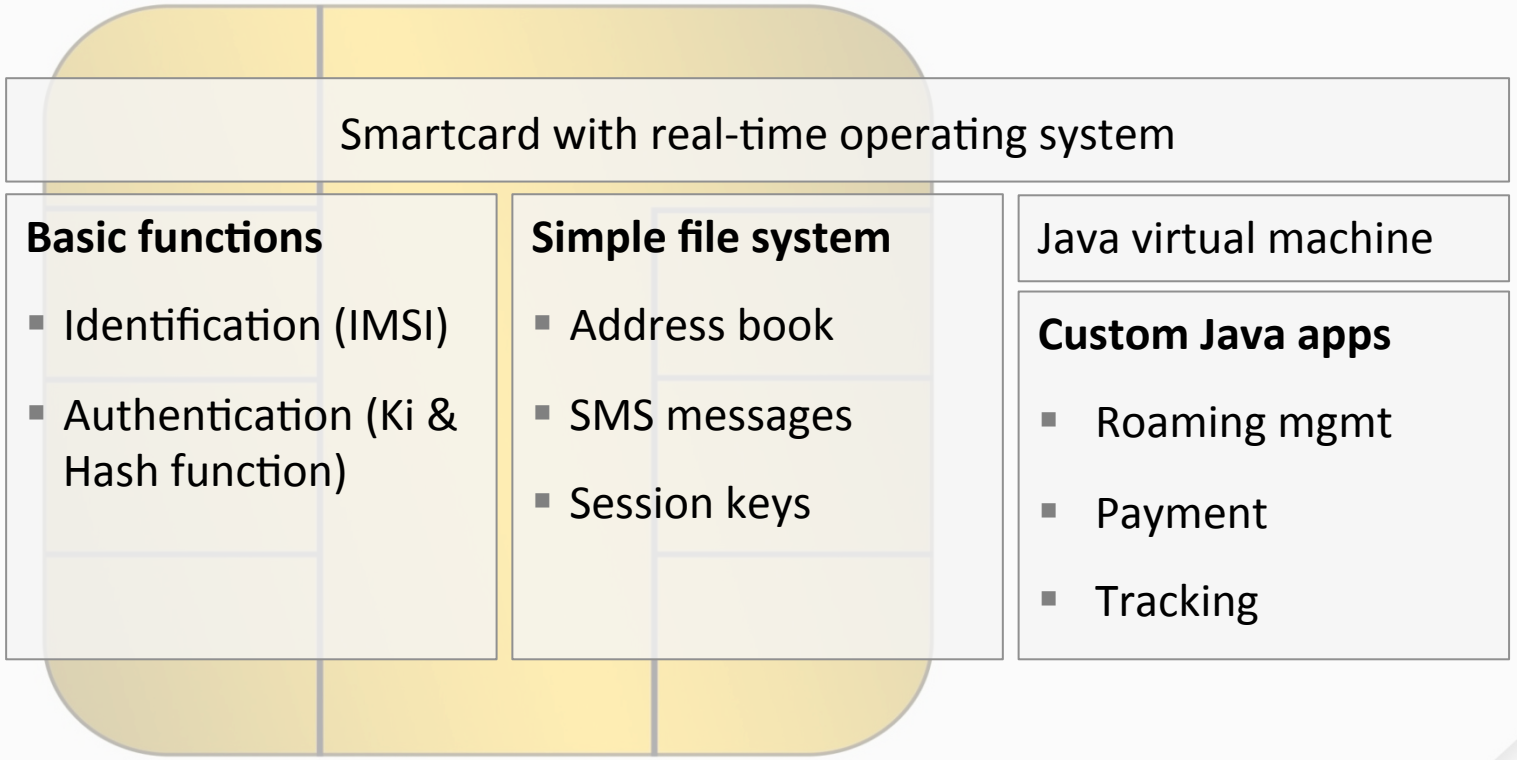
Rooting SIM cards

The SRLabs Team

SIM cards are fully programmable computer systems

Applications on modern SIM card



SIM security involves many layers from smartcards to cryptography and Java process separation

SIM card includes various protection mechanisms

User authentication

by simple comparison

PIN/PUK numbers

SIM authentication

by cryptographic hash function (often Comp128 in GSM; Milenage in 3G/4G)

Ki

A Secure Java deployment

using DES/3DES/AES signature + encryption

OTA keys

Storage protection

through proprietary smartcard security mechanisms

B Application separation: Java VM sand boxing

Individual protection logic for banking applets, identification applets, etc.

…

Java crypto API: DES/3DES/AES; sometimes RSA

Agenda

SIM card background

A **Getting on to the SIM**

B Stealing SIM secrets

OTA security level is chosen by server while SIM enforces mandatory minimum level

ILLUSTRATIVE

OTA server initiates remote transaction

Binary SMS communication

Target app / key set #

Command – possibly encrypted and/or signed | Used security level | Requested security level

Response protected according to request, but not below minimum level stored on card

SIM card stores multiple key sets, possibly with different protection levels

Key set 3

Key set 2

Key set 1

	DES	3DES	AES	Mandatory
Encryption	◉	○	○	☐
Signature	◉	○	○	☑

OTA error handling is underspecified, possibly opening attack surface

Binary SMS communication

Attacker probes cards to gain material for DES key cracking

Command with wrong signature | Use: DES signature | Request: DES signature

Response to mal-signed request differs by card type

- a. (25%* of cards) — (No response)
- b. (50%*) — Error message | Sometimes with all-zeros signatures
- c. (25%*) — Error message | DES signature

Data useable for key cracking

SIM card with DES key (prevalence of DES keys varies between operators; can be up to 100%)

* Estimated from relatively small and geographically skewed measurement set

OTA DES do not withstand key cracking

Challenge: Derive 56 bit DES key from OTA response signature

Cracking strategies	Investment	Cracking time
Be patient Brute force on GPU	EUR 1.000	6 months
Throw money at it Brute force on FPGA cluster	EUR 50.000	1 day
Ride the rainbow Time-memory trade-off using large hard disks & GPU	EUR 1.500 + 1 year pre-computation	1 minute (but <100% success rate)

Only possible when OTA response is fully predictable

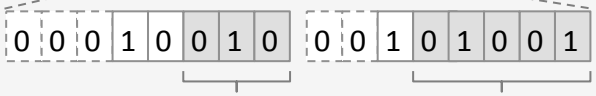
Attacker SMS asks for DES-signed SMS response with fully predictable content

Attack-specific features

Command packet
is sent by the attacker to provoke response

UDHI	PID	DCS	UDH	CPL	CHL	SPI	KIc	KID	TAR	CNTR	PCNTR	CC	Data
1	127	246	027000	Packet length	Header length	▪ No ciphering ▪ Sign ▪ PoR request	No cipher	DES signature	App	01	Padding counter	Rand. invalid	Generic command

Packet details:



- ▪ No ciphering
- ▪ Cryptographic checksum

- ▪ Do not cipher PoR
- ▪ Sign PoR
- ▪ Send PoR in any case

Response packet
may offer attack surface

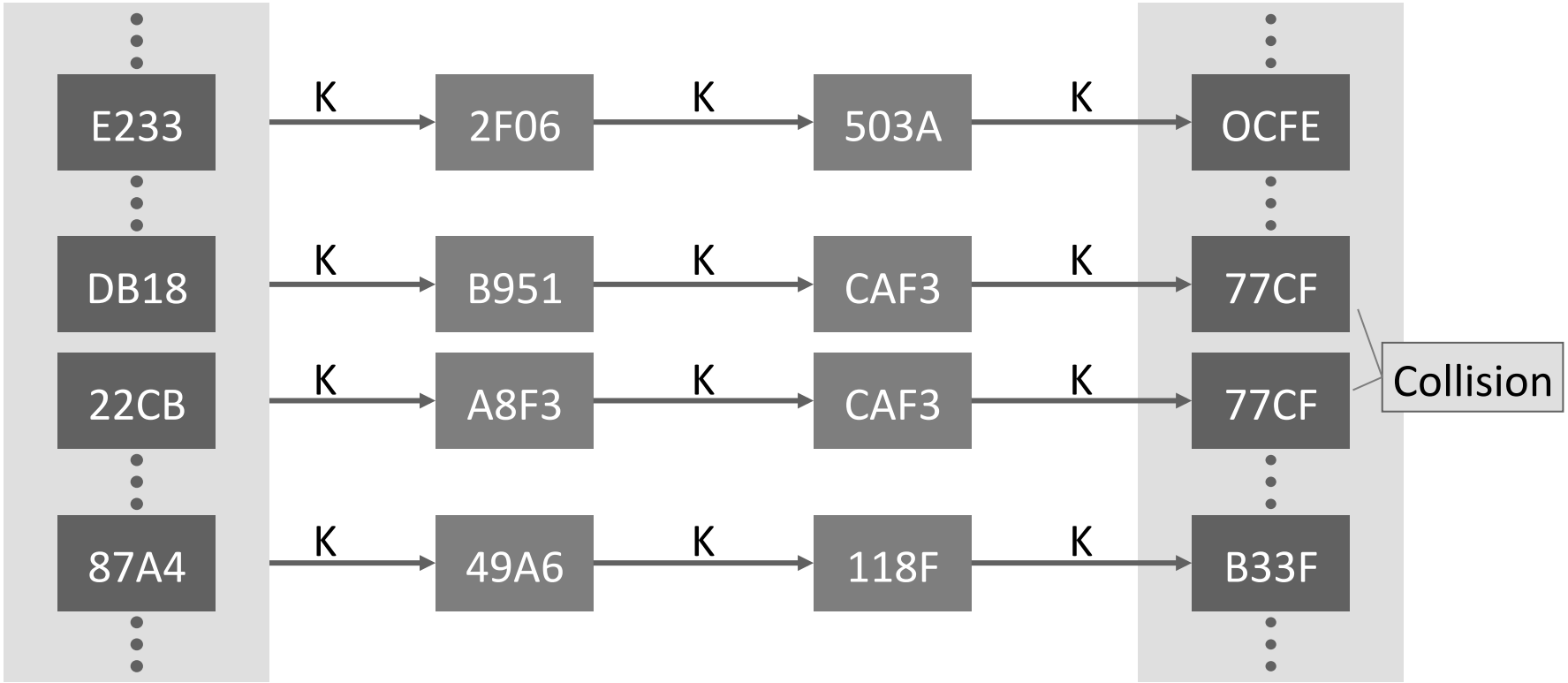
UDH	RPL	RHL	TAR	CNTR	PCNTR	Status Code	CC	Data
027100	Packet length	Header length	App	01	Padding counter	Status Code	Crypto-Checksum	Response

–or –

No response

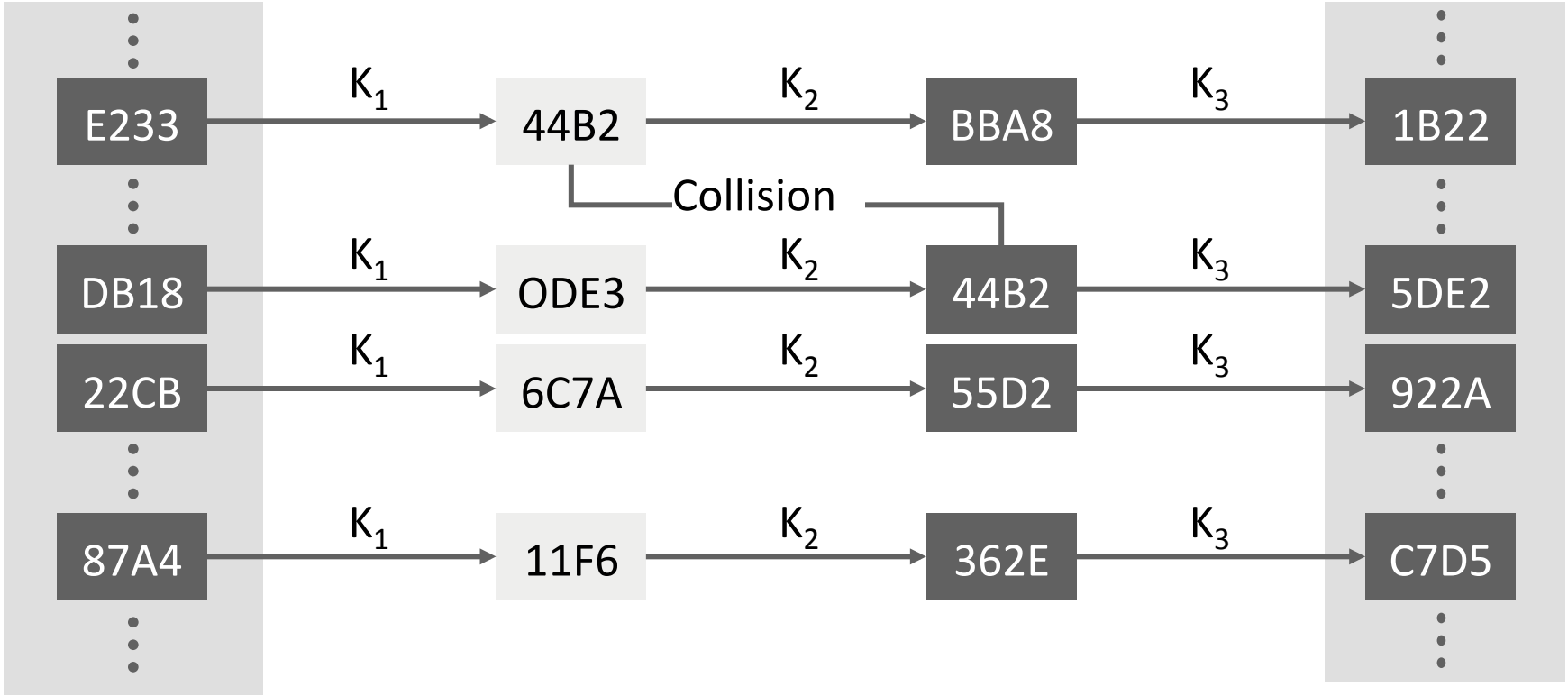
Signature over predictable data useable for rainbow table key cracking

Pre-computation tables store DES code book in condensed form



The uncondensed code book is petabytes in size. Tables provide a **trade-off:** Longer chains := less storage, but also longer attack time

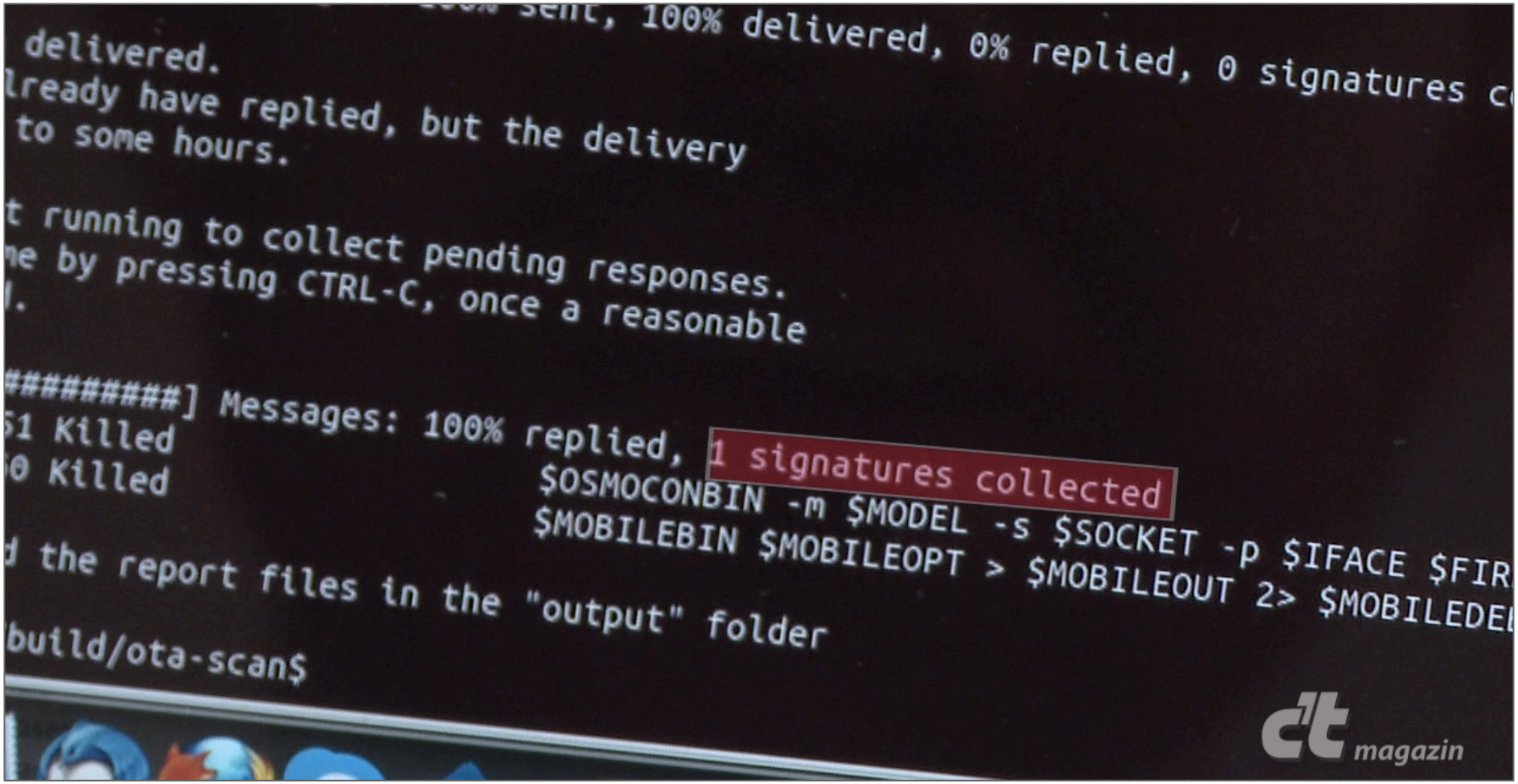
Table optimization: Rainbow tables mitigate the effect of collisions



Rainbow tables have no mergers, but an exponentially higher attack time

Source:c't

Video and live demo: Remotely cracking OTA key



Video source: http://www.heise.de/security/artikel/DES-Hack-exponiert-Millionen-SIM-Karten-1920898.html

OTA attacks extend beyond DES signatures

Many mobile operators responded to the looming SIM hacking risk considerate and faster than we could have wished for others (too?) quickly concluded they were not affected

Recent operator statements

We use encryption instead of signatures; the attack does not apply here

We don't even use OTA

We only use 3DES

Does it make sense?

- **No.** Encrypting a known plaintext with DES is as bad as signing it. Even when both are required, the attack still applies (but needs two rainbow tables)

- **No.** virtually all SIMs are Java cards. Even if you are not using those capabilities, an attacker may (and will probably find that you never cared to update the keys of this virtual waste land)

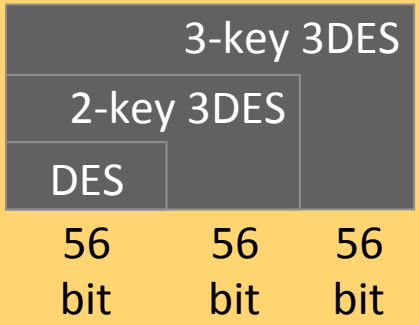
- **Maybe.** 3DES is good, but have you …
  - … made sure to use full entropy 112/168 bit keys instead of multiple copies of a 56 bit key?
  - … changed all the standard keys?
  - … heard of *downgrade attacks*?

For some cards, even 3DES keys are crackable

Downgrade attack flow

Attacker		
	Command	Request DES-signed response (KID = 1)
Crack first third of key	Error	DES-signed
	Command	Request 2-key 3DES response (KID = 5)
Crack second third*	Error	2-key 3DES-signed
	Command	Request 3-key 3DES response (KID = 9)
Crack final third*	Error	3-key 3DES-signed

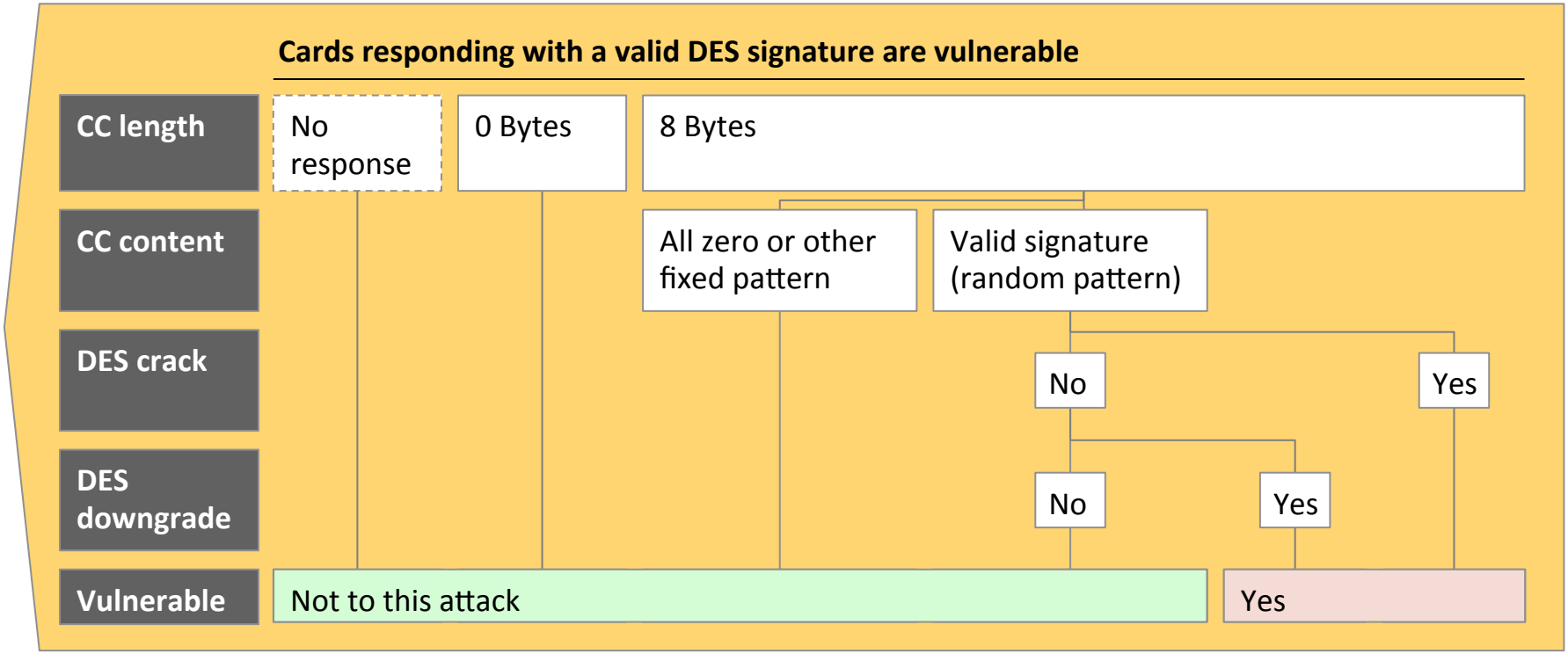
Some **SIM cards with 3DES key** use lower signature schemes when requested (in violation of the standard)



* Must be brute-forced; Rainbow table attack no longer possible

Only some cards provide crackable plaintext-signature pairs

Attacker sends **OTA command** with wrong DES signature requesting DES-signed response



Agenda

SIM card background

A Getting on to the SIM

B **Stealing SIM secrets**

Java virus does not automatically have access to all SIM assets

OTA-deployed SIM virus can access SIM Toolkit API

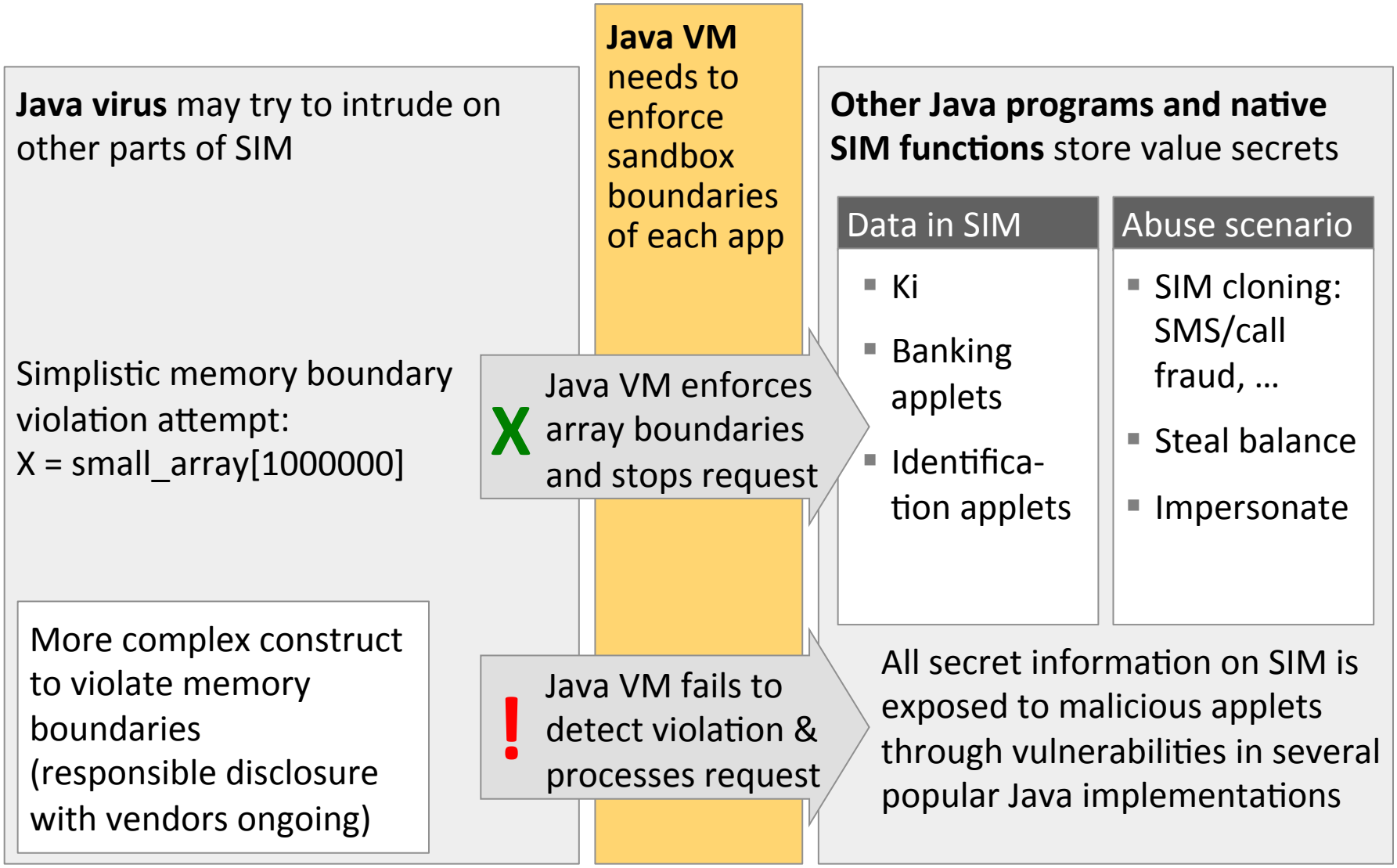
Standard STK function	Abuse potential
Send SMS	▪ Premium SMS fraud
Dial phone numbers, send DTMF tones	▪ Circumvent caller-ID checks ▪ Mess with voice mail
Send USSD numbers	▪ Redirect incoming calls; sometimes also SMS ▪ Abuse USSD-based payment schemes
Query phone location and settings	▪ Track victim
Open URL in phone browser	▪ Phishing ▪ Malware deployment to phone ▪ Any other browser-based attack

Java sand box should protect critical data on SIM

Data access on SIM would enable further abuse

Protected function	Abuse potential
Read Ki	▪ SIM cloning ▪ Decrypt all 2G/3G/4G traffic
Read OTA keys	▪ Lateral attacks
Read Java processes	▪ Clone NFC payment takers and other future SIM applications
Write to Flash or EEPROM	▪ Alter OS to prevent vulnerability patching

Java VM on many SIMs fails to confine malicious applets



Putting it all together – Remote SIM cloning

	A Infiltrate card with malicious Java applet		B Exfiltrate valuable data
Action	Send binary SMS with OTA command to card, requesting card response	▷ Crack DES signing key, then sign Java virus & send through binary SMS	▷ Leverage gaps in Java VM memory separation to access arbitrary SIM card data
Result	Card may respond with a DES- signed error message	Card installs and executes signed Java applet	Malicious applet extracts Ki, banking applets, etc., and send to attacker via SMS

Wide-scale SIM hacking risk must be mitigated on several layers

Mitigation layer for OTA hacking risk	Effectiveness	Cost	
Filter OTA messages from unapproved sources	Prevents probing in home network; leaves SIMs exposed when roaming, to fake base stations, and to phone malware	Functionality readily available in most SMSCs	Network operators short-term mitigation option
Deactivate OTA on card	Prevents attack (but also any future use of OTA w/ DES key)	Can be done through SMS	
Use 3DES or AES OTA keys	Prevents attack (expect for where downgrade attack works)	Some cards need replacing, others updates	Network operators mid-term mitigation option
Use cards that do not disclose crypto texts	Prevents the attack	Some cards need to be replaced	
Filter suspicious messages on phone base band	Prevents the attack	New software function for future phones	Complimentary mitigation option for phone manufacturers

Industry response was encouraging for responsibly disclosing hacking research

The **responsible disclosure** went surprisingly well and is worth mentioning

- We disclosed several months ahead of the release to trusted contracts made around previous releases
- Experts from a few large companies verified the results and created best practice responses
- Industry associations disseminated guidelines to all other operators
- Many networks are now well underway implementing filtering and reconfiguring cards
- Only a single lawyer stumbled into the interaction, but quickly left

Take aways from a number of responsible disclosure that all went well (except for one)

- Find **constructive partners** in the industry; ask other hackers for their recommendations
- **Disclose early** and don't be surprised if even the most motivated disclosure partner takes months to distribute the information confidentially in their industry
- Bring someone with **disclosure experience** to meetings
- **Expect friendliness** and remind your partner of the required etiquette should they ever act rude or arrogant
- Help your technical contacts win the internal battles: **Refuse** to speak to their **lawyers;** never sign an **NDA** prior to your disclosure
- Be extremely careful accepting money; and only ever to help with mitigation

Take aways

A Some DES-secured SIM-cards allow for remote key cracking and applet installation

B Java vulnerabilities enable attacker to remotely extract Ki, banking applet data

- Mitigation options exist on network, baseband, and SIM card level

Questions?

Karsten Nohl <nohl@srlabs.de>