digital Signing

- asymmetric, with private key, enciphered hash added to the authenticated data
- hash is protected (if key is protected)
- source known, key not shared
- DSS NIST standard

Data authentication

Certificates:

- Document signed by a Certificate Authority (CA)
- Identity verifiable by everyone who trust CA
- guaranties authenticity of document and source
- popular for public key exchange
- standards: X.509, PKCS

Prevention of replay attacks through addition of unpredictable (random) data to data being authenticated by checking party.

Data authentication - Protected Read

Smart Card with Ks: **CAD with Kc:**

generates random nr. *Rc*

ReadProtected(file,offset, length, Rc)

reads Data,
calculates MAC_{Ks}(Rc, Data)

Data + MAC

Calculates MAC'_{Kc}(Rc, Data)

OK iff MAC'=MAC

Data authentication - Protected Write

Smart Card with Ks: | **CAD with Kc:**

generates random nr. *Rc*

Rc →

calculates $MAC_{Kc}(Rc, Data)$

← *WriteProtected(file,offset, Data, MAC)*

calculates $MAC'_{Ks}(Rc, Data)$

Writes iff MAC'=MAC

more on cryptographic protocols in ISd2

Authorization

Smart Card security context: situation after authentication of smart card, CAD, and user

Authorization mechanisms determine which actions are permitted in the current security context

Access Control List (ACL) in file header:

ALWAYS	Operation always allowed
CHV1	Operation allowed after succesfull Card Holder Verification with PIN 1
CHV2	Operation allowed after succesfull Card Holder Verification with PIN 1
AUT	For this operation the card (application) must be authenticated succesfully
PRO	For this object a Message Authentication Code must be calculated
CHV1 & AUT	
CHV2 & AUT	
CHV1 & PRO	
CHV2 & PRO	
CRYPT1	Communication is encrypted with key 1
CRYPT2	Communication is encrypted with key 2
NEVER	Operation is never allowed

Authorization

Drawbacks:

- support for security contexts
- support of combinations
- fixed on files instead of objects
- parameter support
- bad extendibility (bit coded)
- storage space proportional to #files

Possible solutions:

Access Control Object (ACO)

Windows for Smart Card ACL approach

ACO

	EF_PATIENT	EF_DIAGNOSIS	EF_MEDICINE
Doctor	EXTAUT (AUT#1)	EXTAUT (AUT#1)	EXTAUT (AUT#1)
(Security Env #1)			
	READ (PRO#2)	READ (PRO#2)	READ (PRO#2)
		UPDATE (PIN#1,PRO#3)	UPDATE (PIN#1, PRO#3)
		APPEND (PRO#3)	APPEND (PRO#3)
Assistant	EXTAUT (AUT#2)	EXTAUT (AUT#2)	EXTAUT (AUT#2)
(Security Env #2)			
	READ (PRO#2)	READ (PRO#2)	READ (PRO#2)
	UPDATE (PRO#4)		
Pharmacist	EXTAUT (AUT#3)	-	EXTAUT (AUT#3)
(Security Env #3)			
	READ (PRO#5)		READ (PRO#5)
			CONF_MED (PRO#5)
Patient	READ (PIN#2)	READ (PIN#2)	READ (PIN#2)
(Security Env #4)			

Proprietary condition: only permitted iff CONF_MED has not been executed

***The Security Functions of Access Control Objects in Smart Cards* – Helmut Scherzer, IBM Germany, in Proceedings of CardTech/Securtech Orlando May 1997.**

ACO

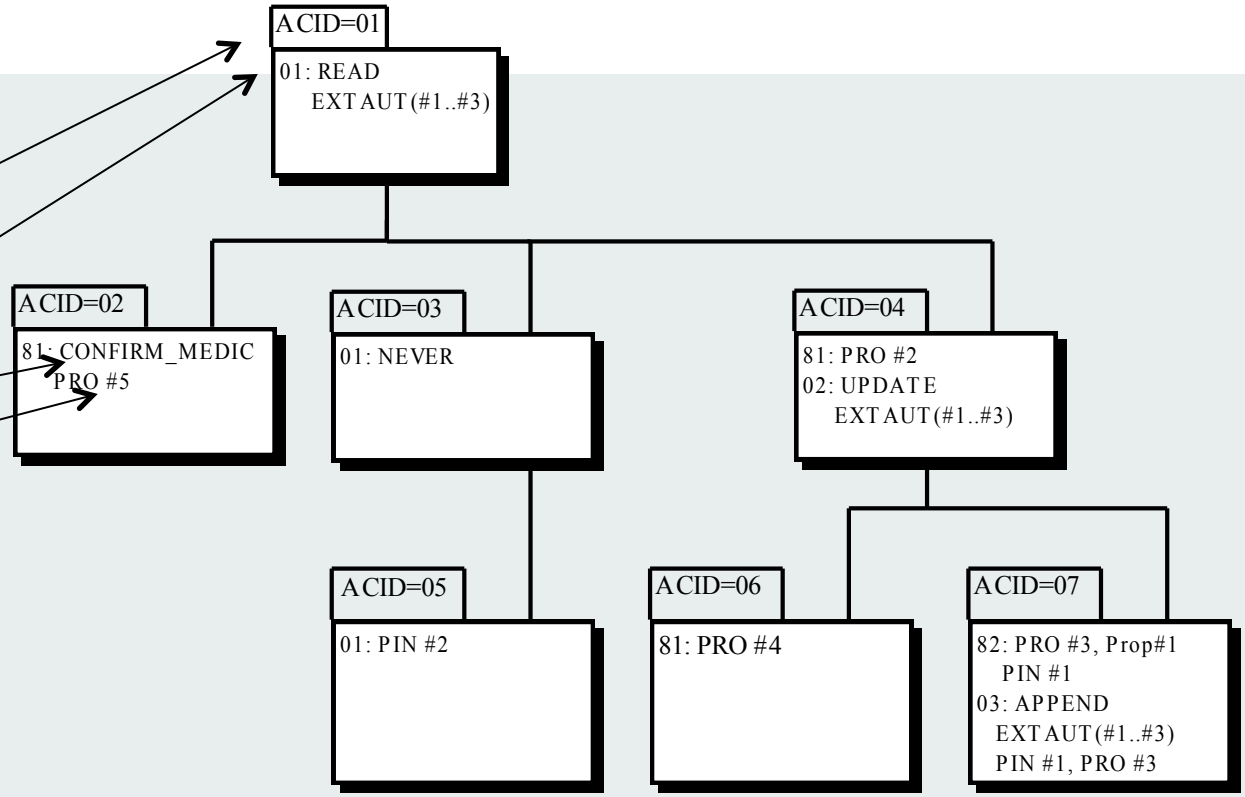
ACO graph:

ACID

house nr.

SCD

ACP



File header info:

FILE_ID	T	L	V (ACID || Security Env.)
EF_PATIENT	86	8	01 01 06 02 02 03 05 04
EF_DIAGNOSIS	86	8	07 01 01 02 03 03 05 04
EF_MEDICINE	86	8	07 01 01 02 02 03 05 04

***The Security Functions of Access Control Objects in Smart Cards* – Helmut Scherzer, IBM Germany, in Proceedings of CardTech/Securtech Orlando May 1997.**

ACO

Algorithm:

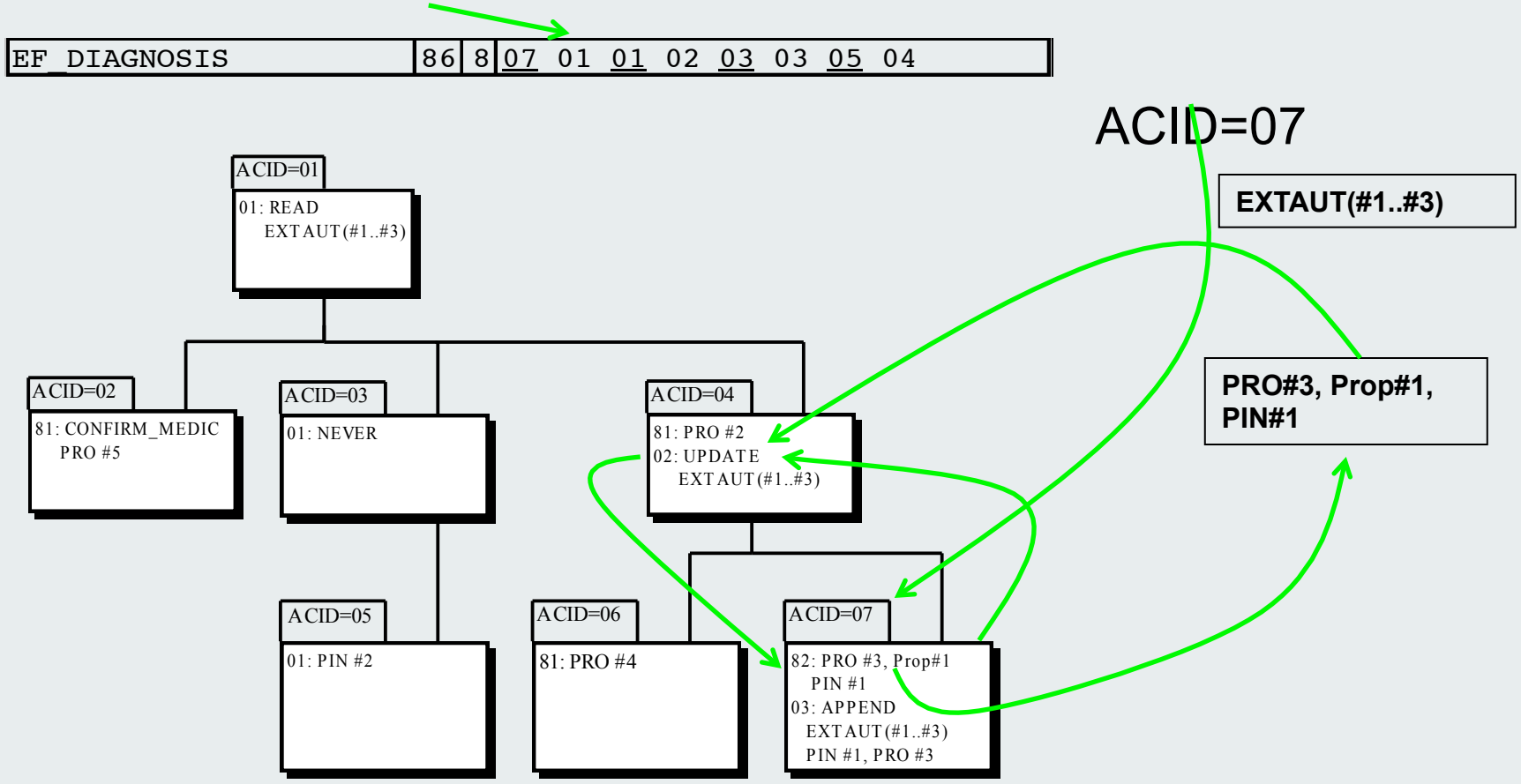
- Given object ACID + security context + command;
- Search command in ACO with ACID. Command not found → go to parent ACO and repeat search until root . Command not found → condition of command: NEVER.

  Command found → remember "house number" from ACO in which command was found.
- Go back to first ACO and search upwards to "house number". Here's the object access condition. For "house numbers" > 0x80 the most significant bit has to be removed first. If it matches the condition counts but the search continues via parent ACO.

***The Security Functions of Access Control Objects in Smart Cards* – Helmut Scherzer, IBM Germany, in Proceedings of CardTech/Securtech Orlando May 1997.**

ACO example

UPDATE Authorization of EF_DIAGNOSIS by a Doctor:



Windows for Smart Cards (WfSC) ACL's

- Known Principals (KP) for Authentication
  - issuer, owner, CAD, application …
  - each KP has a reference to an authentication protocol
    - PIN, challenge-response, applet …
  - Group is a collection of KP's
    - Group authenticated iff at least 1 KP is authenticated
    - Group de-authenticated iff all KP's are de-authenticated
- Access Control List (ACL):
  - list with actions and Boolean KP expressions
  - action is permitted iff expression is valid (KP is authenticated)

Windows for Smart Cards ACL's

- Each file, including each ACL file, on the card requires an ACL file that determines access rules
- File structure after card initialization (bootstrap):

/	(directory)	ACL *default*
/s	(directory)	ACL *default*
/s/k	(directory)	ACL *kpdir*
/s/k/index	(file)	ACL sys
/s/k/anonymous	(file)	ACL *anonymous*
/s/a	(directory)	ACL *default*
/s/a/anonymous	(file)	ACL *anonymous*
/s/a/default	(file)	ACL *anonymous*
/s/a/kpdir	(file)	ACL *anonymous*
/s/a/sys	(file)	ACL *anonymous*

GSM <-> WfSC mapping

Identifier: '2FE2'	Structure: transparent	Mandatory
File size: 10 bytes	Update activity: low	

Access Conditions:

READ	ALWAYS
UPDATE	NEVER
INVALIDATE	ADM
REHABILITATE	ADM

Bytes	Description	M/O	Length
1 - 10	Identification number	M	10 bytes

Identifier: '6F3A'	Structure: linear fixed	Optional
Record length: X+14 bytes	Update activity: low	

Access Conditions:

READ	CHV1
UPDATE	CHV1
INVALIDATE	CHV2
REHABILITATE	CHV2

Bytes	Description	M/O	Length
1 to X	Alpha Identifier	O	X bytes
X+1	Length of BCD number/SSC contents	M	1 byte
X+2	TON and NPI	M	1 byte
X+3 to X+12	Dialling Number/SSC String	M	10 bytes
X+13	Capability/Configuration Identifier	M	1 byte
X+14	Extension1 Record Identifier	M	1 byte

ETSI GSM	**WfSC GSM**
Authentication	
ALWAYS	KP ANONYMOUS
CHV1	KP CHV1 + KP PUK1
CHV2	KP CHV2 + KP PUK2
ADM1… ADM10	KP ADM1 … KP ADM10
ADM	KP Group ADM with KP ADM1…ADM10
NEVER	No ACL entry (or KP index)
Authorization	
READ	READ
UPDATE	WRITE
INVALIDATE	custom INVALIDATE
REHABILITATE	custom REHABILITATE
INCREASE	custom INCREASE

Confidentiality

Symmetric:

- DES
- 3DES
- AES

Asymmetric

- RSA
- Elliptic curves

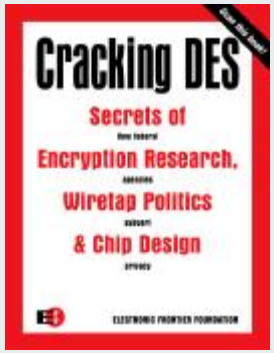
Choice of algorithm: proven technology, scalability, key management, cost, privacy …

more on cryptographic algorithms in ISd2

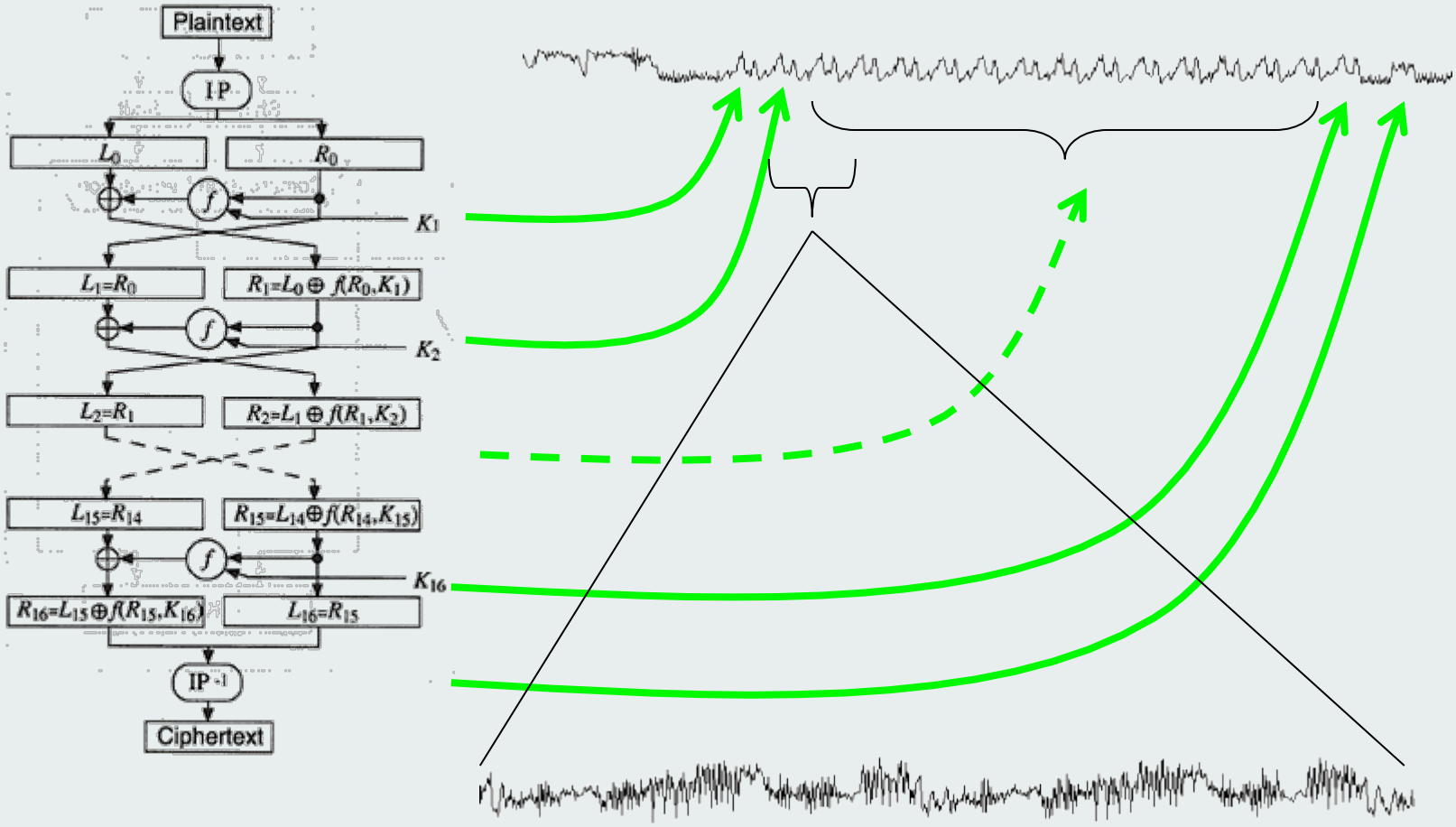
Symmetric systems (DES)

- Civil use since 70's
- *Bit manipulations* → hardware efficient
- No real scalability (not *key-upgradable)*
- Key management → both sides need secure device (SAM)
- Generic attack: exhaustive key-search
  - #keys=$2^{56} \approx 7.2*10^{16}$
  - distributed computing via internet: key within weeks: http://distributed.net/
  - dedicated hardware: key within 1 day
    http://www.eff.org/descracker/
  - Triple-DES → #keys = $2^{112} \approx 5.2*10^{33}$

    exhaustive key-search not realistic

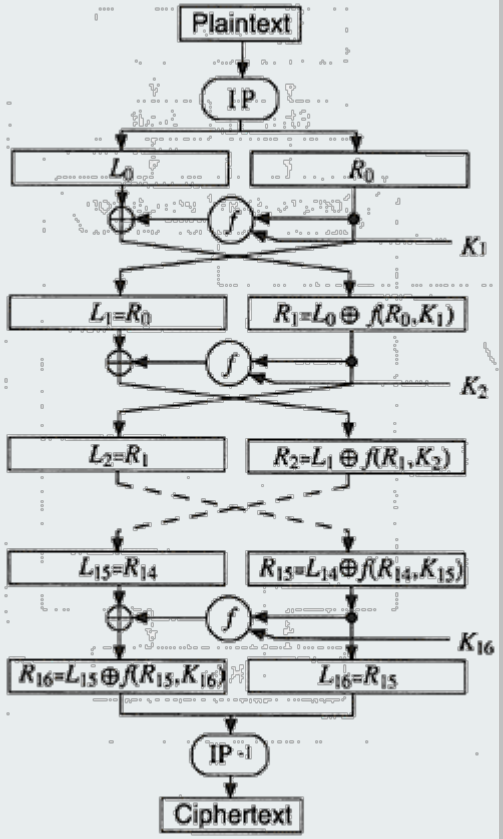


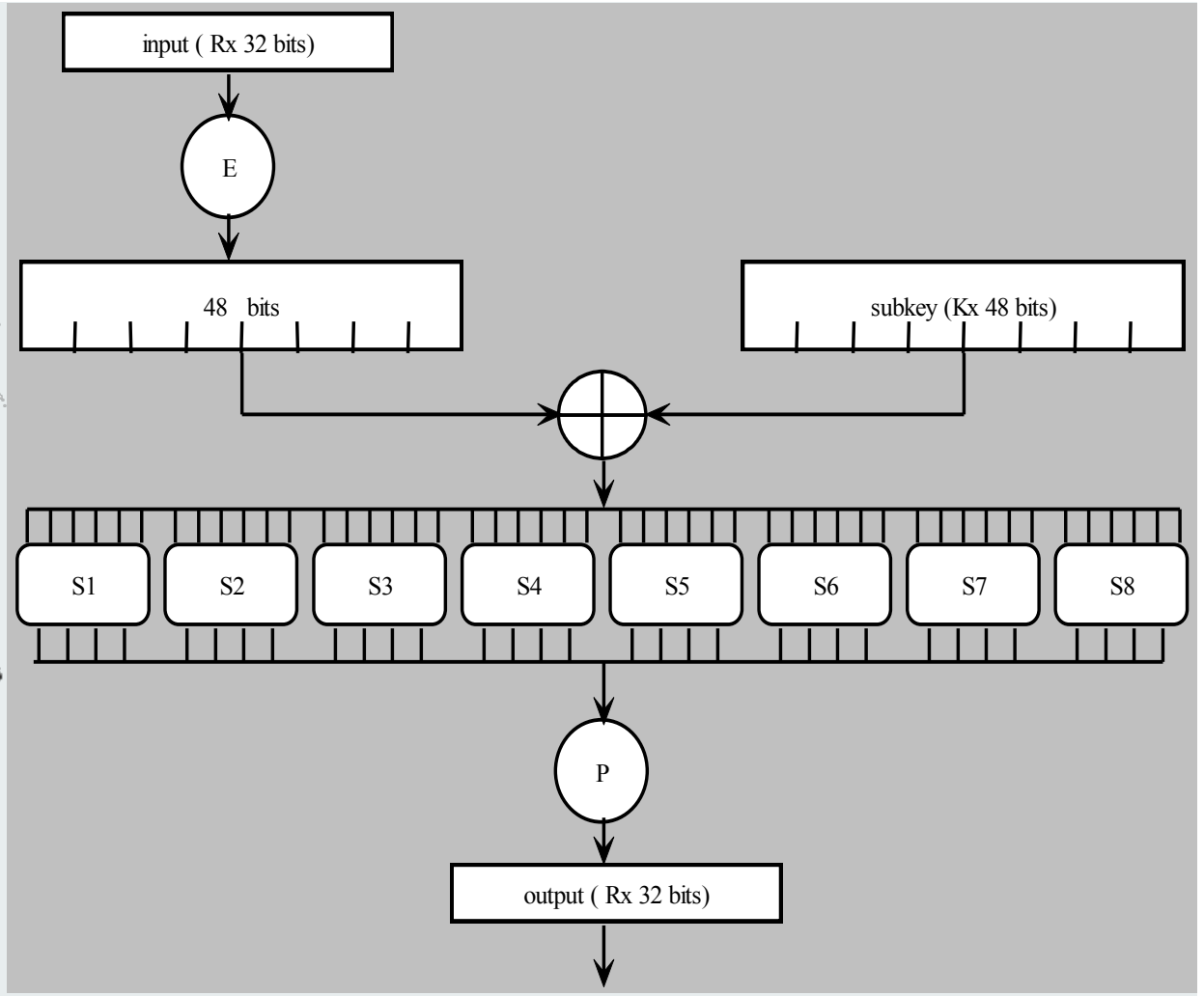
DES implementation attacks - DPA



(http://www.cryptography.com/)

DES implementation attacks - DPA





(http://www.cryptography.com/)

DES implementation attacks - DPA

Variables

- key length : l=56
- sub-key length : s=6
- #S-BOXES : b=8
- #characters : c=2
- average #test keys : t

Goal

- Reduction from exhaustive key search ($t=0.5 \times c^l = 2^{55} = 36028797018963968$) towards exhaustive sub-key search ($t= 2 \times b \times c^s = 1024$)
- Other advantage: only plain OR crypto text needed

DES implementation attacks - DPA

- Collection phase: execute DES with *n* random inputs measuring the Power *P* :

$$P_{i,t}: \ i = \text{input}, \ t = \text{time} \in [0 \ldots T]$$

- Differential Key Search: For each possible sub-key *s* divide P into two *summed traces*: P_0 and P_1 based on a selection function *D* which is dependent on i, s and *something of the sub-key which is correlated to the actual current* (e.g. S-BOX leftmost output bit)
  - incorrect sub-key⇒D uncorrelated to actual Power consumption ⇒ random partitioning ⇒ P1 - P0 ≈ 0
  - correct sub-key⇒D correlated to actual Power traces ⇒ P1 - P0 ≠ 0
- Round 1 reveals 48 key-bits, other 6 bits can be found with same technique on round 2

DES implementation attacks - DFA

Differential Fault Analysis (DFA): (Bellcore, Biham, Shamir ...)

message *m*, *n* bits key *k*, cipher text *c*, c_f=encryption of m with *all_zero_key* k_f

```
// break-down stage
i=0; c_i=E(m);i++;
while (E(m)!=c_f)
{
   use physical stress to force single 1 → 0 key bit change;
   if (new value E(m))
     {c_i=E(m);i++}
} // built-up stage
k=k_f;i=f;
while (i)
{
   while (E_k(m)!=c_i)
      try other k with single 0 → 1 bit change
   i--;
}
```

DES implementation attacks - DFA

Differential Fault Analysis (DFA): (TNO EIB)

Do same encryption twice, first without any faults, second with faults that corrupt the outcome of the 15th round:

$R_{16}=F(R_{15},K_{16}) \oplus L_{15}$

$R'_{16}=F(R'_{15},K_{16}) \oplus L_{15}$

---------- $\oplus$

$R_{16} \oplus R'_{16} = F(R_{15},K_{16}) \oplus F(R'_{15},K_{16})$ $\Leftarrow$ only K_{16} Unknown !

$= S(E(R_{15}) \oplus K_{16}) \oplus S(E(R'_{15}) \oplus K_{16})$

K_{16} can be brute-forced individually for each S-BOX i where $(R_{16} \oplus R'_{16})_i \neq 0 \Rightarrow 2^6=64$ candidates for all such S-BOX'es

DES implementation attacks

Differential Fault Analysis (DFA):

- single bit changes not easy
- execution errors more likely (clock glitching)

Differential Power Analysis (DPA):

- leakage of run-time OS (crypto algorithm) information via smart card power consumption
- not only a DES threat
- Counter measures:
  - hardware leakage reduction (filters, noise generators …)
  - software adaptations (loop balancing, retry counters …)

more on side channel analysis in ISd2

Asymmetric systems

- Concept from 1976, in use during last decades
- Algebraic structures → less hardware efficient
  - discrete logarithm
  - factoring
  - elliptical curves (efficient in processing power and key-length)
- Scalable *(key-upgradable)*
- Key management → secure device for private key, certificate for public key
- Cryptographic coprocessor needed + key EEPROM → more expensive
- Key generation takes time !

RSA - Implementation Attacks

$s = x^y \bmod n$ *via binary square & multiply*:

```
s = 1;
while (y)
  {
        if (y&1)
            s=(s*x) mod n;
        y>>=1;
        x=(x*x) mod n;
  }
 return (s);
```

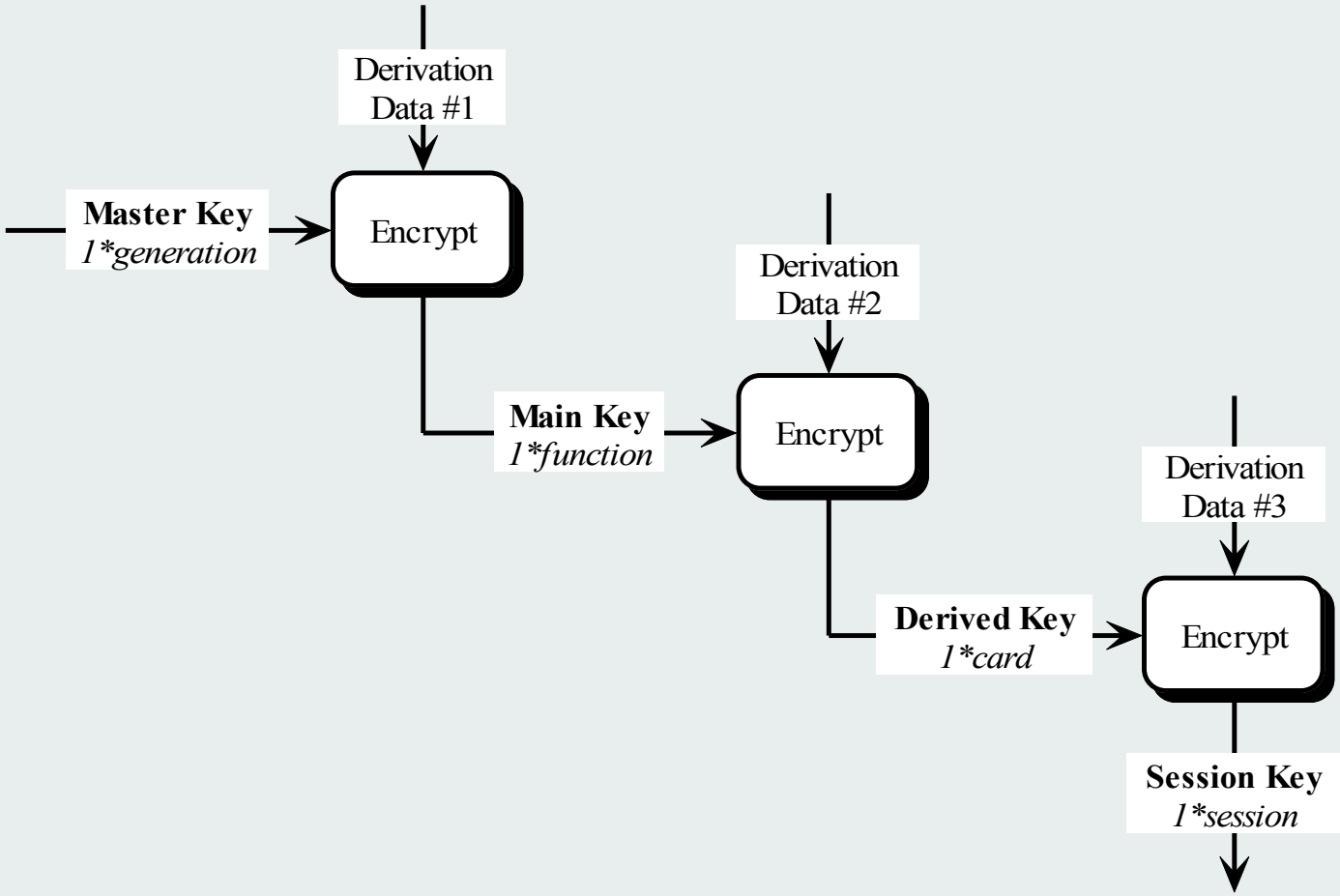
Only multiply iff corresponding key-bit is 1 !

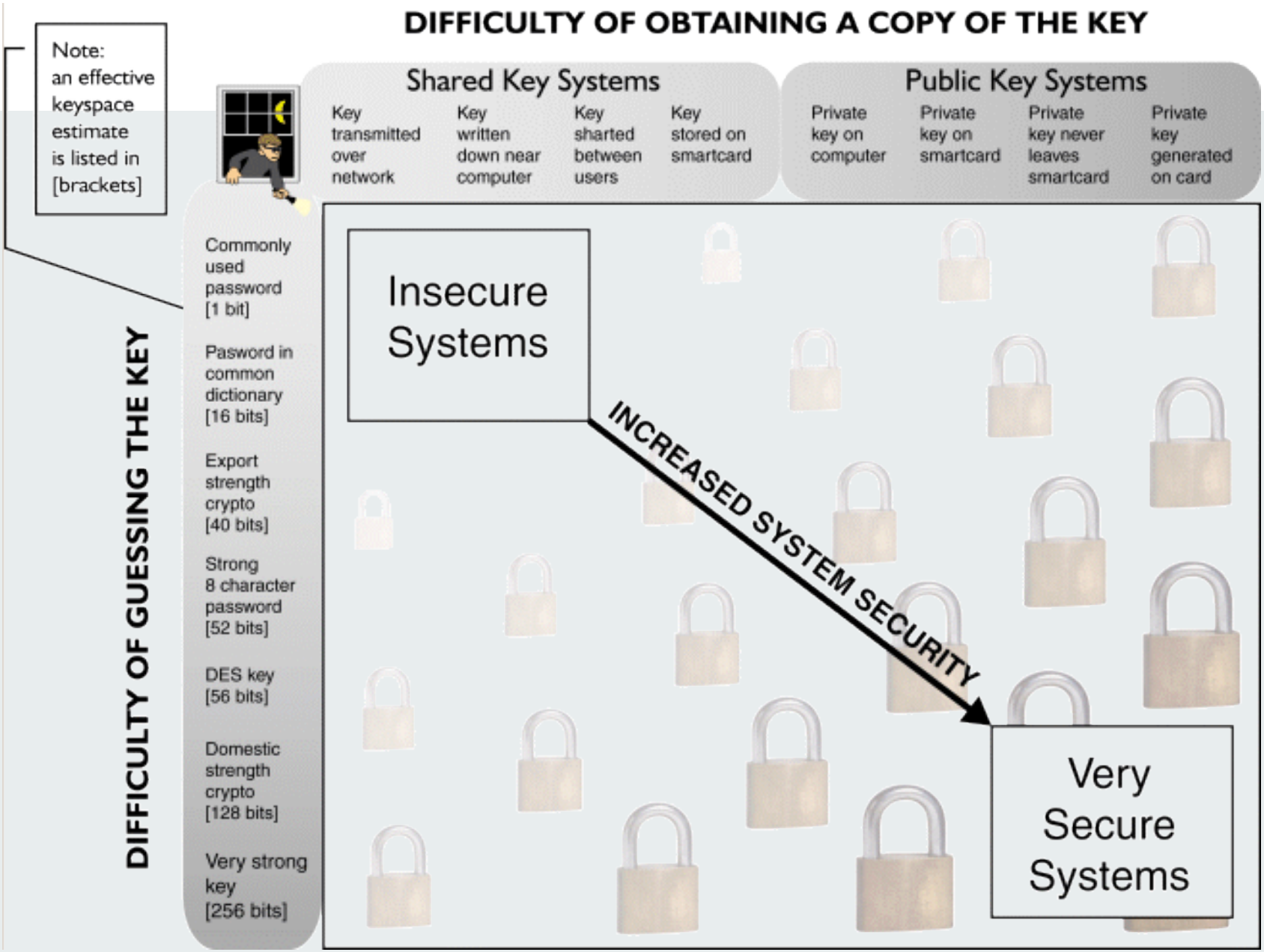
Time measurement : Timing Attack

Crypto Processor (On/Off) : (D)PA Attack

Paul Kocher (http://www.cryptography.com/)

Smart Card Symmetric Key Hierarchy





Source: Litronic

Privacy

identity known - identity card during postal parcel pick up

pseudo-anonymous ATM transaction at shopping mall

anonymous - phone call with pre-paid GSM subscription

unlinkable - successive phone calls from phone booth with the same pre-paid phone card

unobservable - freedom network??

Security Evaluation

- Certainty about system correctness and system sensitiveness
- Security objectives determine the scope of evaluation expressed in *evaluated assurance level*
- Makes comparison possible
- *Criteria:* Agreements about the evaluation process
- formal requirements + number of tests

more on evaluation in ISd2

IT Security Criteria

- *Trusted Computer Security Evaluation Criteria* (TCSEC) = *Orange Book*
- *Information Technology Security Evaluation Criteria* (ITSEC)
  - *IT Security Evaluation Manual* (ITSEM) - methods for the execution of ITSEC evaluations
- *Common Criteria for Information Technology Security Evaluation* (CC)
  - Version 2.1 ISO/IEC 15408

Common Criteria for Information Technology Security Evaluation

- 8 Evaluation Assurance Levels (EAL's)
  - EAL0 - insufficient certainty

    …
  - EAL7 (implementation) ⇒ (formal specification)
- Strength Of Function (SOF) classification:
  - **SOF-basic** *smart outsider;* intelligent; not enough system know-how; no advanced equipment
  - **SOF-medium** *educated insider;* experience and advanced equipment
  - **SOF-high** *financed organization;* hire specialists and equipment

Smart Card Protection Profiles

- Protection Profiles: Definition of security requirements for a product group, independent of implementation
- Smart Card PP:

description of TOE and position in smart card life cycle

ID	Type	Version	Status	EAL	SOF
PP/0303	Java Card Protection Profile	1.0b			
	SCSUG Smart Card Protection Profile	3.0	E+V	4	High
PP/0010	Smart Card IC with Multi-Application Secure Platform	2.0	C	4	High
PP/0002	Transactional SC Reader	2.0	C	4	High
PP/9911	Smart Card Integrated Circuit with Embedded Software	2.0	C	4	High
PP/9909	Intersector Electronic Purse and Purchase Device	1.2	C	4	High
PP/9903	Transportation ticketing contact & contactless	1.2	C	4	High
PP/9810	Smartcard Embedded Software Protection Profile	1.2	C	4	High

to guarantee the level of security

http://niap.nist.gov/cc-scheme

Risk Analysis

- Social risks of smart cards
- Risk=Probability*Effect
  - negligible: 1 card holder
  - small: 1 card issuer (loss of money and reputation)
  - large: go beyond the card issuer (system compromised)
- Risk classes on behalf of prevention
  - impersonal smart cards (pre-paid SIM)
  - smart cards with (pseudo)-identity (chipknip)
  - smart cards with third-party identity function (W-document)

Prevention Measures

government intervention		public law enforcement	creating additional conditions	additional government control
Card category		• self-regulation	• self-regulation • identification obligation for card issuers; obligation to submit proof of identity for card holders; for card issuers a legal right to check the validity of a submitted proof of identity	• self-regulation • identification obligation for card issuers; obligation to submit proof of identity for card holders; for card issuers a legal right to check the validity of a submitted proof of identity • licence system and government supervision of the issuing process
I	**impersonal cards**	**light regime** prepaid phonecard, prepaid non-loada-ble electronic purse (gift voucher), prepaid GSM-card		
II	personalised cards with a contractual **(pseudo-)identity function** (not to be used by third parties)		**normal regime** PIN-card, loadable electronic purse, GSM-card, asylum seekers identity card, city card (with or without biometrics)	
III	personalised cards with a **general identity function** (intended for use by third parties)			**heavier regime** city card with personalised biometrics intended for general use, aliens identity card, electronic driving licence

www.minjust.nl/c_actual/rapport/smartnl.htm

Design Hints

- Maximize reuse
  - standards
  - algorithms
  - API's
  - risk analysis
  - incidents
- Use an evaluation method
  - Common Criteria …
- Peer review
- *Security by Obscurity* can only work as a delay factor
- Take system failure as a starting point of security
  - how is it detectable
  - how can the damage be restricted
  - how can the failure be corrected