PV079: Cryptographic smartcards and their applications

Cryptographic secure hardware

Petr Švenda svenda@fi.muni.cz @rngsec
Centre for Research on Cryptography and Security, Masaryk University

Centre for Research on Cryptography and Security

Overview

- Smartcards – introduction
- Applications – where to use?
- Smartcard programming
- Side-channel attacks
  - power analysis
  - reverse engineering
  - timing attacks

INTRO TO SMART CARDS

Basic types of (smart) cards

1. Contactless "barcode"
   – Fixed identification string (RFID, < 5 cents)
2. Simple memory cards (magnetic stripe, RFID)
   – Small write memory (< 1KB) for data, (~10 cents)
3. Memory cards with PIN protection
   – Memory (< 5KB), simple protection logic (<$1)

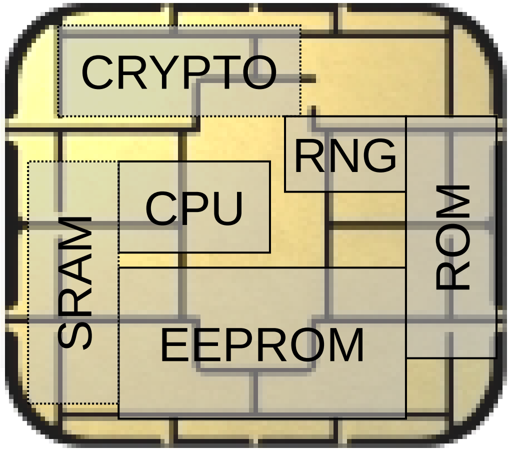
Basic types of (smart) cards (2)

4. Cryptographic smart cards
   - Support for (real) cryptographic algorithms
   - Mifare Classic ($1), Mifare DESFire ($3)
5. User-programmable cryptographic smart cards
   - JavaCard, .NET card, MULTOS cards ($2-$30)
   - Chip manufacturers: NXP, Infineon, Gemalto, G&D, Oberthur, STM, Atmel, Samsung...
6. Secure environment (enclave) inside more complex CPUs
   - ARM TrustZone, Intel SGX…

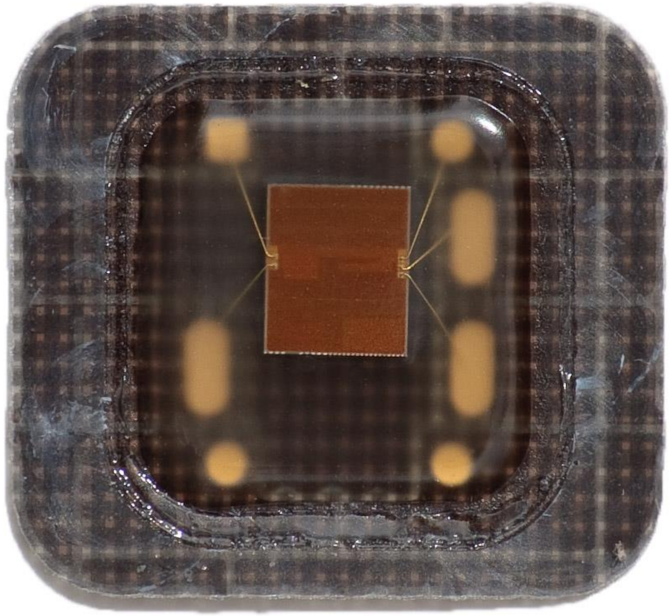
We will mainly focus on categories 4 and 5

Credit Wikimedia Commons

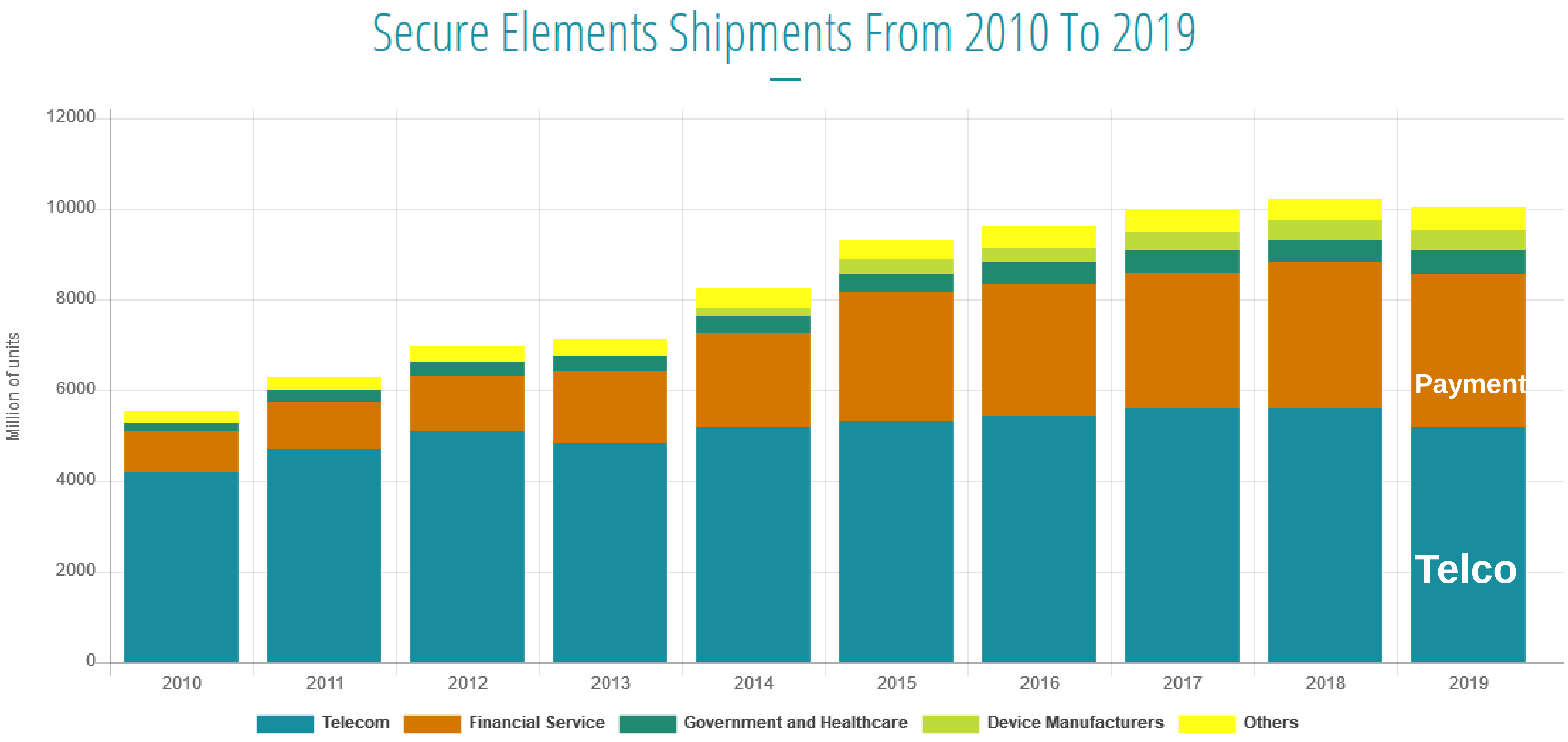
Cryptographic smart cards



- SC is quite powerful device
  - 8-32 bit processor @ 5-50MHz
  - persistent memory 32-200+kB (EEPROM)
  - volatile fast RAM, usually <<10kB
  - truly random generator
  - cryptographic coprocessor (3DES,AES,RSA-2048,ECC...)
- ~10 billion units shipped in 2019 (EUROSMART)
  - mostly smart cards, telco, payment and loyalty...
  - ~1.5 billion contactless (EUROSMART)
- For environments where attacker have physical access
  - NIST FIPS140-2 standard, security Level 4
  - Common Criteria EAL4+/5+



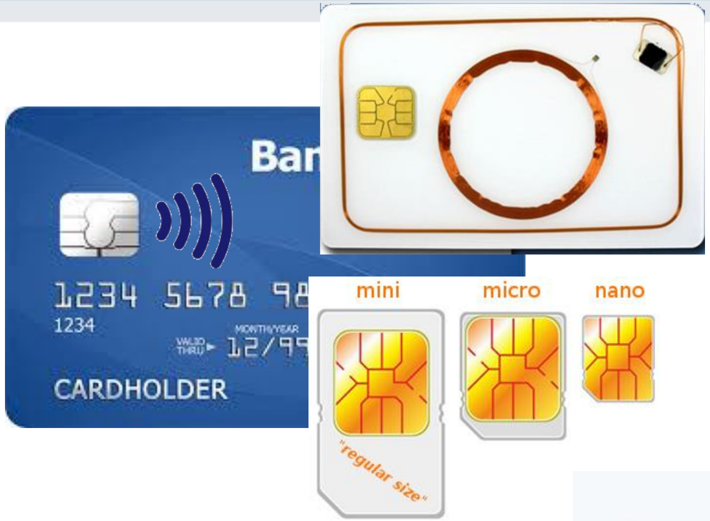
Primary markets for smartcards



https://www.eurosmart.com/eurosmarts-secure-elements-market-analysis-and-forecasts/

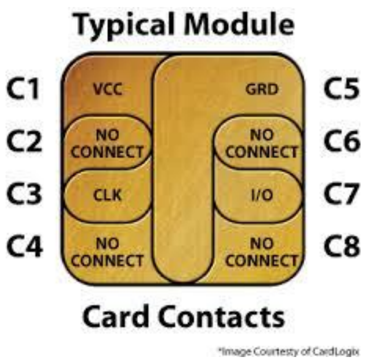
Smart cards forms

- Many possible forms
  - ISO 7816 standard
  - SIM size, USB dongles, Java rings…
- Contact(-less), hybrid/dual interface
  - contact physical interface
  - contact-less interface (NFC phone can communicate!)
  - hybrid card – separate logics on single card
  - dual interface – same chip accessible contact & c-less
- Card emulation (contactless)
  1. Card emulation mode (physical in-phone secure element)
  2. Host-based card emulation (without physical element)
     - Apple Pay, Google Pay



http://simcardsize.com/sim-card-sizes/ https://shop.cobo.com/products/cobo-vault-essential https://www.infineon.com/ https://yubico.com

Contact vs. contactless, powerless vs. battery-powered



- Contact cards (ISO7816-2)
  - I/O data line, voltage and GND line
  - clock line, reset lines
- Contactless cards
  - ISO/IEC 14443 type A/B, radio at 13.56 MHz (NFC)
  - Chip powered by current induced on antenna by reader
  - Reader → chip communication - relatively easy
  - Chip → reader – dedicated circuits are charged, more power consumed, fluctuation detected by reader
  - Multiple cards per single reader possible

- Additional battery possible
  - Higher cost, need to charge, but longer distance and faster communication (Bluetooth LE)

Smart card is highly protected device

- Intended for physically unprotected environment
  - NIST FIPS140-2 standard, security Level 4
  - Common Criteria EAL5+/6+…
- Tamper protection
  - Tamper-evidence (visible if physically manipulated)
  - Tamper-resistance (can withstand physical attack)
  - Tamper-response (erase keys…)
- Protection against side-channel attacks (timing, power, EM)
- Periodic tests of TRNG functionality
- Approved crypto algorithms and key management
- Limited interface, smaller trusted computing base (than usual)
  - http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140val-all.htm
- Designed for security and certified != secure

What the smartcards can be used for?

What problem is cryptographic smartcard solving?

- What problem is cryptographic smartcard solving?
  - Secure storage (keys and sensitive data)
  - Protected secrets even if physically attacked (tamper resistant)
  - Secure (cryptographic) computational device (signature, authentication)
  - Hardware root of trust (initial check of boot sequence)
  - Unspoofable logging
  - Enforcement of specific policy (PIN before sign, four eyes policy)
  - Easy to carry, easy to embed into another device, low battery usage

Which of these can’t be solved with laptop or mobile phone?

Applications

- SIM modules
  - key storage, session key derivation
  - GSM banking
  - PIN protection
- Bank payment card
  - cryptographic checksum on payment bill
  - offline PIN verification
  - contactless small payments
- Secure system authentication
  - Windows credential provider, Linux PAM modules
  - password storage only, challenge-response protocols
  - door access cards – mostly memory cards only

Application (cont.)

- Electronic identity cards (ePassports, eIDs)
  - contactless cards with Machine Readable Zone (MRZ)
  - secure messaging between reader and passport
  - active authentication - challenge-response with on-card key
- Multimedia distribution
  - Digital Rights Management (decryption keys, licenses)
  - pre-paid satellite TV (decryption keys)
- Secure storage and encryption/signing device
  - Cryptocurrency hardware wallets…

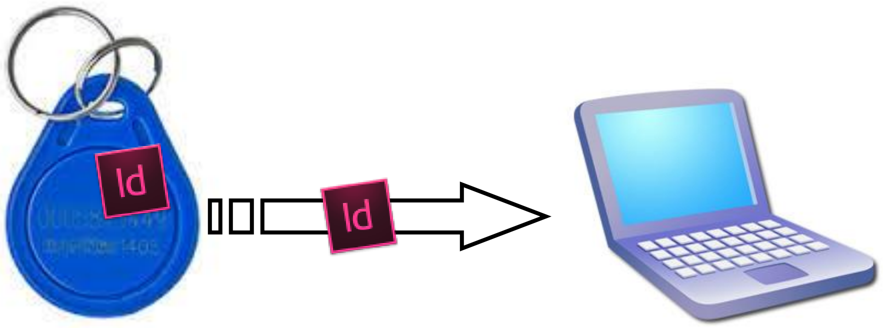
Application domains changes in time

- Cheap yet relatively hard to attack despite physical access
  - Sensitive data can be stored and used yet carried in pocket
  - Protection against the end-user (SIM, satellite decoders…)
- But we now have smartphones!
  - Payments via Apple Pay, Google Pay without physical smartcard
    - Still uses VISA/Mastercard payment infrastructure
  - Smartphones can make smartcards obsolete in large portion of previous usage domains!
- But smartphones are also quite too complex (=> bugs)
  - Sensitive data / keys etc. on smartphone are more vulnerable
- New use-cases
  - Trusted Platform Module (smartcard on the motherboard)
  - FIDO U2F tokens (improved authentication tokens)
  - Cryptocurrency hardware wallets (smartcard with trusted display)

MODES OF USAGE

Smart card carries fixed information

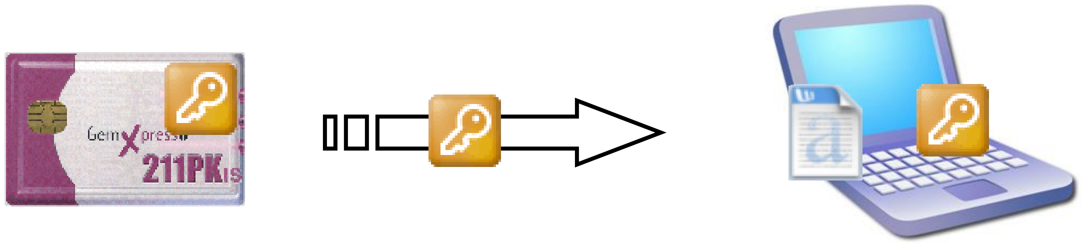
- Fixed information ID transmitted, no secure channel
- Low-cost solution (nothing “smart” needed)
- Problem: Attacker can eavesdrop and clone chip



Smart card as a secure carrier

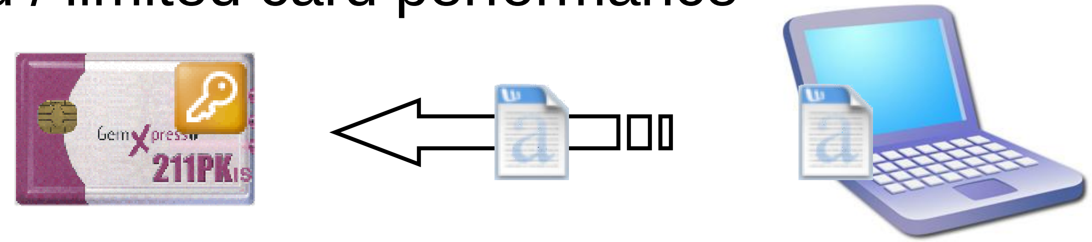
- Key(s) stored on a card, loaded to a PC before encryption/signing/authentication, then erased
- High speed usage of key possible (>>MB/sec)

- Attacker with an access to PC during operation will obtain the key
  - key protected for transport, but not during the usage



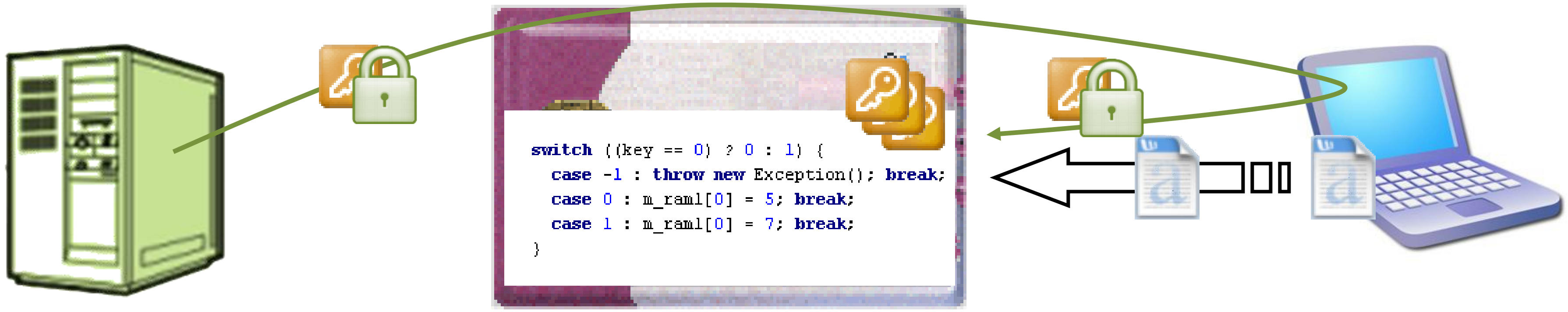
Smart card as encryption/signing device

- PC just sends data for encryption/signing…
- Key never leaves the card
  - personalized in secure environment
  - protected during transport and usage
- Attacker must attack the smart card
  - or wait until card is inserted and PIN entered!
- Low speed encryption (~kB/sec)
  - low communication speed / limited card performance



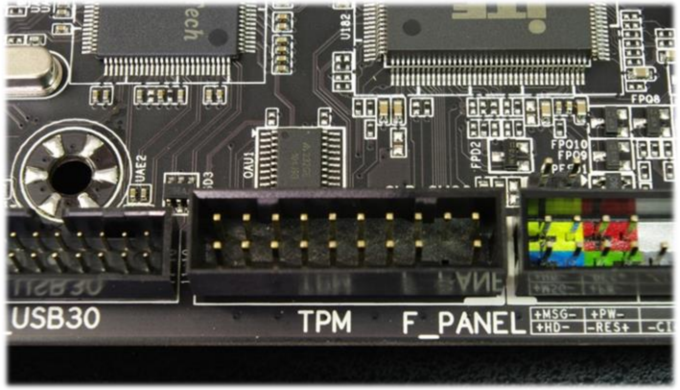
Smart card as computational device

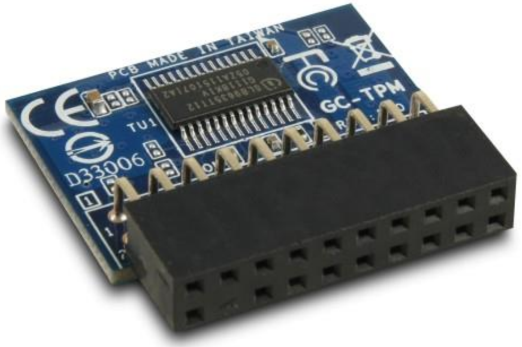
- PC just sends input for application on smart card
- Application code & keys never leave the card
  - card can perform complicated programmable actions
  - new code can be uploaded remotely
  - can open secure channels to other entity
    - secure server, trusted time service…
    - PC act as a transparent relay only (no access to data)
- Attacker must attack smart card or initial input



Smart card as root of trust (TPM)

- Secure boot process, remote attestation
- Smart card provides robust store with integrity
- Application can verify before pass control (measured boot)
- Computer can authenticate with remote entity…

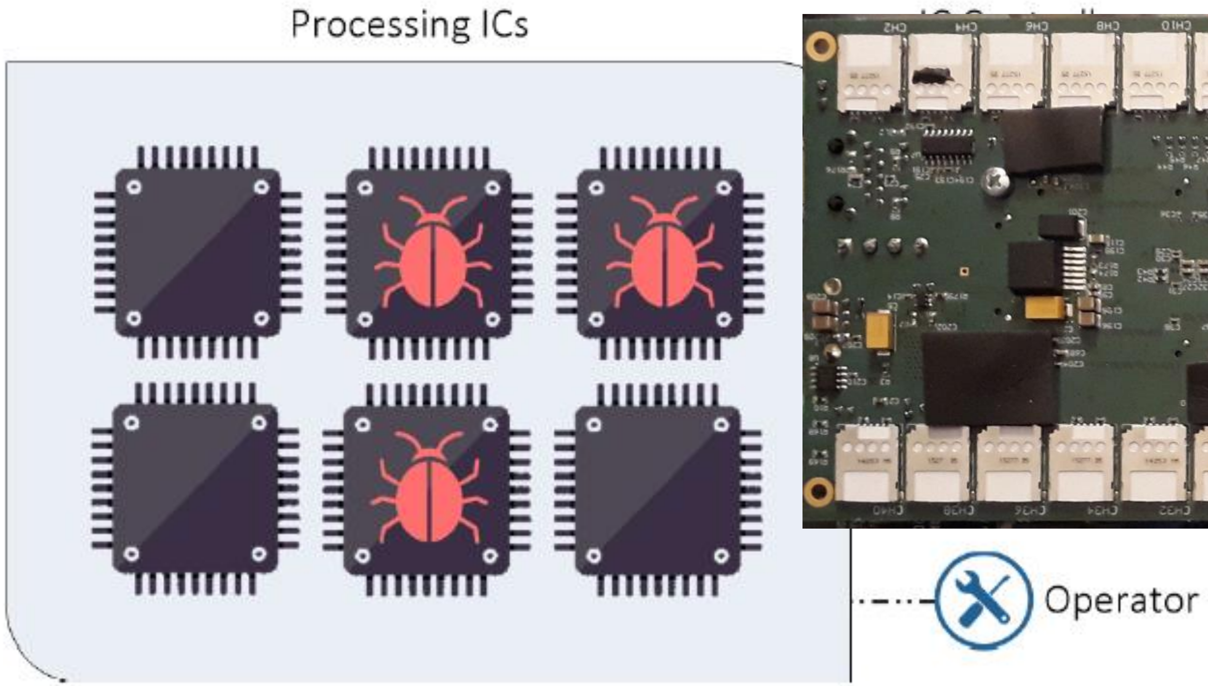


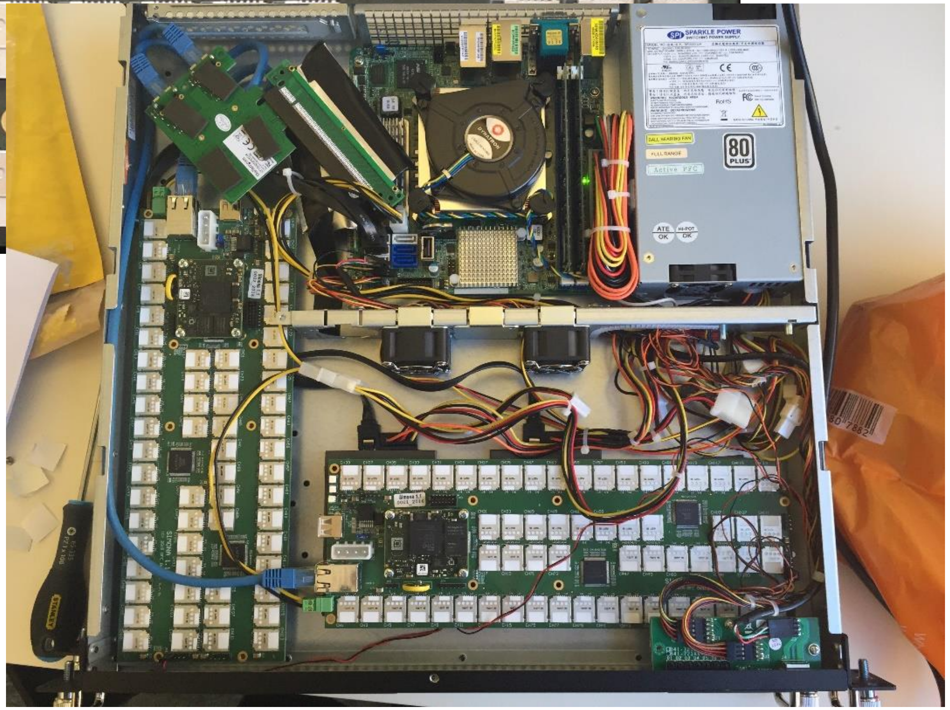


TPM : provided security functions

1. “Measured” boot with remote attestation
   – Provide signed log of what executed on platform (PCR)
2. Storage of keys (disk encryption, private keys…)
   – Can be additionally password protected
3. Binding and Sealing of data
   – Encryption key wrapped by concrete TPM’s public key
4. Platform integrity
   – Software will not start if current PCR value is not right

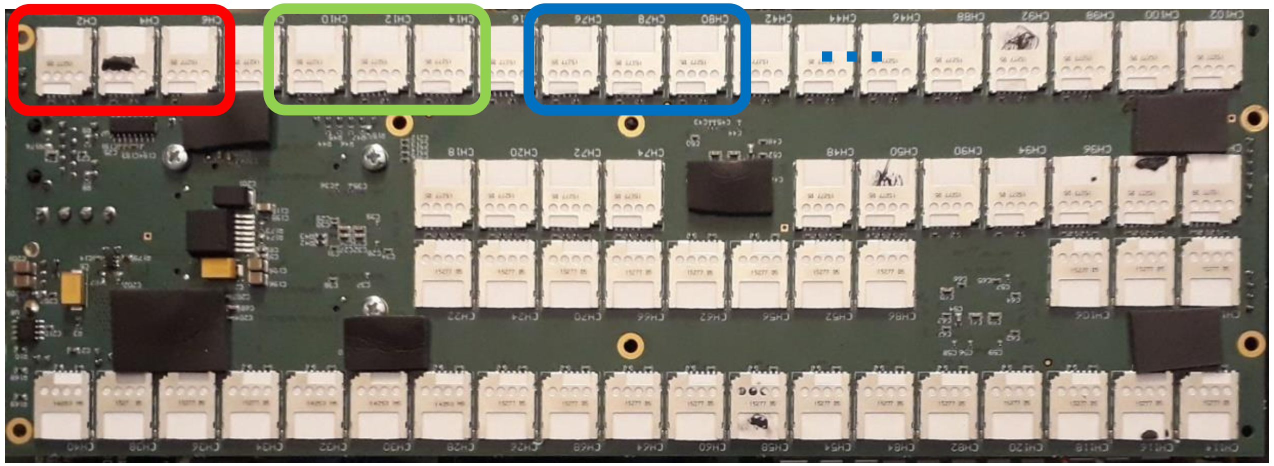
Myst: secure multiparty signatures





- High-speed, multi-tenant (120 cards)
- Robust against bugs, backdoors

SmartHSM for multiparty (120 smartcards, 3 cards/quorum)



120 cards => 40 quorums

=> 300+ decryptions / second

=> 80+ signatures / second

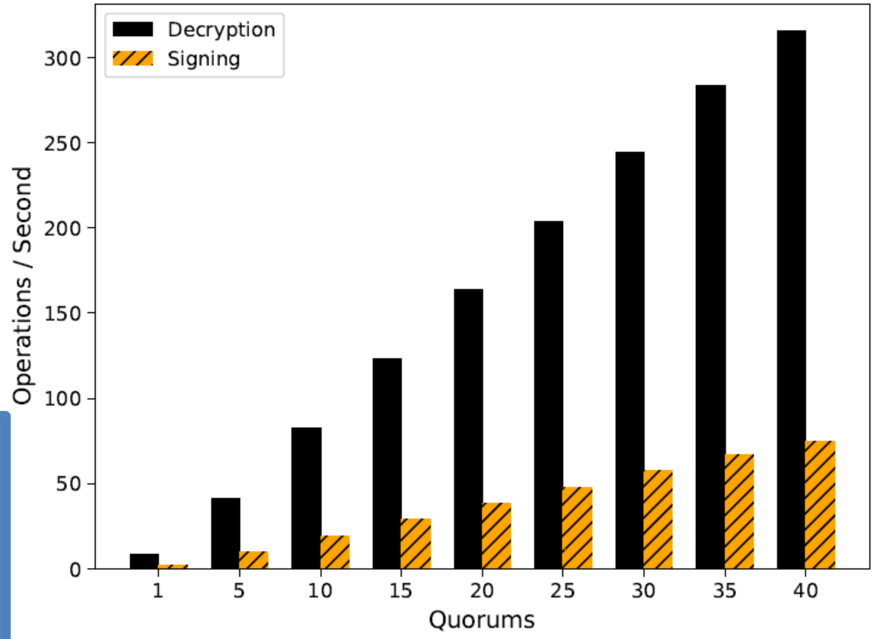


Figure 10: The average system throughput in relation to the number of quorums ($k = 3$) that serve requests simultaneously. The higher is better.

SMARTCARD ALGORITHMS AND PERFORMANCE

Common algorithms

- Basic - cryptographic co-processor
  - Truly random data generator
  - 3DES, AES128/256, (national algorithms)
  - MD5, SHA1, SHA-2 256/512
  - RSA (up to 2048b common, 4096 possible)
  - ECC (up to 256b common, 521b possible)
  - Diffie-Hellman key exchange (DH/ECDSA)
- Custom code running in secure environment
  - E.g., HMAC, OTP code, re-encryption
  - Might be significantly slower (e.g., SW AES 50x slower)

Cryptographic operations

- Supported algorithms (JCAlgTester, 100+ cards)
  - https://github.com/crocs-muni/JCAlgTest
  - https://www.fi.muni.cz/~xsvenda/jcsupport.html

javacard.security.MessageDigest	introduced in JavaCard version	c0	c1	c2	c3	c4	c5	c6	c7	c8	c9	c10	c11	c12	c13
ALG_SHA	<=2.1	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
ALG_MD5	<=2.1	no	yes	yes	yes	yes	yes	yes	no	yes	yes	yes	yes	yes	yes
ALG_RIPEMD160	<=2.1	no	no	no	yes	yes	yes	no	no	no	no	no	no	no	no
ALG_SHA_256	2.2.2	yes	no	no	suspicious yes	yes	no	no	yes	no	no	no	no	no	no
ALG_SHA_384	2.2.2	no	no	no	no	no	no	no	yes	no	no	no	no	no	no
ALG_SHA_512	2.2.2	no	no	no	no	no	no	no	yes	no	no	no	no	no	no
ALG_SHA_224	3.0.1	no	-	-	-	no	no	no	no	-	-	-	-	-	-
javacard.security.RandomData	**introduced in JavaCard version**	**c0**	**c1**	**c2**	**c3**	**c4**	**c5**	**c6**	**c7**	**c8**	**c9**	**c10**	**c11**	**c12**	**c13**
ALG_PSEUDO_RANDOM	<=2.1	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	no
ALG_SECURE_RANDOM	<=2.1	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
javacard.security.KeyBuilder	**introduced in JavaCard version**	**c0**	**c1**	**c2**	**c3**	**c4**	**c5**	**c6**	**c7**	**c8**	**c9**	**c10**	**c11**	**c12**	**c13**
TYPE_DES_TRANSIENT_RESET	<=2.1	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
TYPE_DES_TRANSIENT_DESELECT	<=2.1	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
TYPE_DES LENGTH_DES	<=2.1	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
TYPE_DES LENGTH_DES3_2KEY	<=2.1	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
TYPE_DES LENGTH_DES3_3KEY	<=2.1	yes	no	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
TYPE_AES_TRANSIENT_RESET	2.2.0	yes	no	suspicious yes	yes	yes	no	yes	yes	yes	yes	no	no	no	no

What is the typical performance?

- Hardware differ significantly
  - Clock multiplier, memory speed, crypto coprocessor…
- Typical speed of operation is:
  - Milliseconds (RNG, symmetric crypto, hash)
  - Tens of milliseconds (transfer data in/out)
  - Hundreds of millisecond (asymmetric crypto)
  - Seconds (RSA keypair generation)
- Operation may consists from multiple steps
  - Transmit data, prepare key, prepare engine, encrypt
    - $\rightarrow$ additional performance penalty
  - Usability rule of thumb: operation shall finish in 1-1.5sec

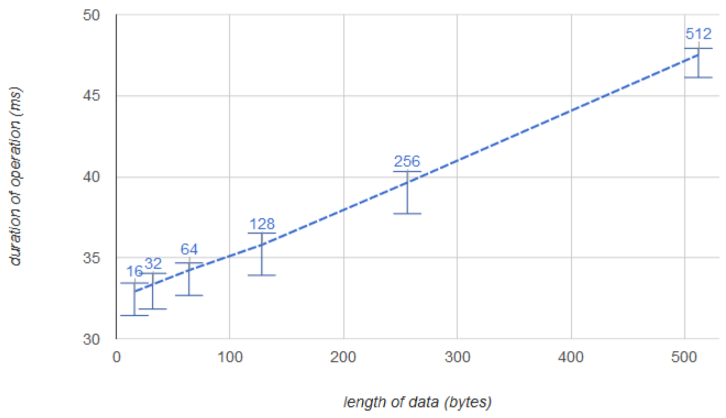
Performance tables for common cards

- Visit http://www.fi.muni.cz/~xsvenda/jcalgtest/

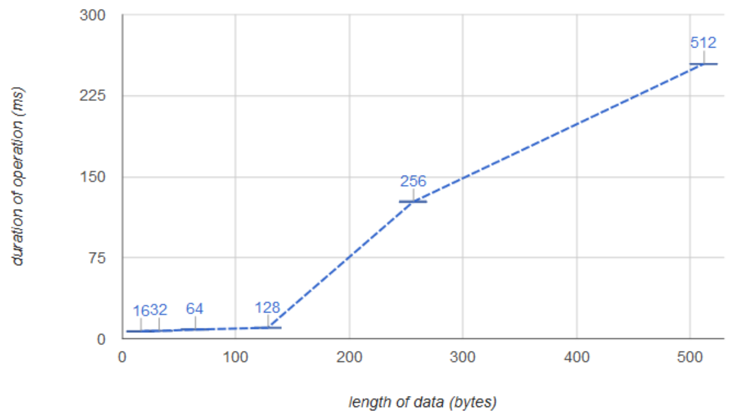
CARD/FUNCTION (ms/op)	SECURE RANDOM (256B)	SHA-1 hash (256B)	SHA2-256 hash (256B)	3DES encrypt (256B)	AES128 encrypt (256B)	AES256 encrypt (256B)	3DES setKey(192b)	AES setKey(128b)
Gemplus GXP R4 72K	2.45	3.69	-	53.71	26.05	31.52	9.4	9.28
NXP JCOP 31 V2.2 36K	6.92	19.84	-	7.27	-	-	26.1	-
NXP JCOP 21 V2.2 36K	7.28	20.91	-	7.68	-	-	25.84	-
NXP JCOP41 v2.2.1 72K	7.58	21.77	-	8.02	-	-	15.44	-
NXP J2D081 80K	10.4	11.73	21.18	7.1	6.73	7.66	20.12	16.31
NXP CJ3A081	13.8	11.45	21.05	12.8	10.33	11.35	11.04	10.9
NXP JCOP CJ2A081	14.14	11.9	22.46	13.3	10.78	11.81	5.39	5.22
NXP J2A080 80K	19.59	31.09	60.16	18.11	18.57	20.12	12.24	11.91
NXP JCOP31 v2.4.1 72K	20.97	34.1	66.02	19.95	20.44	22.24	6.7	6.38
NXP J3A080	21.64	35.78	69.32	20.92	21.41	23.2	15.48	12.28
Infineon CJTOP 80K INF SLJ 52GLA080AL M8.4	24.9	17.42	35.58	61.49	25.53	31.18	6.61	6.08
NXP JCOP21 v2.4.2R3	33.77	12.35	22.39	12.24	11.65	14.02	31.35	23.48
Oberthur ID-ONE Cosmo 64 RSA v5.4	52.49	23.53	-	16.05	-	-	25.31	-
G+D Smart Cafe Expert 4.x V2	322.91	33.66	-	37.19	-	-	3.59	-

Performance with variable data lengths

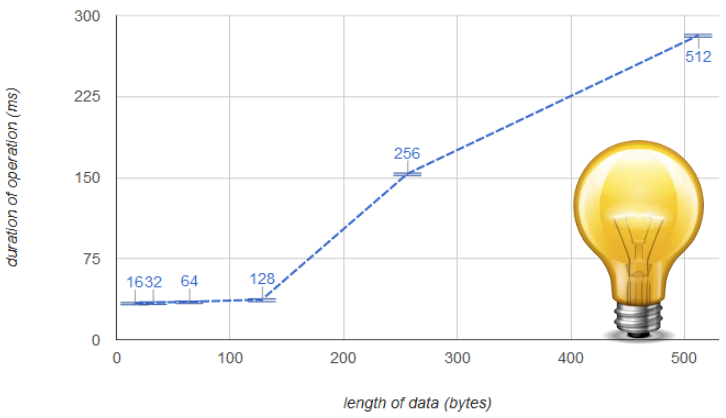
TYPE_DES LENGTH_DES ALG_DES_CBC_NOPAD Cipher_setKeyInitDoFinal()



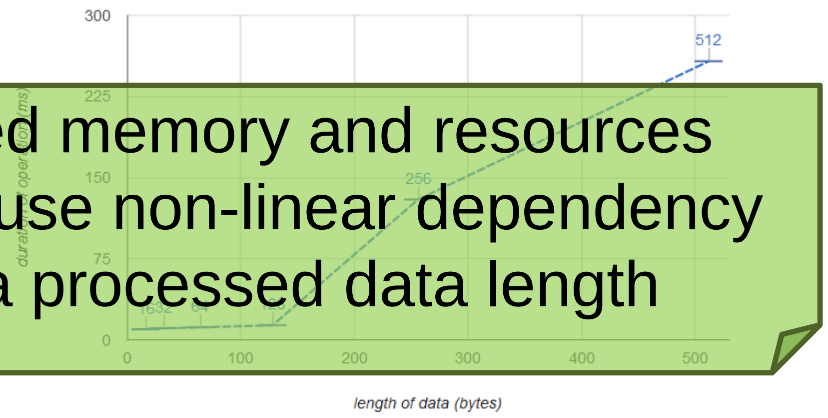
TYPE_DES LENGTH_DES ALG_DES_CBC_ISO9797_M1 Cipher_doFinal()



TYPE_DES LENGTH_DES ALG_DES_CBC_ISO9797_M1 Cipher_setKeyInitDoFinal()



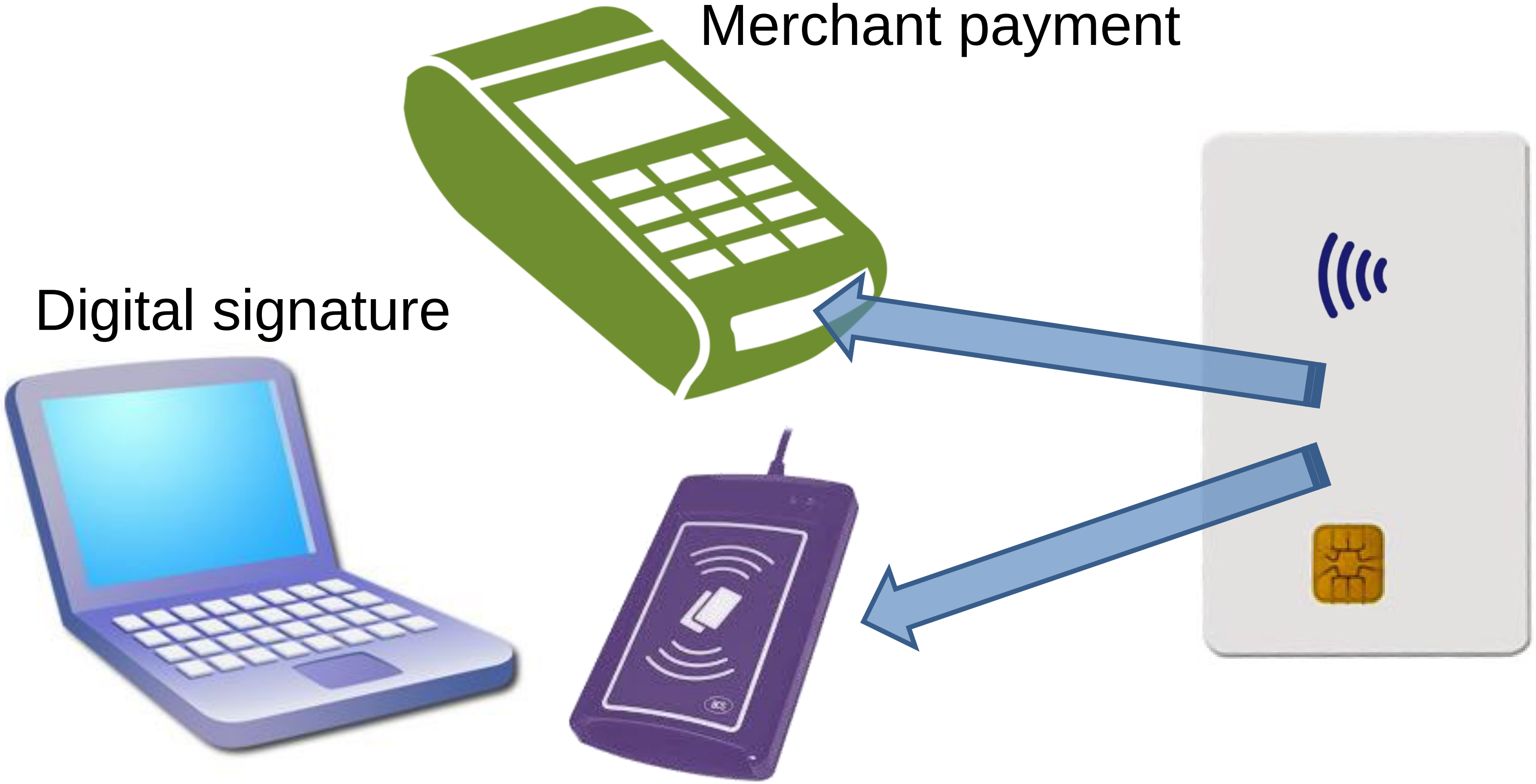
TYPE_DES LENGTH_DES ALG_DES_CBC_ISO9797_M2 Cipher_doFinal()



Limited memory and resources may cause non-linear dependency on a processed data length

Smartcards programming and use from programs

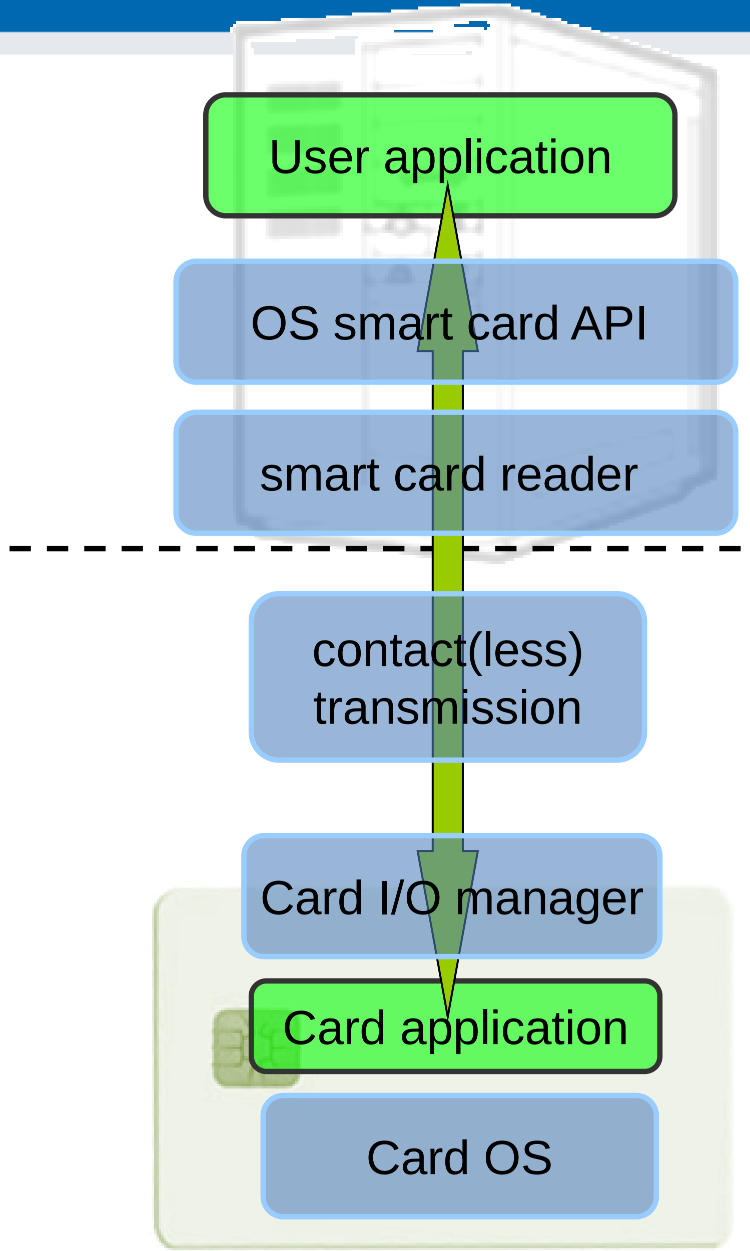
Big picture – terminal/reader and card



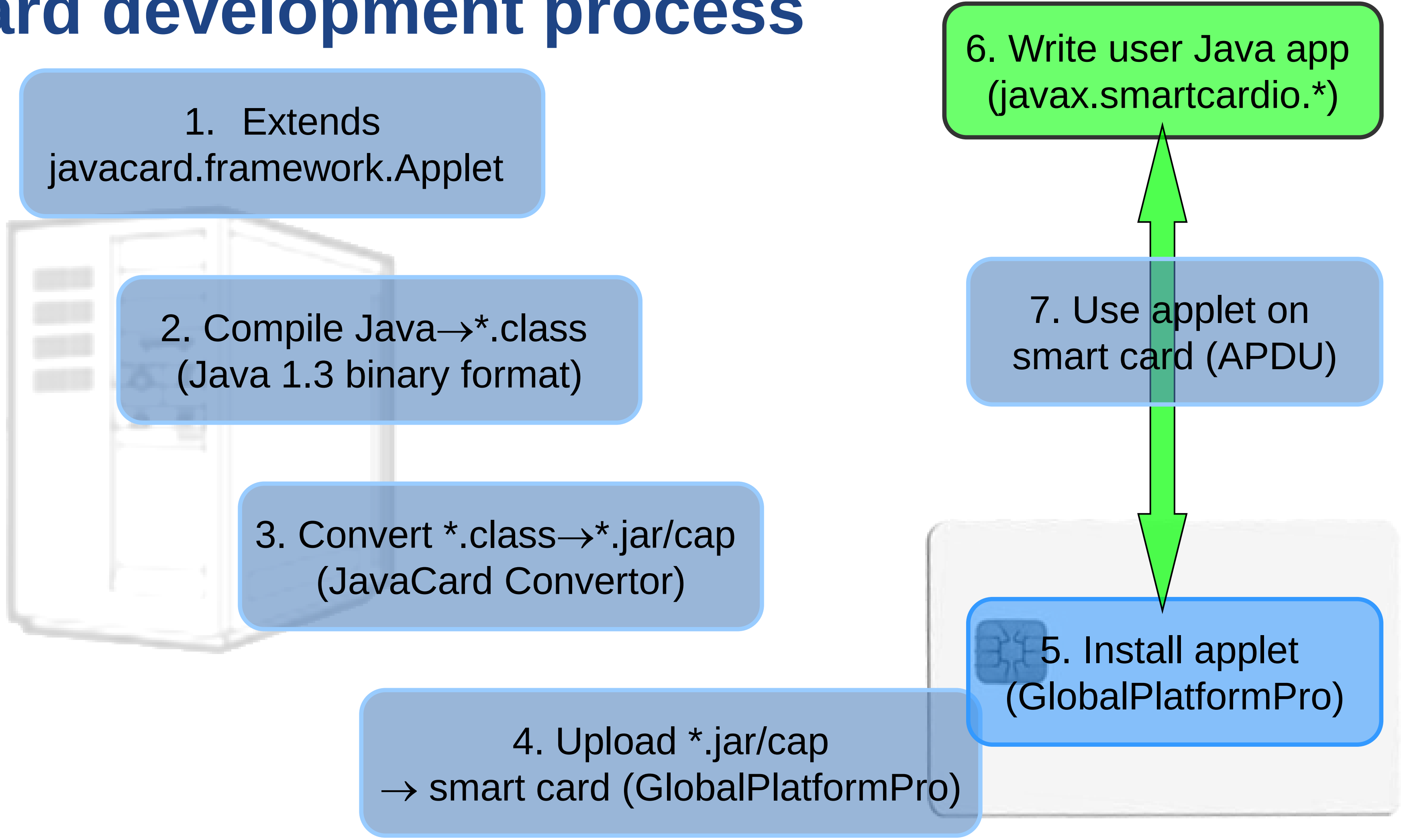
What principles and standards are used?

Big picture - components

- User application
  - Merchant terminal GUI
  - Banking transfer GUI
  - Browser TLS
  - …
- Card application
  - EMV applet for payments
  - SIM applet for GSM
  - OpenPGP applet for PGP
  - U2F applet for FIDO authentication
  - …



How to develop on-card application? JavaCard development process

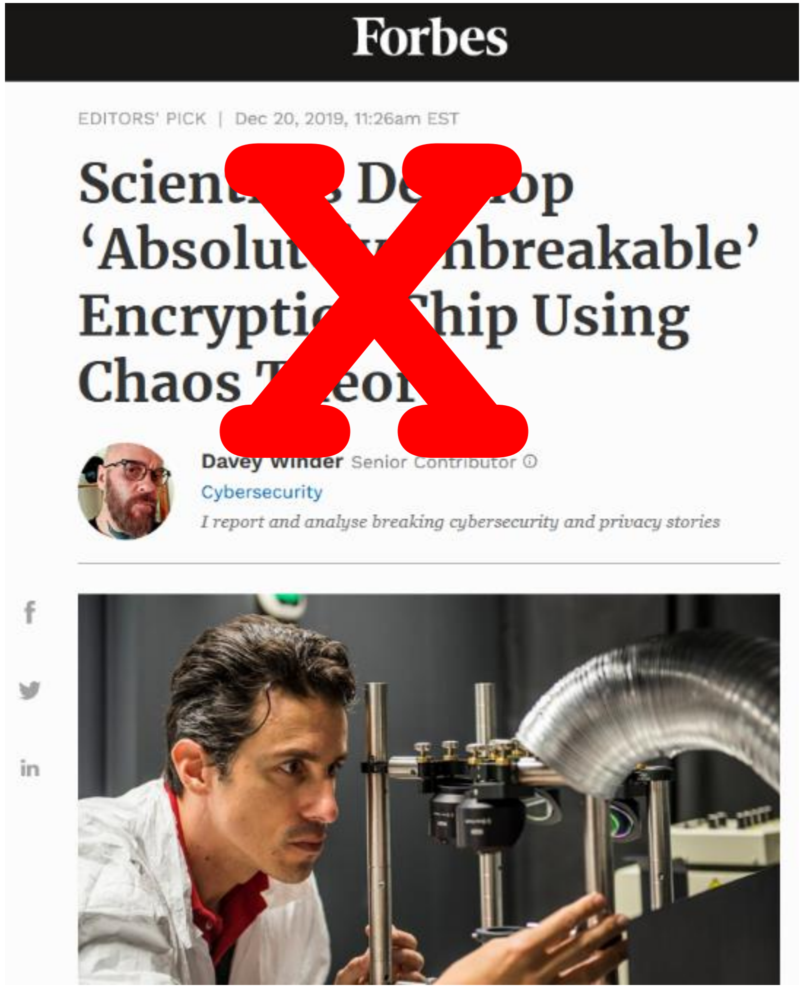


Pains for users/developers

- Closed-source, IP-heavy, NDA-based industry
- Primary users for manufactures/vendors are large customers
  - No interest in small / niche users (< 100k units)
  - Important API proprietary and/or not accessible (ARM TrustZone, proprietary JC packages, detailed specs…)
  - Supply chain issues (resellers, difficult to securely obtain card)
- What is open or available
  - Open API for applets (JavaCard API)
  - Open-source development toolchain for JavaCard
  - Common Criteria and FIPS140-2 certificates (but details omitted)
  - Results of reverse engineering

Smartcard security

- Invasive attacks
- Semi-invasive attacks
- Side-channel attacks
- Logical attacks



Attacks against smartcards

- "Secure hardware" != absolutely unbreakable hardware
  - Always depends on attacker motivation, knowledge, resources…
- The goal of security design is to increase the difficulty of attack
  - Higher than the value of data protected
  - Some attack harder to perform than other (equipment, time, knowledge, physical vs. remote access…)
  - Security is process (design, test, fix, repeat)
- Invasive attacks – physical dismantling of chip
  - E.g., read keys directly from physical memory
- Semi-invasive attacks – partial dismantling, chip still works
  - E.g., expose communication bus, read data by microprobe
- Side-channel attacks – unintended leakage of physical device
  - correlated with the secret data processed (keys)
  - E.g., power consumption analysis, timing attack
- Logical attacks - exploits logical flaw in code running inside chip

Focus of this lecture

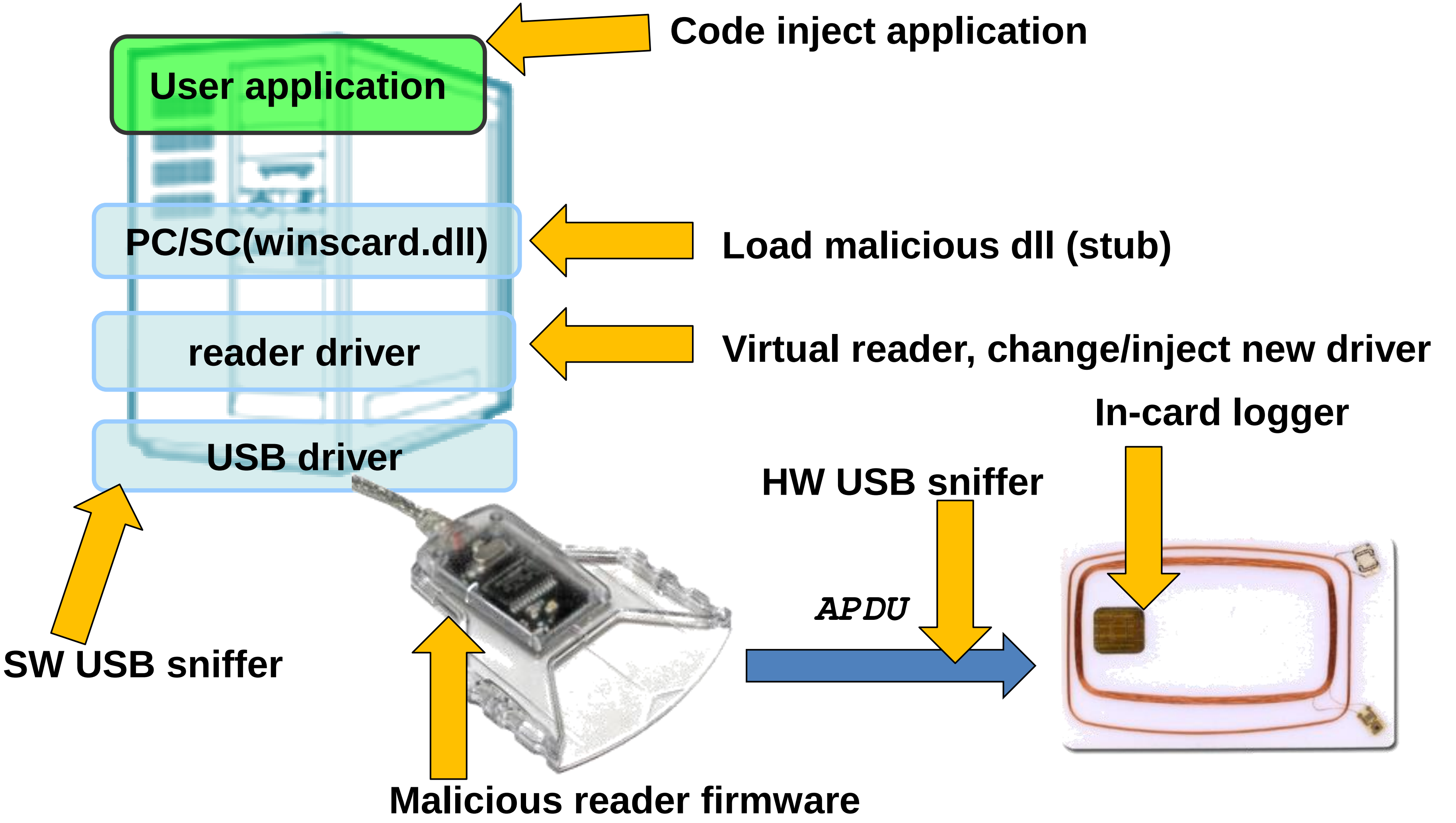
Discussion – attacking smartcard-based solutions

- Scenario: attack Brno transport ticket card
  - Contactless communication, pre-registered EMV-based card
- Scenario: attack Bitcoin hardware wallet
  - Private key derived, then used to sign transaction (inputs, outputs, amounts)
- Scenario: attack contactless EMV payment card
  - Pay at merchant terminal

Application attacks

- Focus on logical attacks possible by “malware”
  - No physical access to target card is assumed, remote attacks
  - Man-in-the middle attacks
  - Redirection of traffic, remote smart card access
- Target applications
  - Banking app (login, transaction authorization)
  - Resources protected by two-factor authentication (VPNs…)
  - DRM applications (user is attacker)
  - Citizen ID cards (ID theft)
  - …

Where to log/manipulate communication?

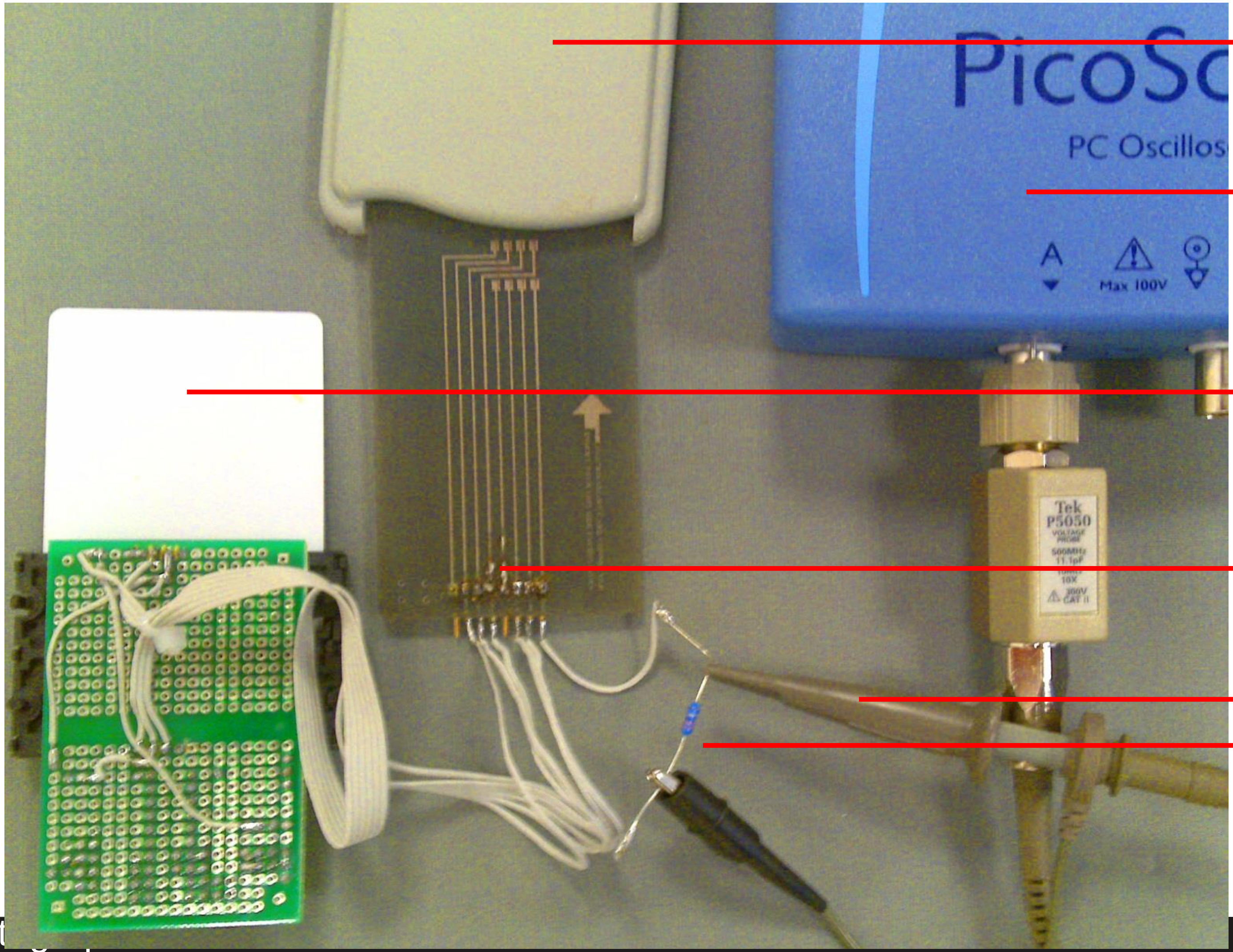


Power analysis

- External power supply - no battery on SC
- Power consumption depends on actual ops/data
- Voltage variation measured using digital oscilloscope and small resistor

- Real threat – and not only for smart cards
  - Mifare DESfire
  - KeeLoq
  - Xilinx bitstream

Power analysis – basic setup



Smart card reader

Oscilloscope

Smart card

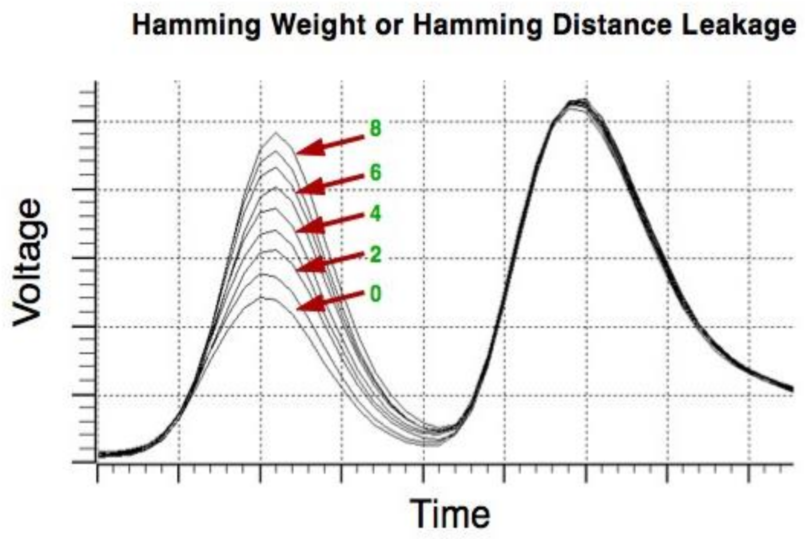
Inverse card connector

Probe

Resistor 20-80 ohm

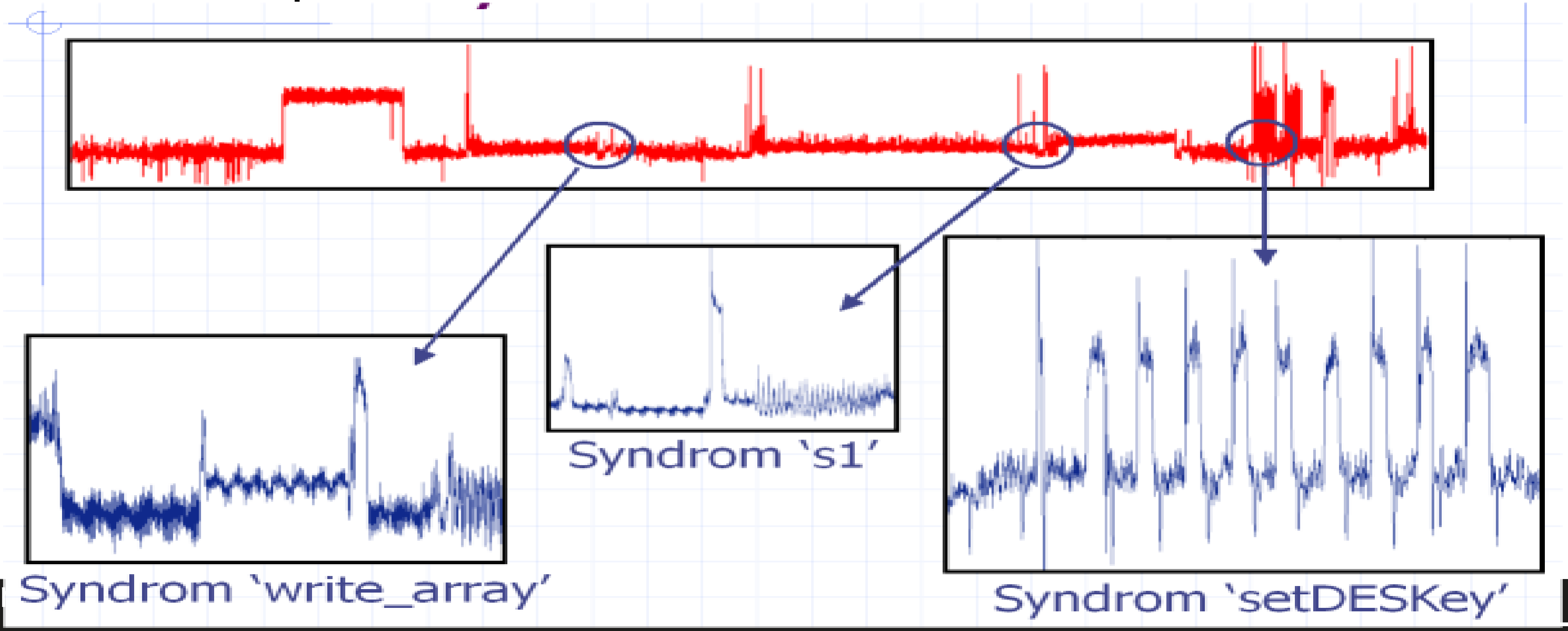
Simple power analysis

- Direct processing of single power trace
  - operations => reverse engineering
  - data => additional information about secret keys
    - hamming weight of separate bytes of key (2^{56}-> 2^{38})
- Averaging over multiple traces to reduce noise
- Exact implementation must be known
  - position of instruction
  - obtained by reverse engineering

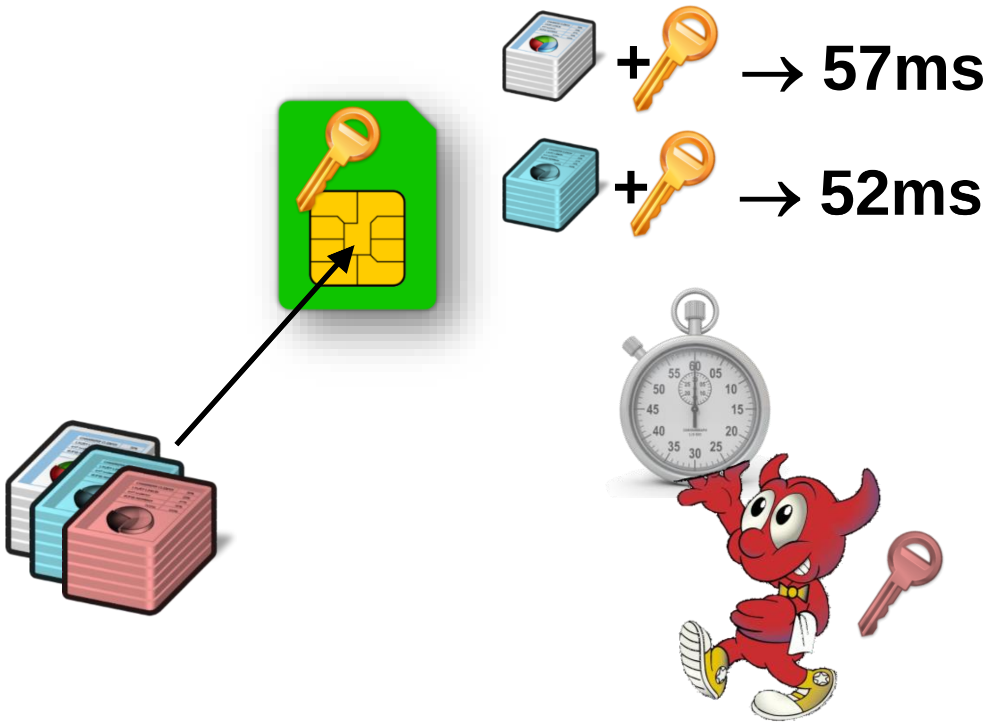


Reverse engineering – operation level

- Semi-automatic recognition of operations
  - from typical power consumption patterns
  - database of corresponding operation and pattern
- Often easier than obtain processed data



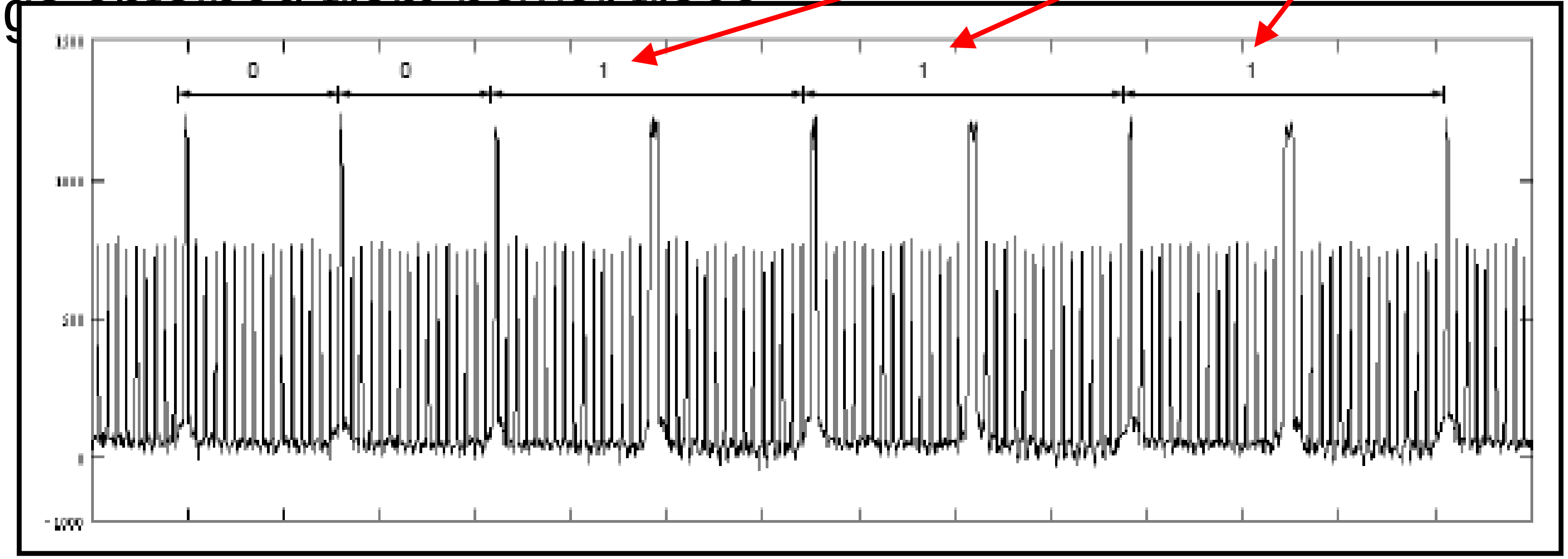
Timing (side-channel leakage) attack



Timing analysis

- Length of operation depends on processed data
  - due to speed optimization (limited resources)
  - due to un-aware algorithm design
  - e.g. Montgomery ladder
- Timings obtained from power trace

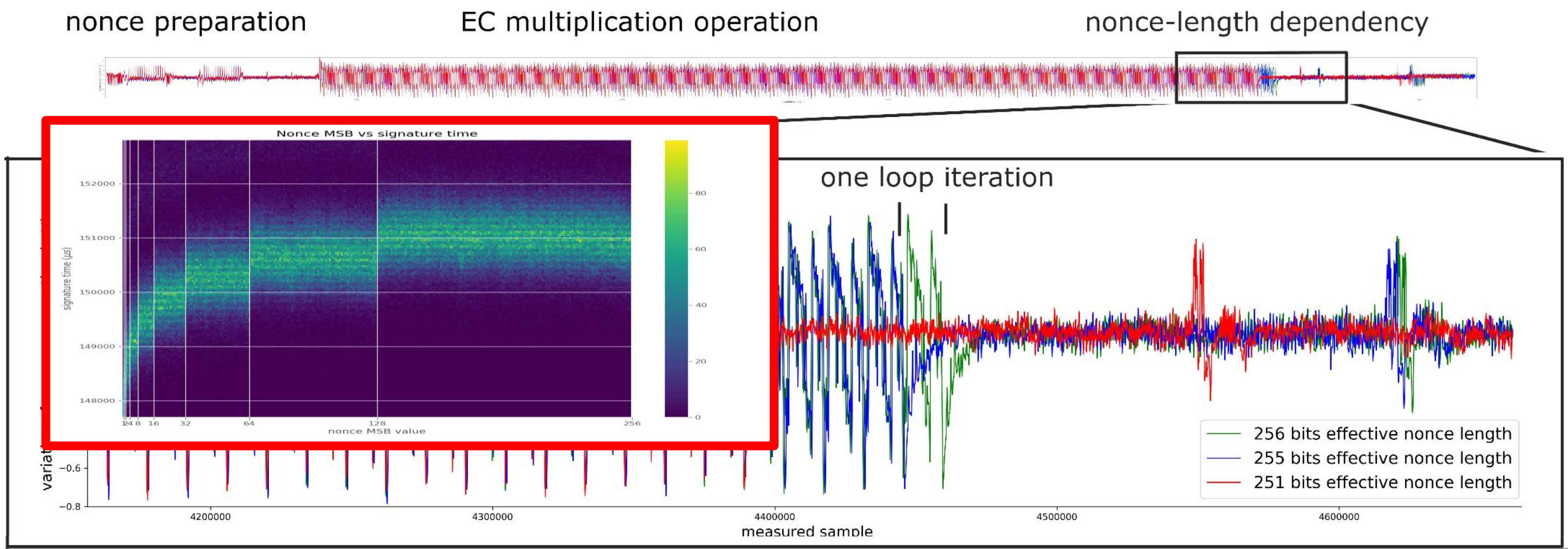
$x = m$
FOR $i = n-2$ DOWNTO 0
  $x = x^2$
  IF $(k_j == 1)$ THEN
    $x = x \cdot m$
ENDFOR
RETURN x



Minerva vulnerability (10/2019)

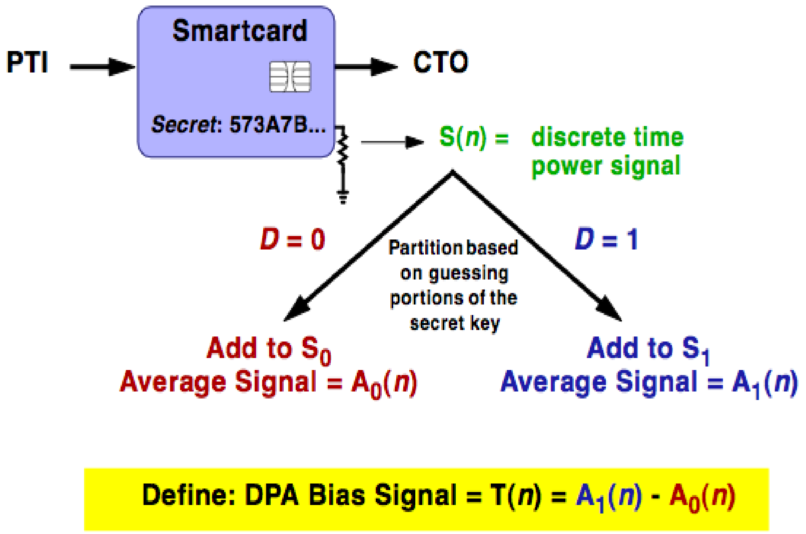
https://minerva.crocs.fi.muni.cz/

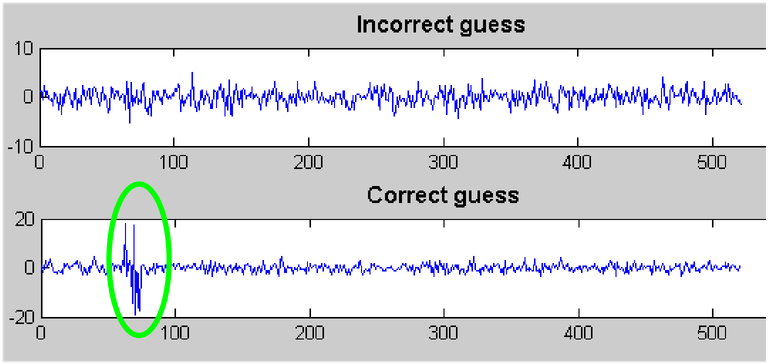
- Length of ECDSA nonce leaked
  - shorter nonce => shorter signature time
- Enough to extract whole ECC private key in 20-30 min
- Athena IDProtect smartcard (EAL 4+), Libgcrypt, SunEC/OpenJDK/Oracle JDK…



Differential power analysis

- Powerful attack on secret values
  - e.g. encryption keys
- Multiple power traces with key usage
  - 10^3-10^5 traces with known I/O data
  - KEY ⊕ KNOWN_DATA
- Key is guessed byte-per-byte
  - correct guess reveals correlation with traces
  - all possible values of single byte tried (256)
  - traces divided into 2 groups
  - groups are averaged
  - averaged signals are compared
  - significant peaks if correct
- No need to know exact implementation
  - big advantage





Conclusions

- SC massively deployed ($20*10^9$), mainly w.r.t. security
  - wide range of usage (banking, SIM, access control)
  - secure storage (encryption/signature keys)
    - on-card asymmetric key generation!
  - secure code execution
  - interesting protocols involving smart cards
- Limited memory (10^2 kB) and CPU power (8-32b,5-50MHz)
  - Low-cost small computer designed specifically for security
  - crypto operation accelerated by co-processors
- Still can be attacked
  - typically need for special knowledge and/or equipment
  - still far more secure than standard PC