Crash and Pay: Owning and Cloning Payment Devices

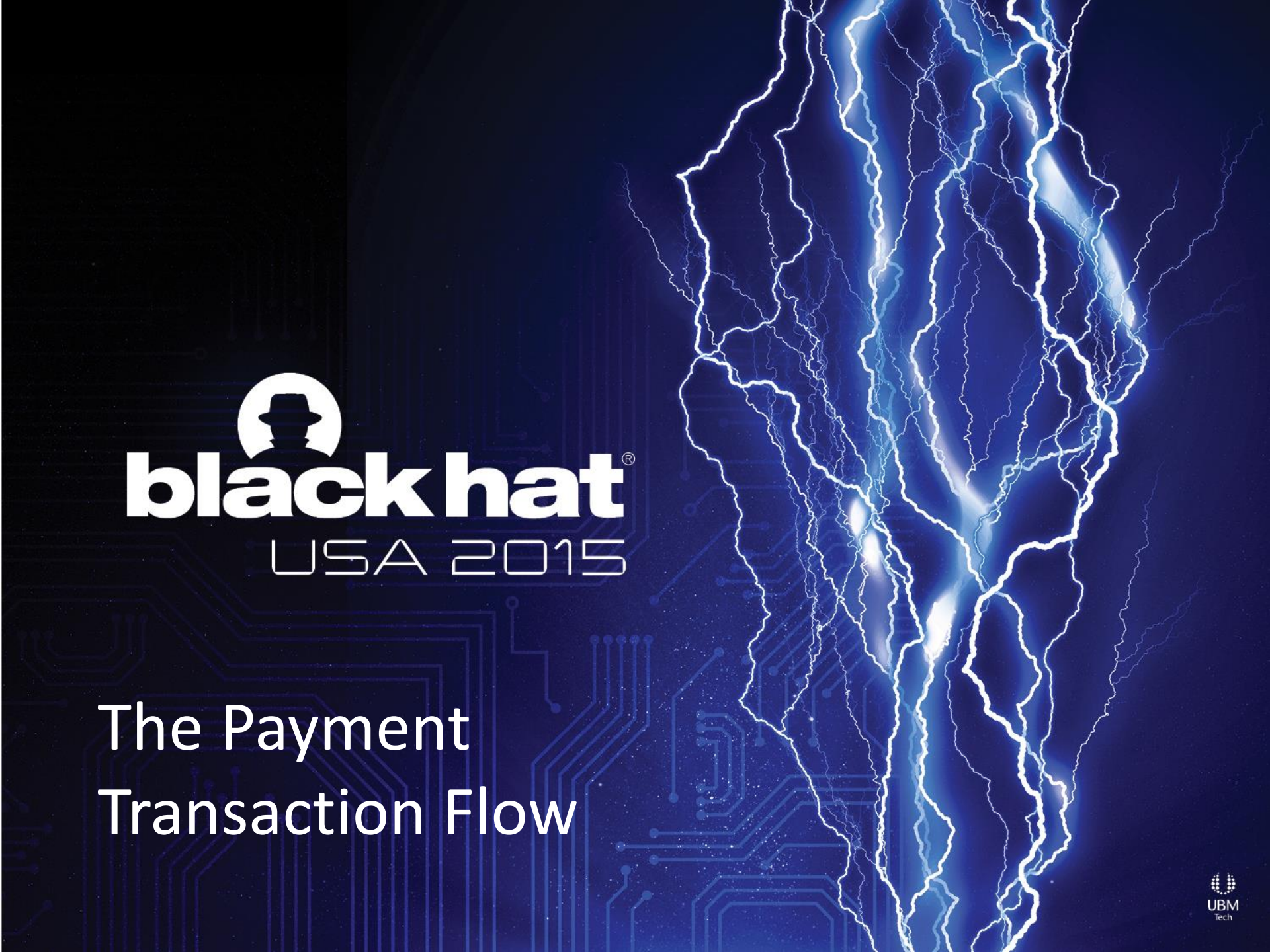
UBM Tech

Agenda

- Basics of an EMV payment transaction
- Review of Attacks
- Cloning A Mastercard
- Cloning A VISA
- EMV Issues
- ApplePay
- Tools Used
- Software Developed
- Key takeaways from this talk

Definitions

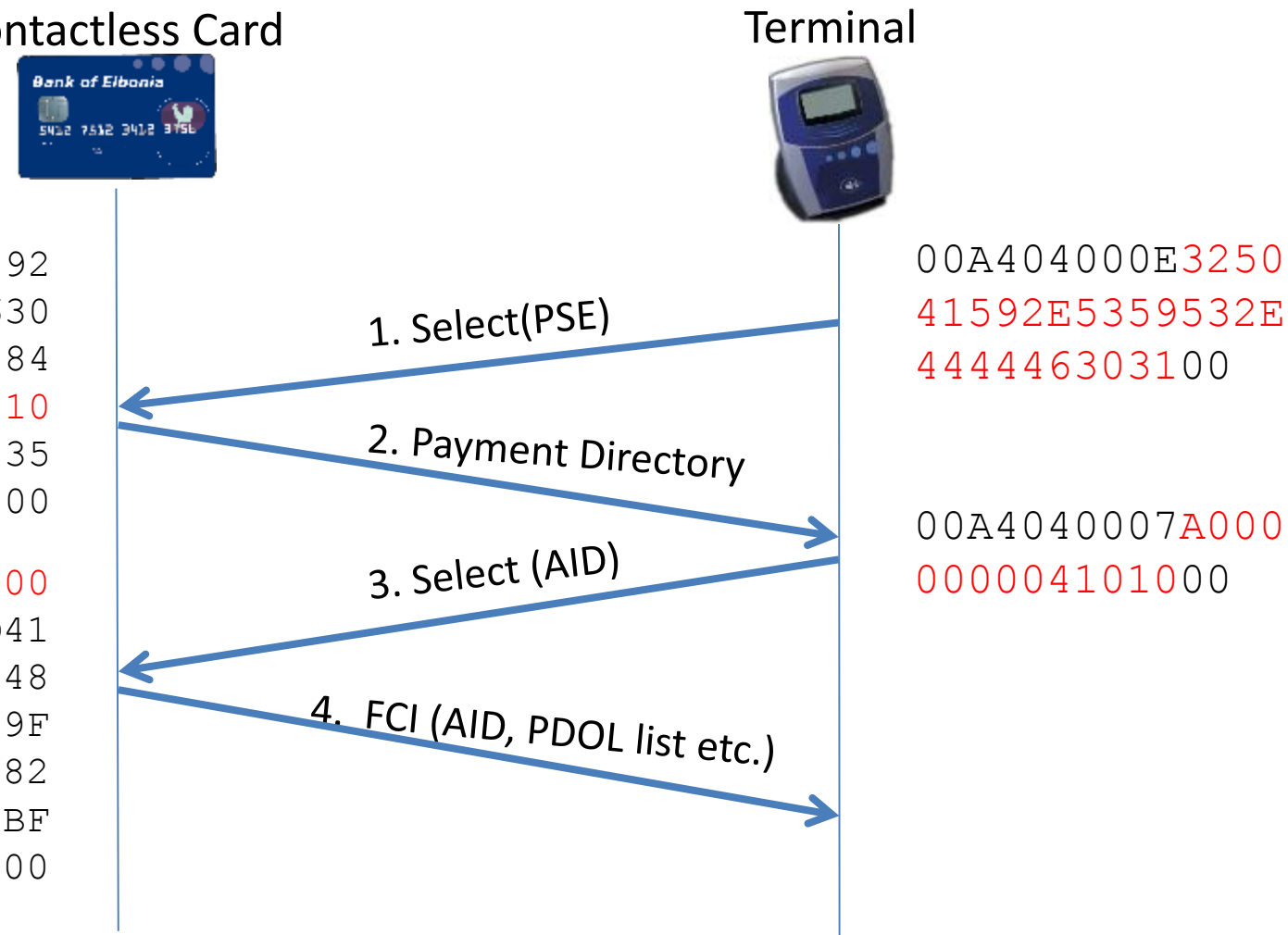
ATC	Application Transaction Counter	Monotonic counter of transactions performed
UN	Unpredictable Number	Random number used in transaction
CVV/ CVC	Card Verification Value (VISA)/ Card Verification Code (Mastercard)	Used to prevent alteration of data on the card.
dCVV/ CVC3	CVV3(MasterCard)/ dynamic CVV(Visa)	Used to prevent alteration of card data and prevent cloning of cards.
TTQ	Terminal Transaction Qualifiers (Visa)	Indicates what kind of card verification the terminal supports
PAN	Personal Account Number	Account Number assigned to the user
PSE	Payment Systems Environment	Tells terminal that the card is a banking card
AID	Application Identifier	Tells terminal what brands the card supports (MasterCard, Visa etc.)
PDOL	Processing Data Options List	List of tags we need the terminal to send the card (amount, UN etc.).
AFL	Application File Locator	Indicate what records the terminal needs to read.
AIP	Application Interchange Profile	Field to tell the terminal what authentications the card supports



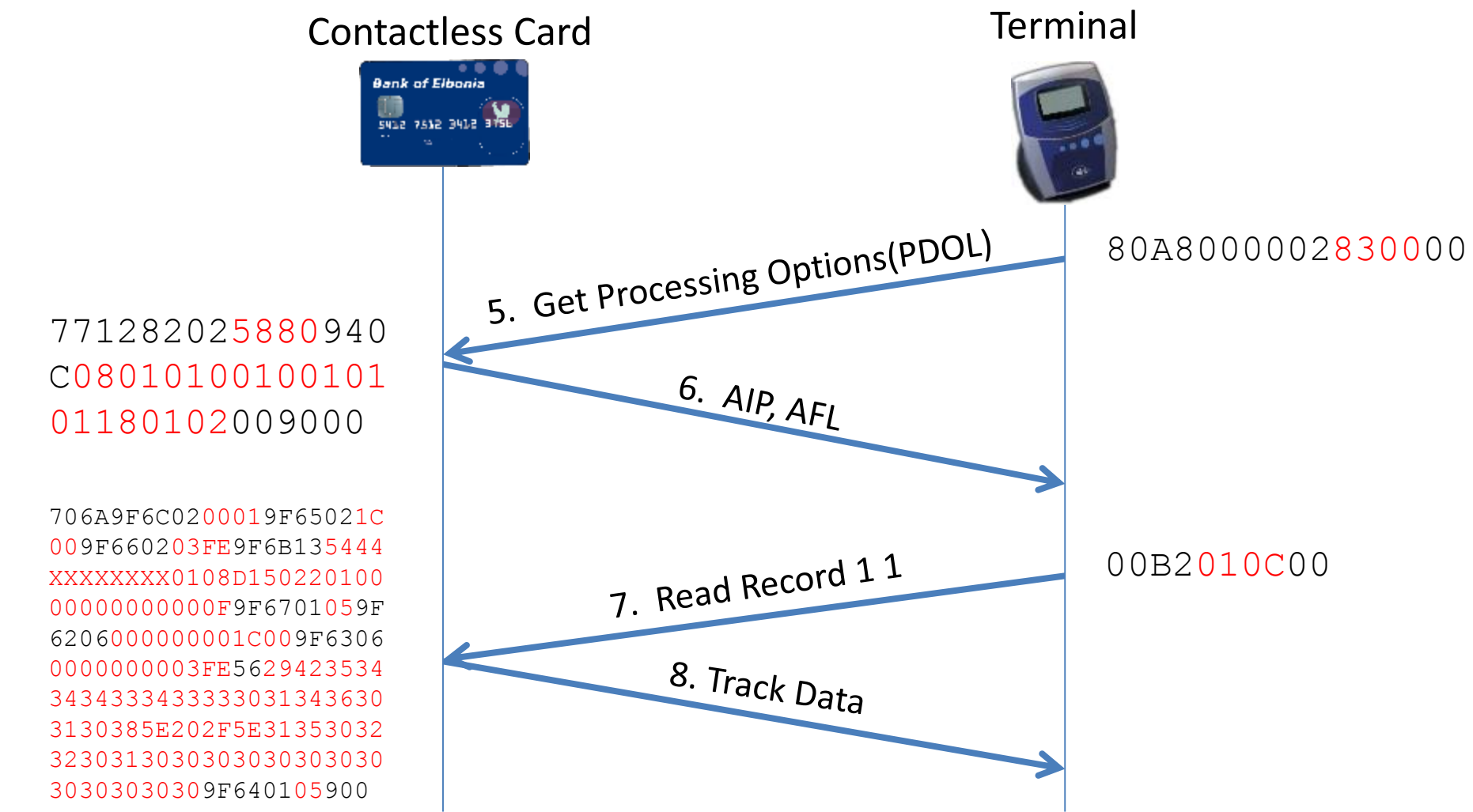
The Payment Transaction Flow

UBM Tech

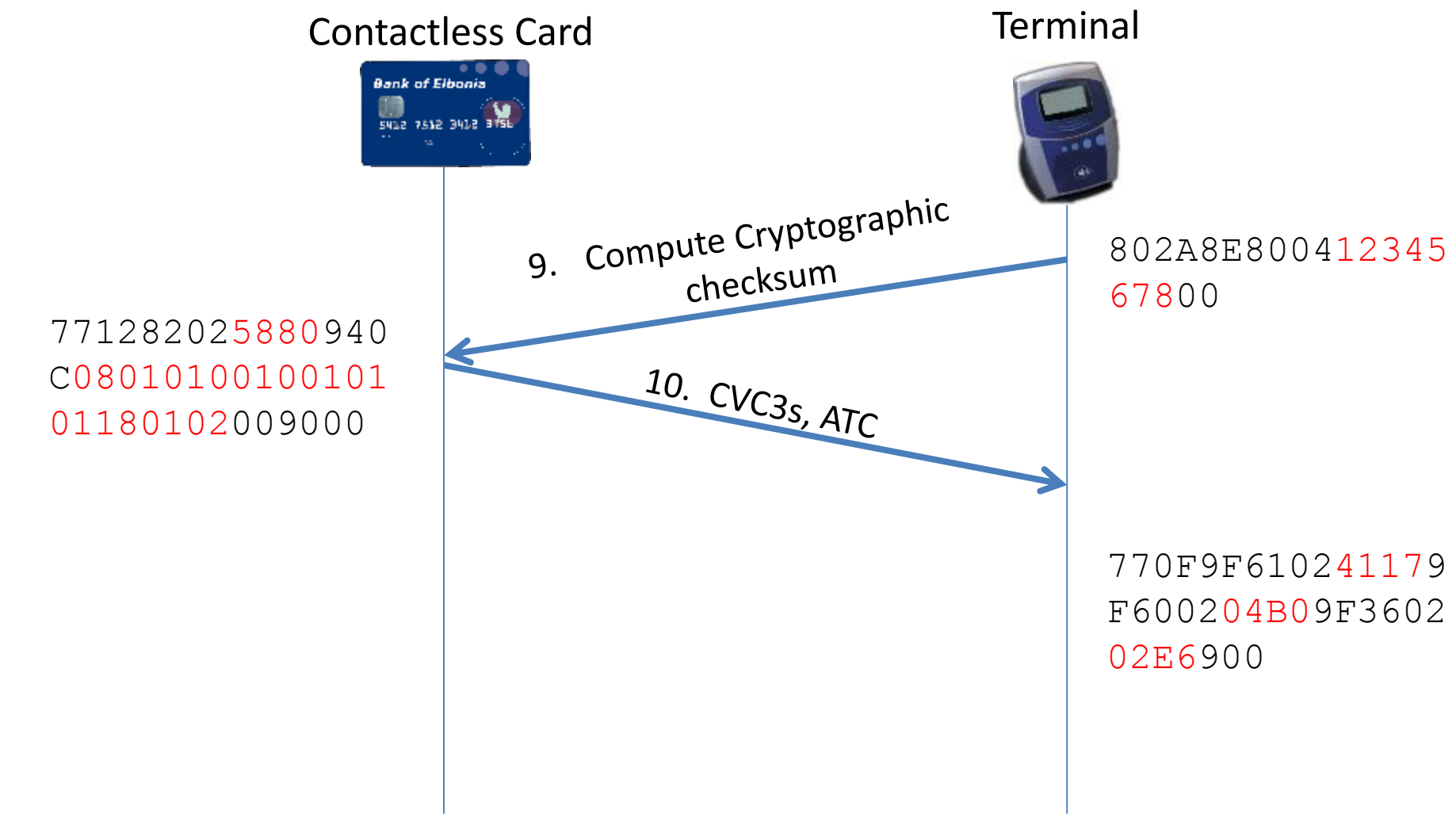
Transaction Initialization



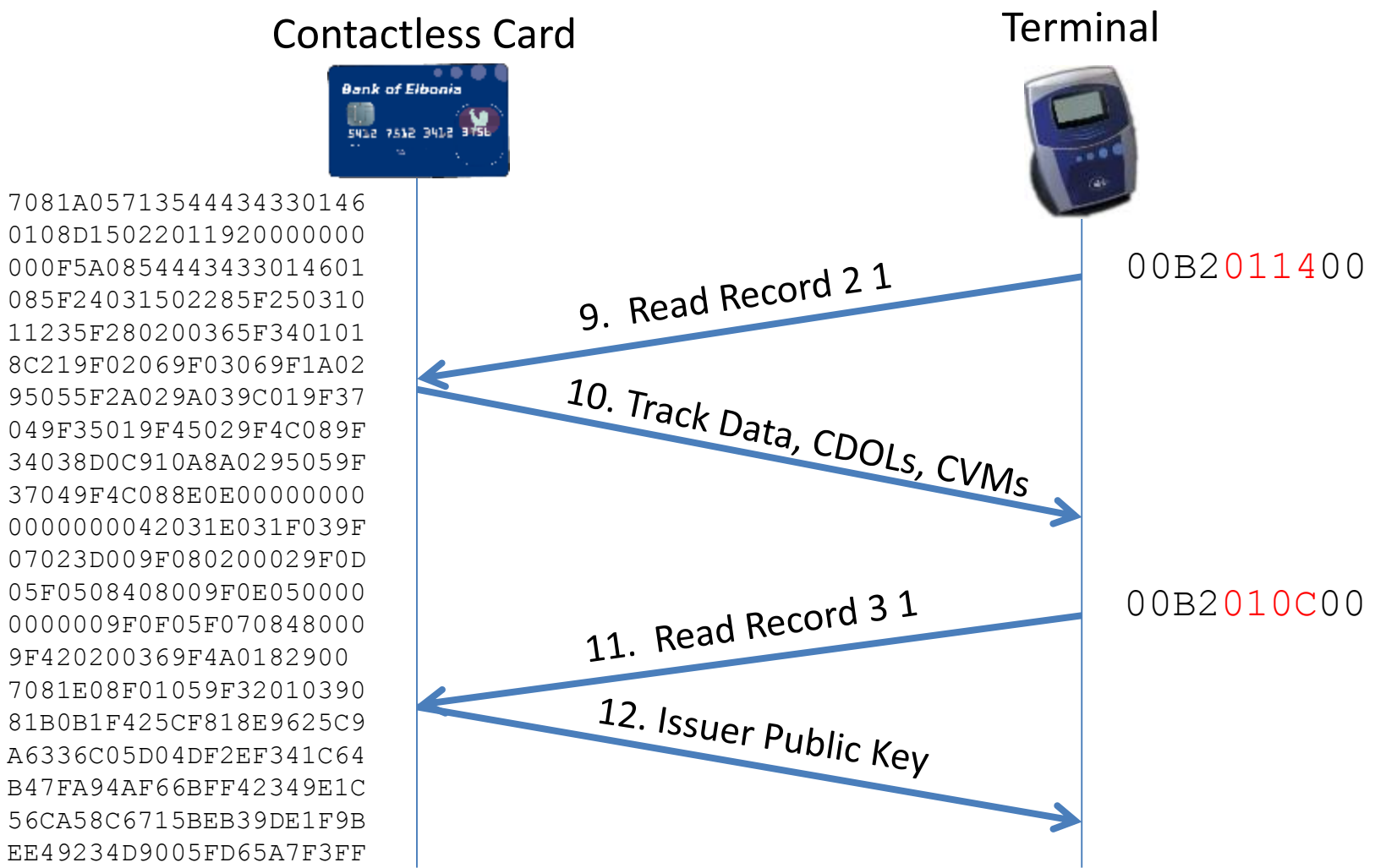
Get Transaction Parameters and Records



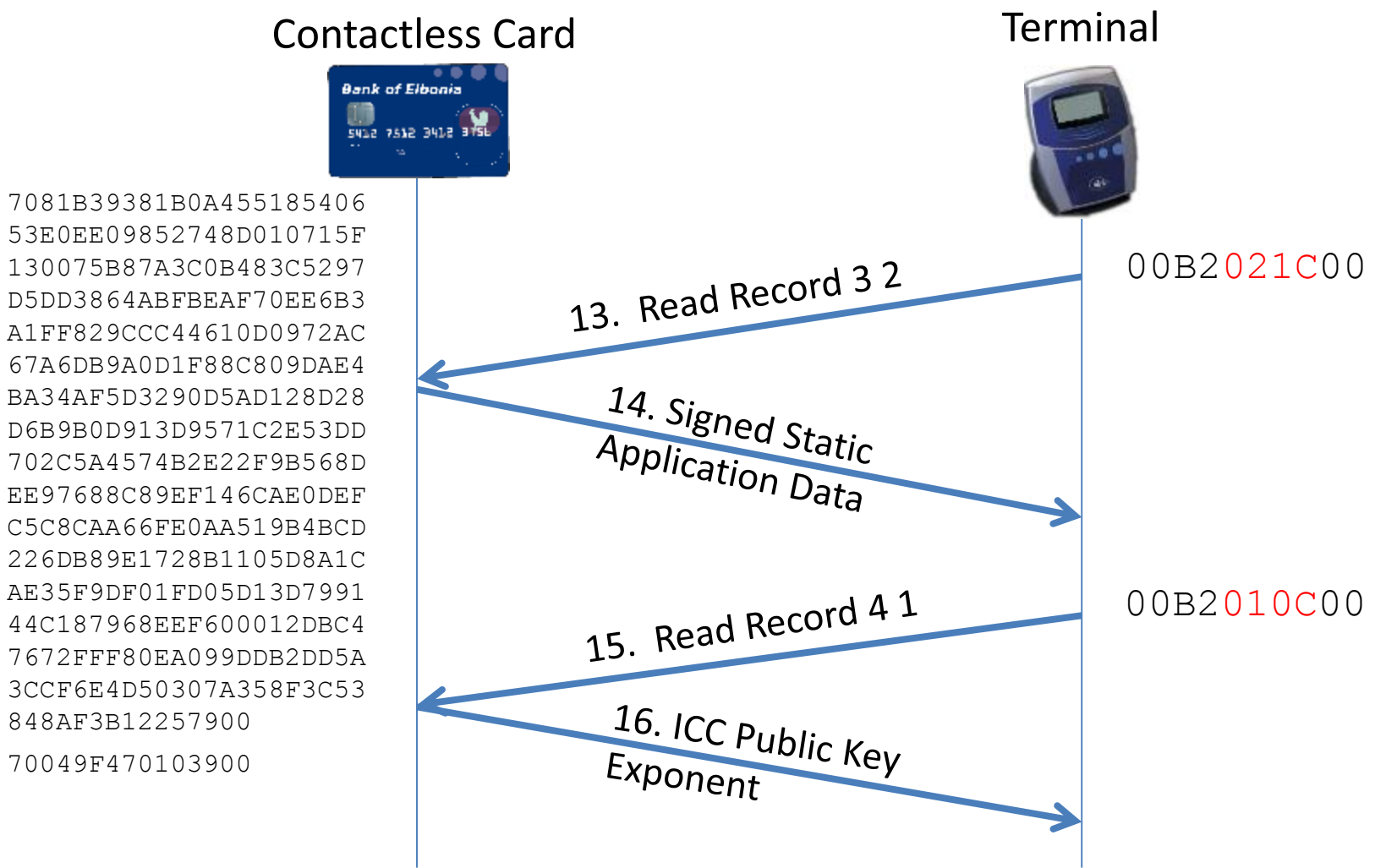
MasterCard Magstripe Transaction



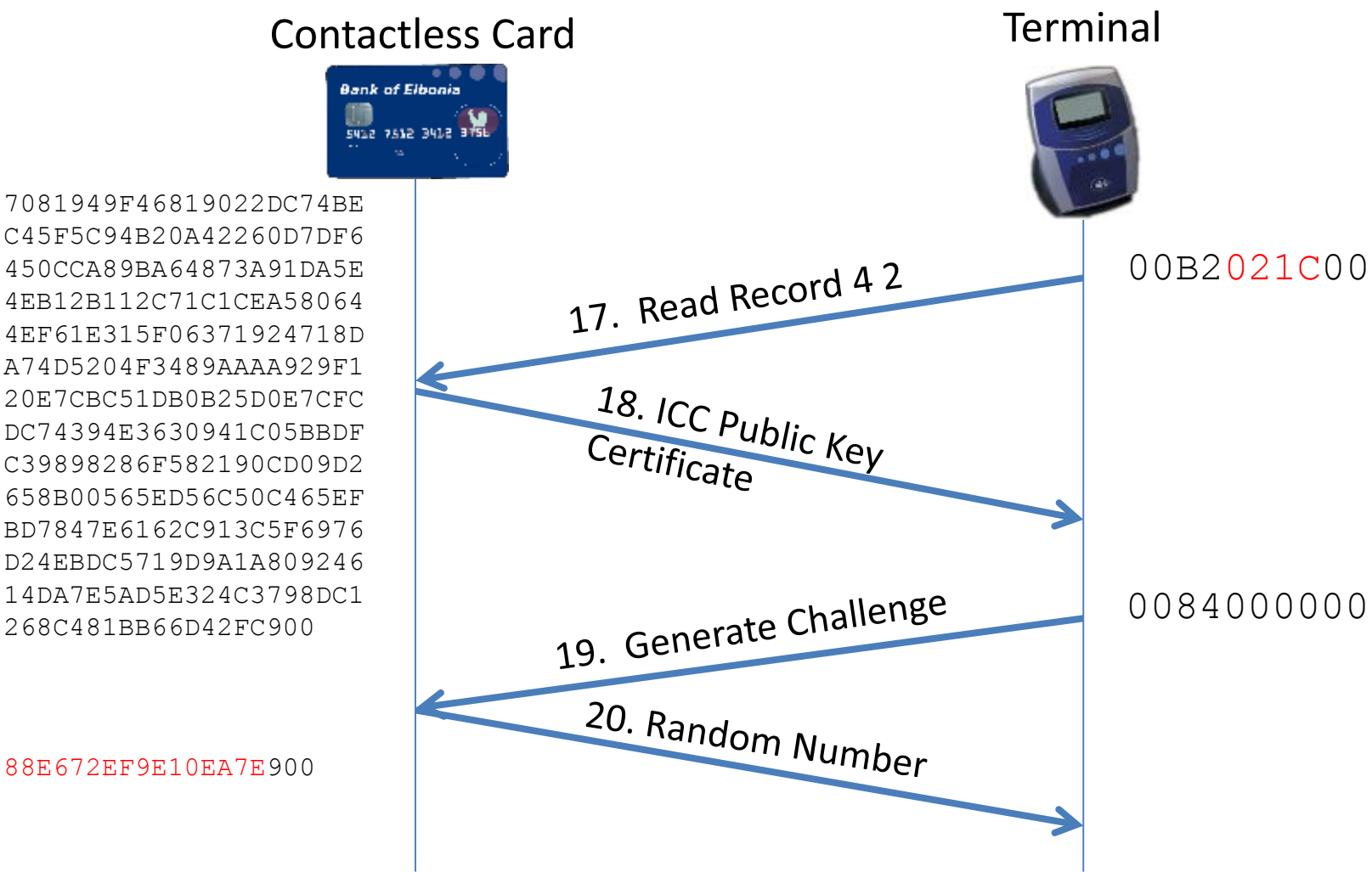
MasterCard M/CHIP Transaction



MasterCard M/CHIP Transaction



MasterCard M/CHIP Transaction



MasterCard M/CHIP Transaction

Contactless Card

Terminal

```
7781819F2701809F360208
D99F4B6052382F51D261DB
ED1D801A1FED56D2DA279F
4EA048FE0FFB296875D5DA
056D606582849307A9EAF2
1D96FAF9648C80AF50118F
40495877DD6D6E32A404CB
C0B67D48490216D7307361
D5B380909F7B6CC45D311F
2C9AC08802944528B35AA0
859F10120210A040012200
00000000000000000000FF
900
```

21. Generate Application Cryptogram (CDOL)

22. CID, ATC, Signed Dynamic Data

```
80AE90002B00000
0000001000000000
0000003600000000
0000036140101000
0000000011000088
E672EF9E10EA7E00
00000000
```

A Review of Attacks

UBM Tech

Steal Someone's Card

Police in talks with banks to fight tap-and-go crime wave

MARK BUTTLER HERALD SUN JANUARY 16, 2015 10:00PM

- This is the major form of fraud from contactless transactions.
- Contactless (and Chip in the US) require no authentication.
- Limited by transaction amounts
  - £20 in the UK
  - AUD$80 in Australia etc.

Relay Attack

- Contactless/contact transactions contain no distance bounding protections.
- Made easier with the emergence of native support for Host Card Emulation on Android

See the talks and papers:

- 2007 – “Keep Your Enemies Close: Distance Bounding Against Smartcard Relay Attacks.” Saar Drimer and Steven J. Murdoch @ Cambridge
- 2012 - “NFC Hacking: The Easy Way” Eddie Lee @ Defcon
- 2015 – “Relay Attacks in EMV Contactless Cards with Android OTS Devices” Ricardo Rodriguez and Pepe Vila @ HITB

Solutions

Card Cloning

Why?

Cards are mini HSMs

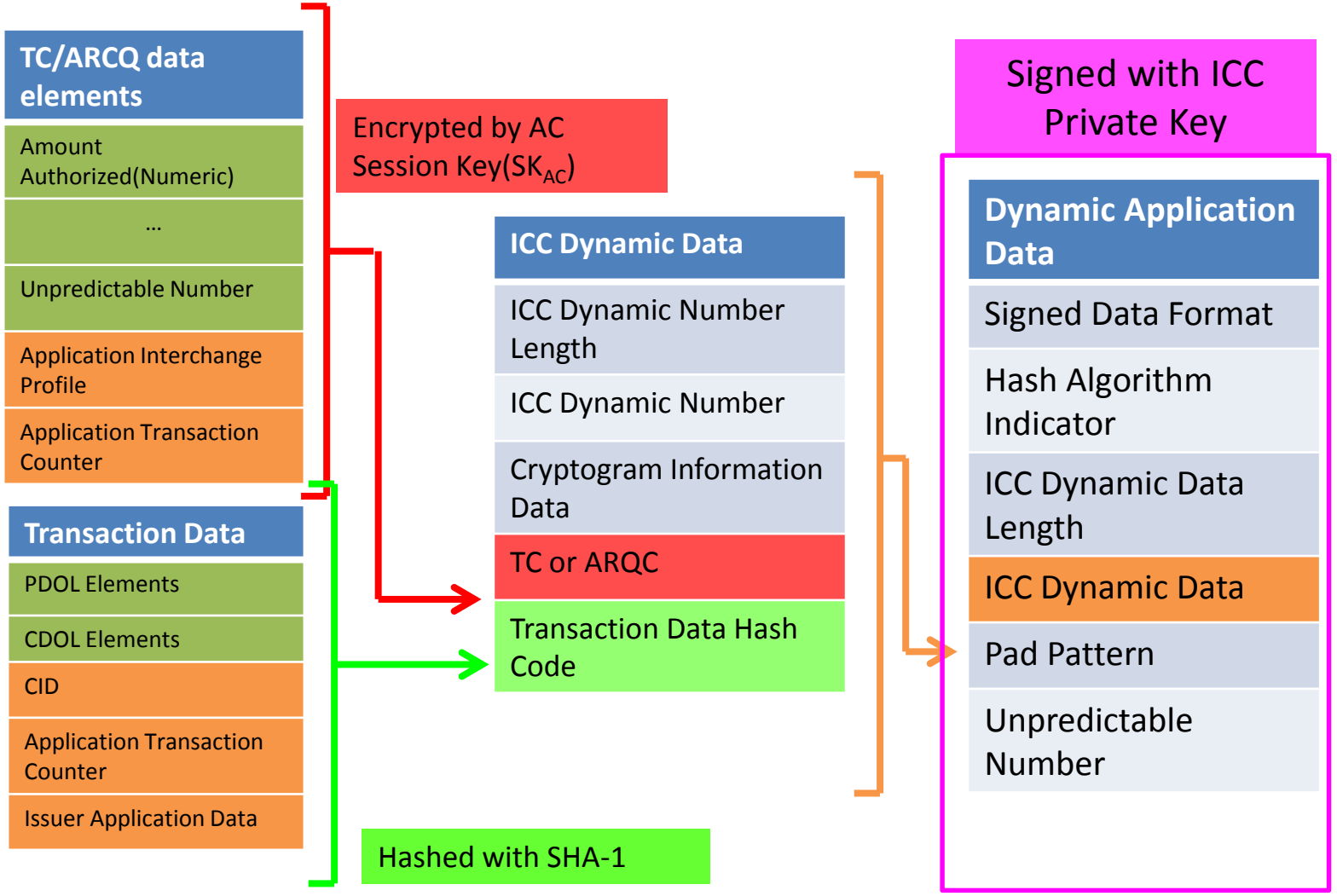
- Cards have hardware crypto accelerators and key storage.
- Physical protections against attack
- Small attack surface
- Normally EAL CC 5+ certified
  - Semiformally Designed and Tested
  - Includes testing for side-channel attacks

Dynamic CVCs

Symmetric Keys

Key	Name	Description
KD_{CVC3}	ICC Derived Key for CVC3 Generation	Symmetric Key used for generating the CVC3
MK_{AC}	ICC Application Cryptogram Master Key	Symmetric Key used to derive the session key for generation of the Application Cryptogram
SK_{AC}	ICC Application Cryptogram Session Key	Symmetric Key used to generate the Application Cryptogram

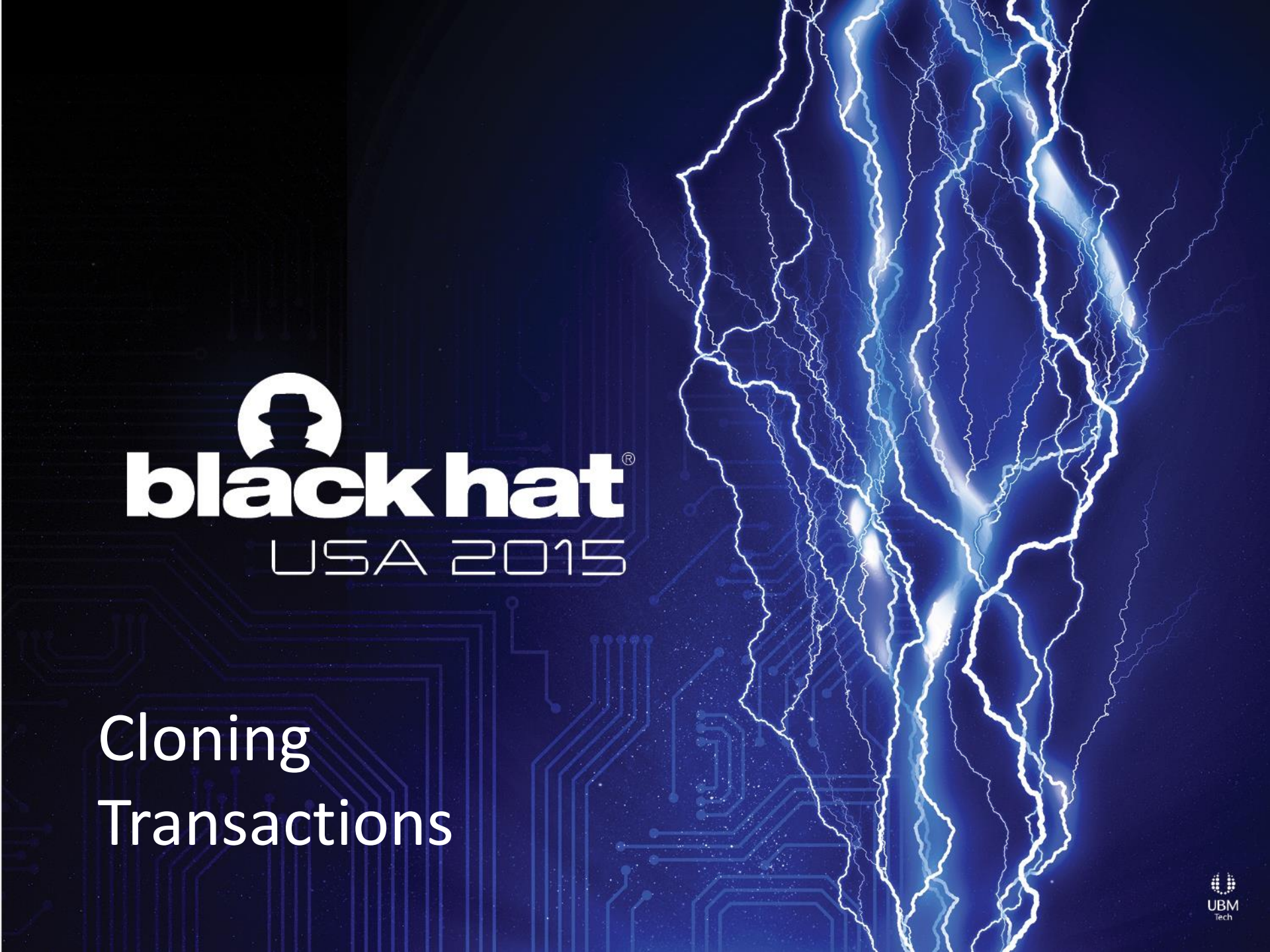
Dynamic Signing



Talk over?

Transaction Cloning

- Full chip based EMV transaction take time
- Requires upstream equipment to support (terminal upgrades, new HSMs etc.).
- So the contactless standards includes modes to support old equipment and quick transactions.
- Key to the cloning of transactions is the “Magstripe” modes
- These are designed to be used with equipment that can only support magnetic card data
- MasterCard – Magstripe Mode
- VISA – dCVV and CVN17



Cloning Transactions

UBM Tech

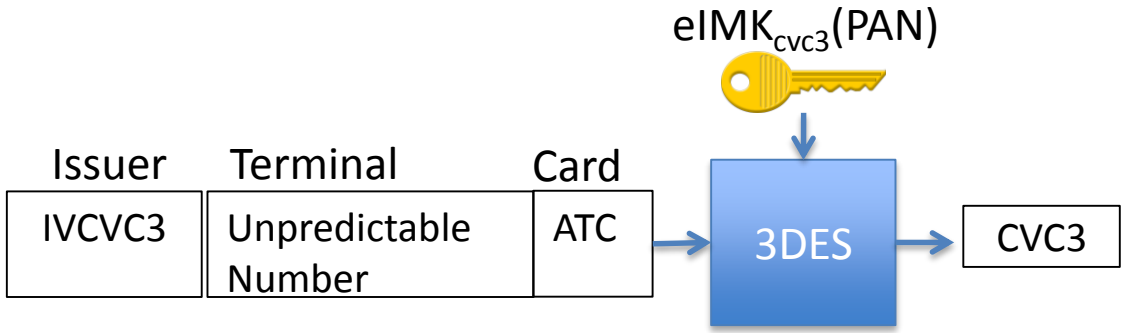
Magstripe Mode

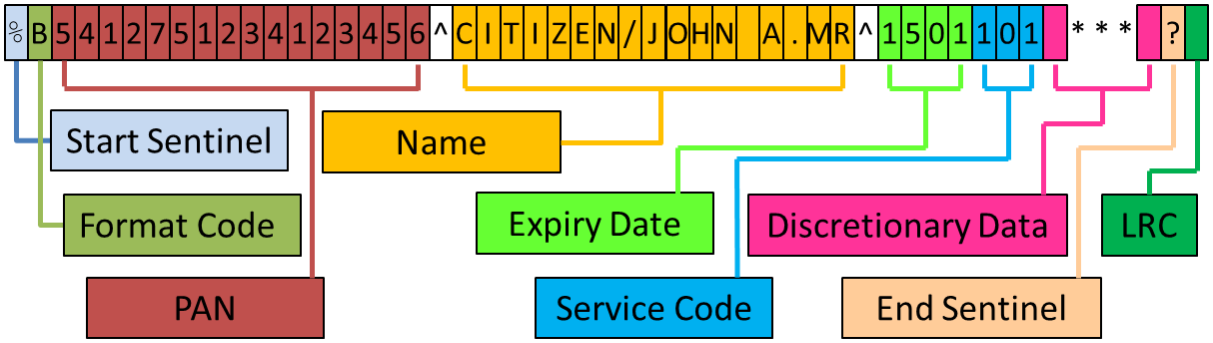
- Magstripe mode consists of the terminal generating track data similar to the physical magstripe.
- We get the card to generate a dynamic CVV that the terminal insert into the track.
- This is sent off to the payment processor for verification.
- The weaknesses is how the CVVs are generated

MasterCard

Compute Cryptographic Checksum

CLA	INS	P1	P2	Lc	Data	Le
00	2A	8E	80	Var.	UDOL related data	00





Forming the UN

Length of UN = NumBitsSet(Ktrack) – Ttrack

Ktrack = Number of non zero bits in the track 1 bit map
Ttrack = Number of digits of ATC to be included

CVV is formatted as Binary Coded Decimal.

Take a UN of 4 bytes:

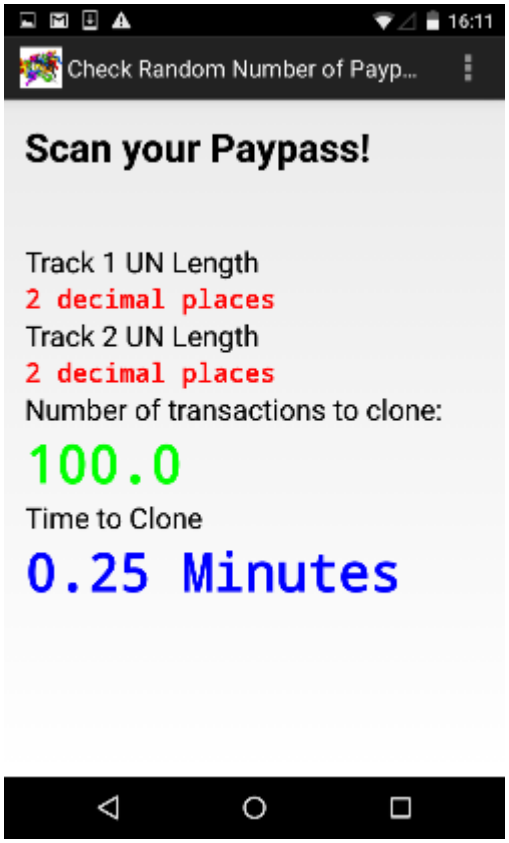
- 4 bytes binary = 2^{32} values = 4,294,967,296
- 4 bytes BCD = 10^{8} values = 100,000,000
- UN length of 2 = 10^{2} values = 100

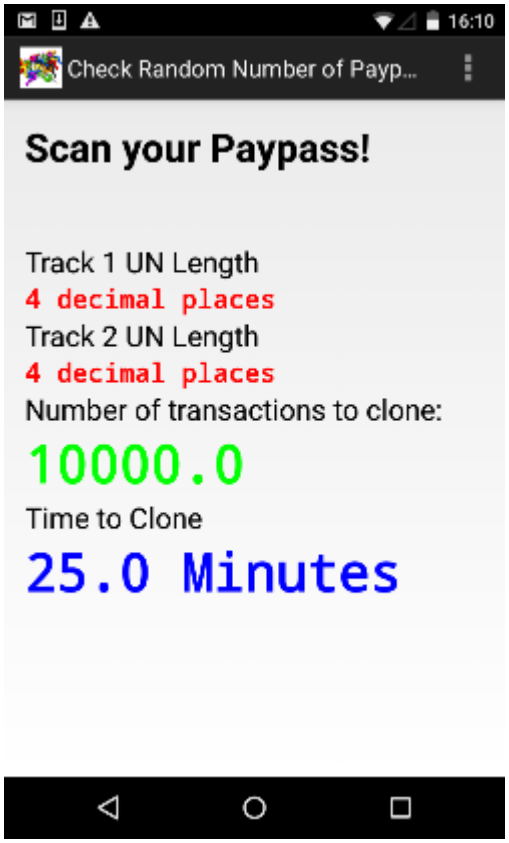
How to clone a transaction

1. Read and copy card records
2. Generate dictionary of COMPUTE CRYPTOGRAPHIC CHECKSUM responses for all possible terminal random numbers
3. Flip the M/CHIP support bit (tag 82)
4. Replay stored records to the terminal
5. Look up UN returned by the terminal in the dictionary
6. Collect purchase and get out of there.

MasterCard

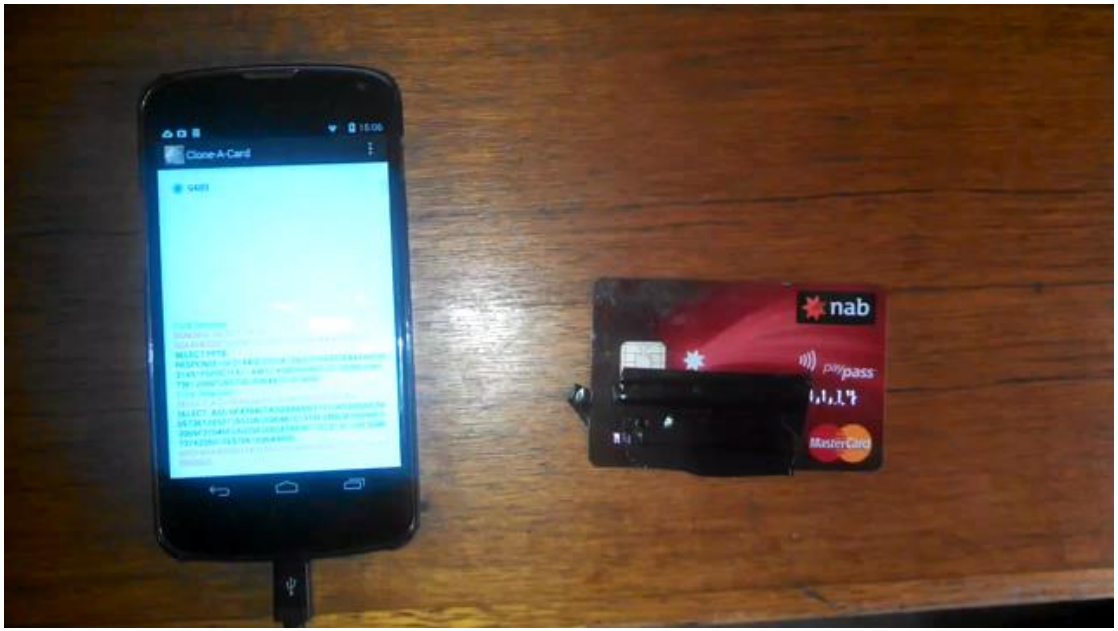
How long does it take?





https://github.com/peterfillmore/Check-Paypass-Random-Number

Demo



MasterCard

The flaw is built into the system

A *PayPass* card using the MasterCard brand:

- Must support *PayPass*—Mag Stripe transactions (unless for domestic use only)
- May support *PayPass*—M/Chip transactions

R	**ALL**	The Unpredictable Number must be at least 2 digits in length.

The terminal should monitor the number of aborted transactions. If the frequency is high it is likely that a fraudster is trying to get a specific value of the Unpredictable Number. The terminal should take appropriate measures to reduce the risks of an attack, such as introducing wait times after three aborted transactions.

Paypass – M/Chip Requirements 2013

Fixes

- Attack can be detected on the card issuers side
  - ATC will jump.
- Card issuers need to issue cards with UN lengths of at least 4.
- Issuers should prompt for a second factor of authentication on failed transaction
  - PIN, Insert Chip Card
- Payment Processors should reject non M/Chip transactions over contactless.
- Terminals must be tested on RNG generation

Cloning a VISA

UBM Tech

VISA Modes

- Dynamic CVV - dCVV
  - Legacy magstripe equivalent mode
  - Terrible, broken on release
- Cryptogram Version Number 17 - CVN17
  - Updated to magstripe equivalent mode
  - Lot better than dCVV
- Quick Visa Smart Debit/Credit - qVSDC
  - Reduced EMV mode
  - Defined in standard for speed
- Visa Smart Debit/Credit - VSDC
  - Full EMV mode (i.e. CDA)
  - Slower – requires card to be in field for complete transaction

VISA Modes

9F66 - TTQ – Terminal Transaction Qualifier

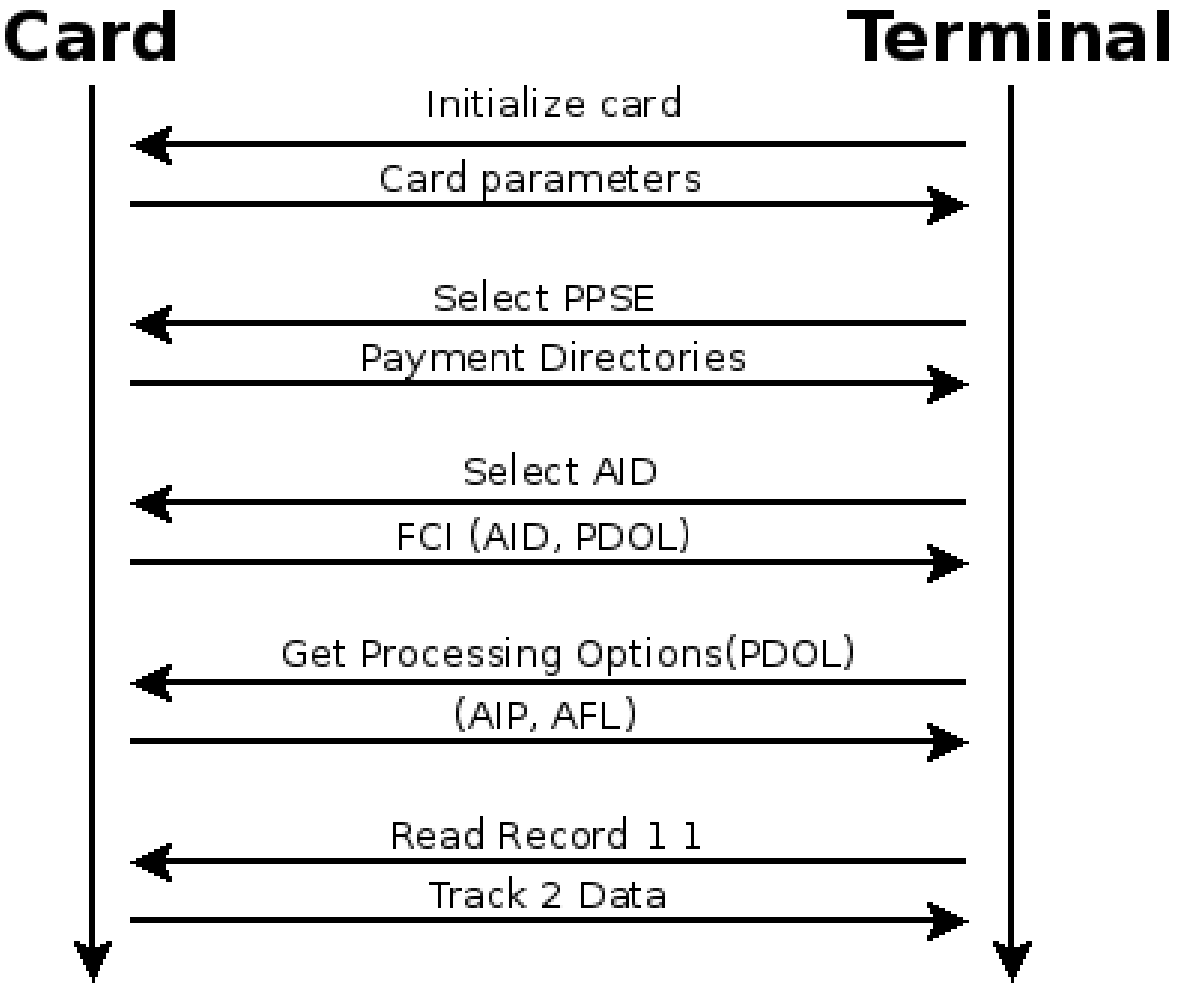
Table 1 – Summary of Possible Card / Reader Interactions

Reader Configuration \ Contactless Card Capability	MSD and qVSDC	MSD, qVSDC, and VSDC
MSD and qVSDC	qVSDC	qVSDC
qVSDC only	qVSDC	qVSDC
qVSDC and VSDC	qVSDC	VSDC
MSD, qVSDC, and VSDC	qVSDC	VSDC
MSD and VSDC	MSD	VSDC
MSD	MSD	MSD

Table 3 – Terminal Transaction Qualifiers (Tag '9F66')

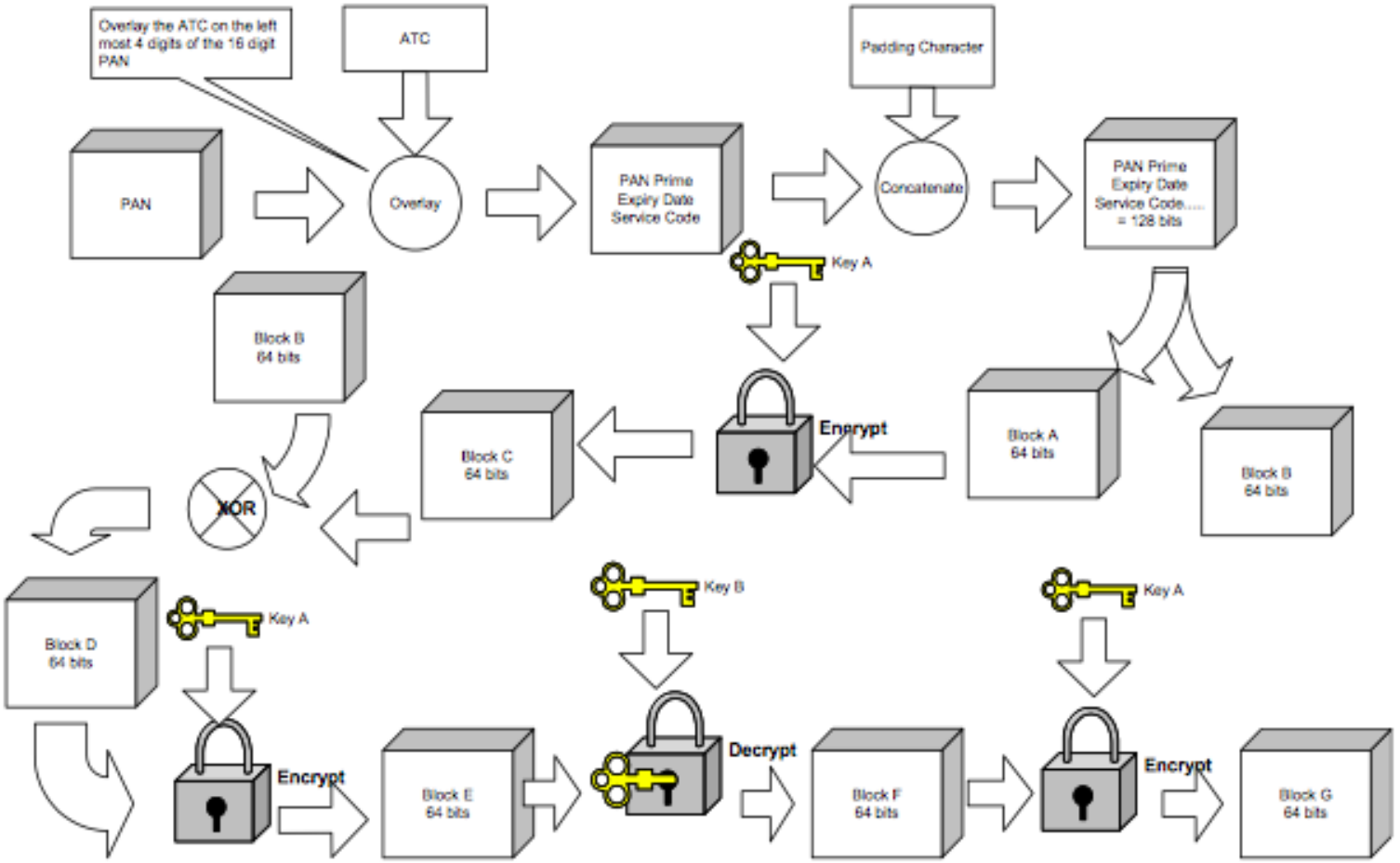
Byte	Bit	Definition
1	8	'1' – Contactless magnetic stripe (MSD) supported '0' – Contactless magnetic stripe (MSD) not supported
	7	'1' – Contactless VSDC supported '0' – Contactless VSDC not supported
	6	'1' – Contactless qVSDC supported '0' – Contactless qVSDC not supported
	5	'1' – Contact VSDC supported '0' – Contact VSDC not supported
	4	'1' – Reader is Offline Only '0' – Reader is Online Capable
	3	'1' – Online PIN supported '0' – Online PIN not supported
	2	'1' – Signature supported '0' – Signature not supported
	1	RFU – b'x'
2	8	'1' – Online cryptogram required '0' – Online cryptogram not required
	7	'1' – CVM required '0' – CVM not required
	6-1	RFU – b'xxxxxx'
3	8-1	RFU – b'xxxxxxxx'
4	8-1	RFU – b'xxxxxxxx'

dCVV Mode


Card
Terminal
Initialize card
Card parameters
Select PPSE
Payment Directories
Select AID
FCI (AID, PDOL)
Get Processing Options(PDOL)
(AIP, AFL)
Read Record 1 1
Track 2 Data

dCVV algorithm

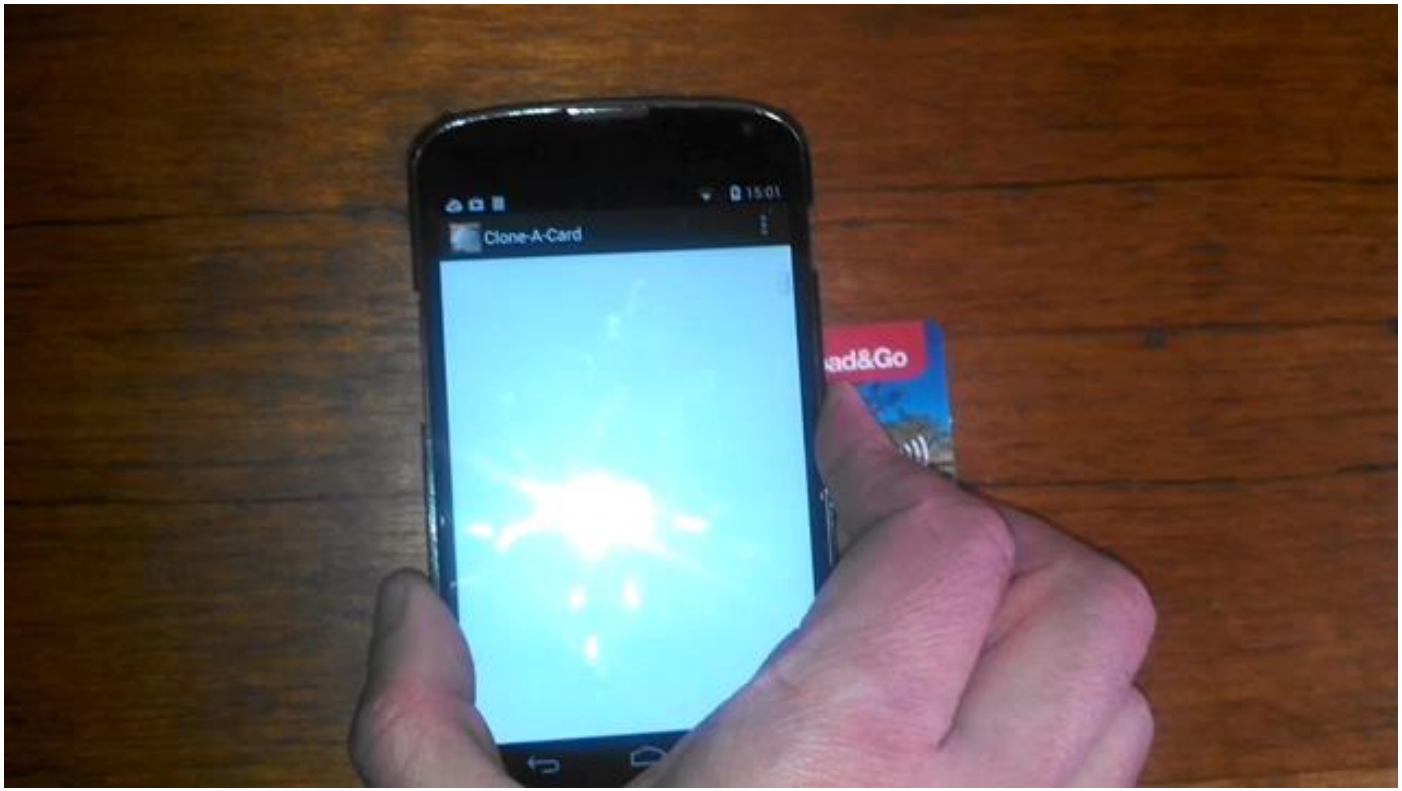
Figure 5 – Dynamic CVV Algorithm



Cloning

1. Read and copy card records
2. Turn the magstripe bit on (set AIP bytes to 0x0080
3. Replay stored records to the terminal
4. Collect purchase and get out of there.

Demo



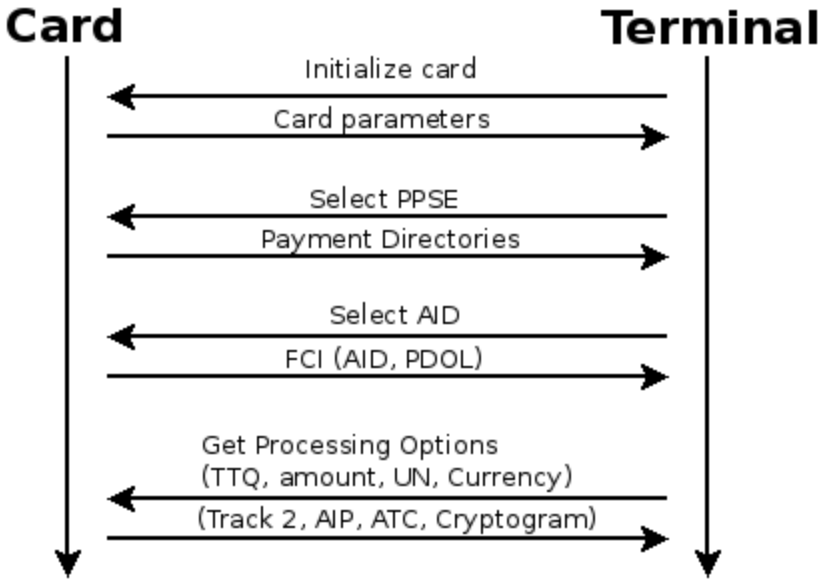
Prevention

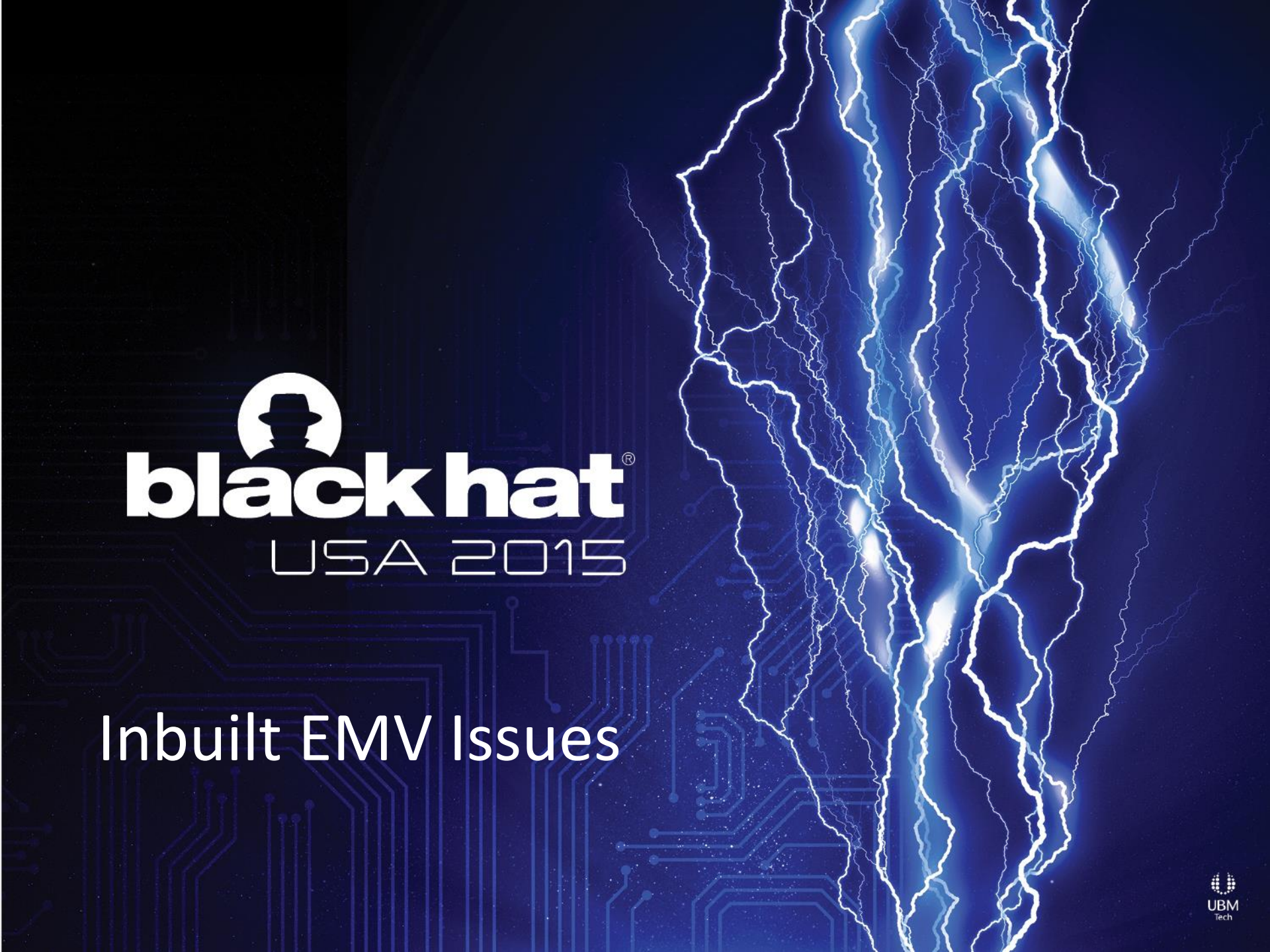
It has been agreed that a migration from dCVV to Cryptogram Version 17 will take place for MSD readers and that by a migration date to be determined, MSD readers will support Cryptogram 17. An MSD market is not a full data market and Cryptogram 10 is not supported in these markets.

- Requires explicit support from the terminal
- This mode has been obsoleted by the CVN17 method
- Requires that the terminal support dCVV mode.
- So the payment processor should disable dCVV mode on the terminal.
- Issued cards should not support this mode (my debit VISA card only supports fDDA modes)

CVN17

- Here we use the “Get Processing Options” command to generate the Application Cryptogram
- This is transmitted separately from the track data
- Also contains amounts, currency and a terminal UN

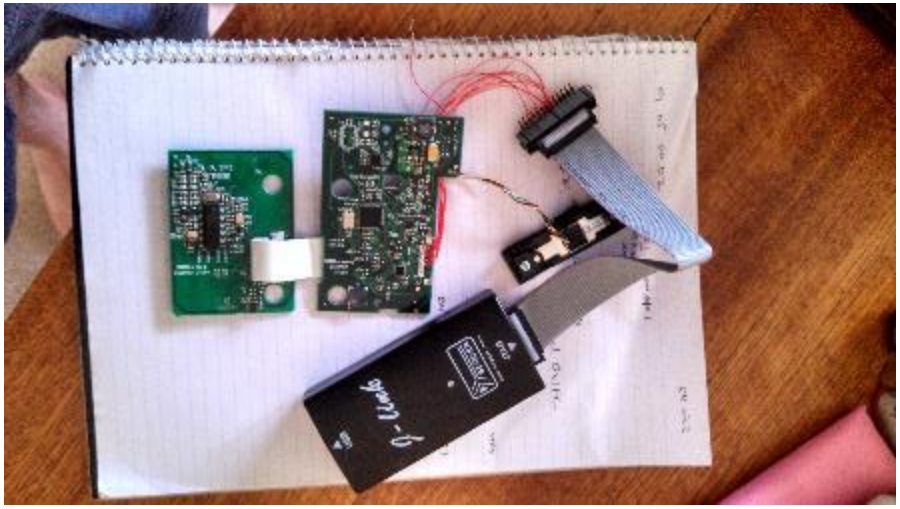




Inbuilt EMV Issues

Terminals

- EMV Only terminals require no tamper resistance.
- Certification only covers interoperability
  - Does the terminal play well?
- Terminals are not required to be tested for any logical security
- Update mechanisms are not defined by EMVco



UNs

- Are an integral part of EMV security – yet:
- UN is not defined to be generated by a cryptographic random number source
  - Frequently will include a date or counter value

New text:

6.5.6 Unpredictable Number

The terminal shall be able to generate an Unpredictable Number (tag 9F37) to be used for input to the card cryptograms (Application Cryptograms and DDA/CDA signatures) so as to ensure the unpredictability of data input to this calculation and thereby the freshness of the cryptogram.

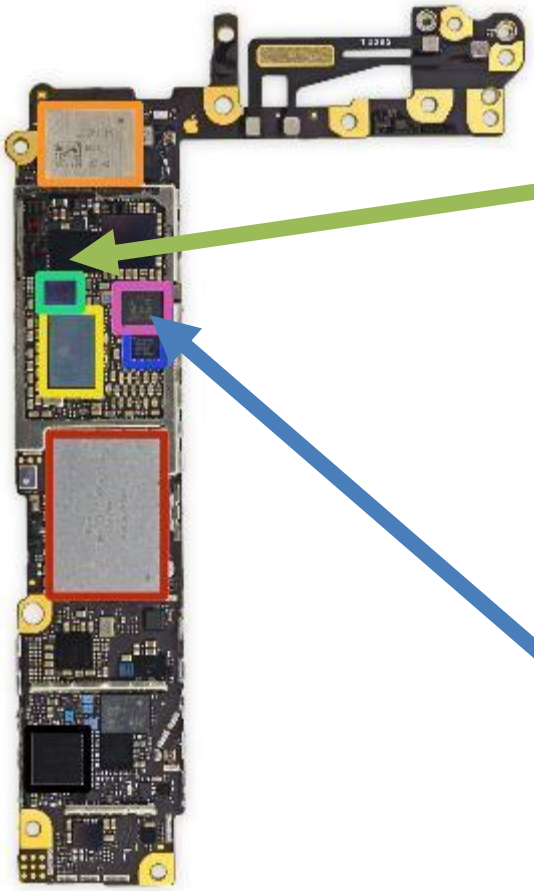
A terminal may use the same Unpredictable Number throughout a transaction. The Unpredictable Number could be generated by a dedicated hardware random number generator or could, for example, be a function of previous Application Cryptograms, the terminal Transaction Sequence Counter and other variable data (e.g. date/time). In the second example the function could be a hash function or a keyed encipherment function.

Section 11.3 of Book 2 provides an example of an approved method for generating the Unpredictable Number using a hash function.

Apple Pay

UBM Tech

Hardware



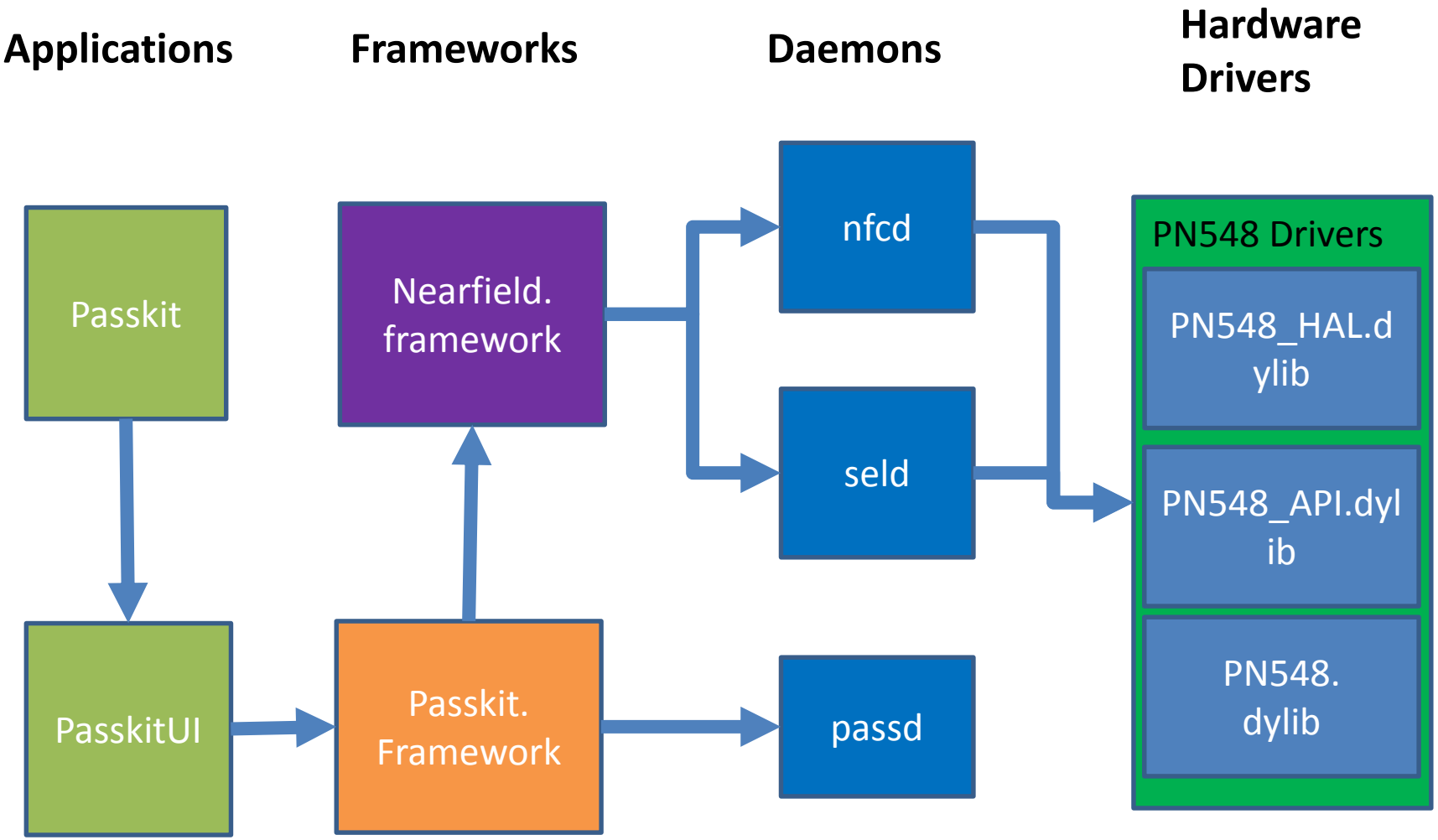
AMS AS3923
Power Booster

NXP 65v10

PN548

Secure Element

Software


Applications
Frameworks
Daemons
Hardware Drivers
Passkit
PasskitUI
Nearfield. framework
Passkit. Framework
nfcd
seld
passd
PN548 Drivers
PN548_HAL.dylib
PN548_API.dylib
PN548. dylib

NFC Hardware

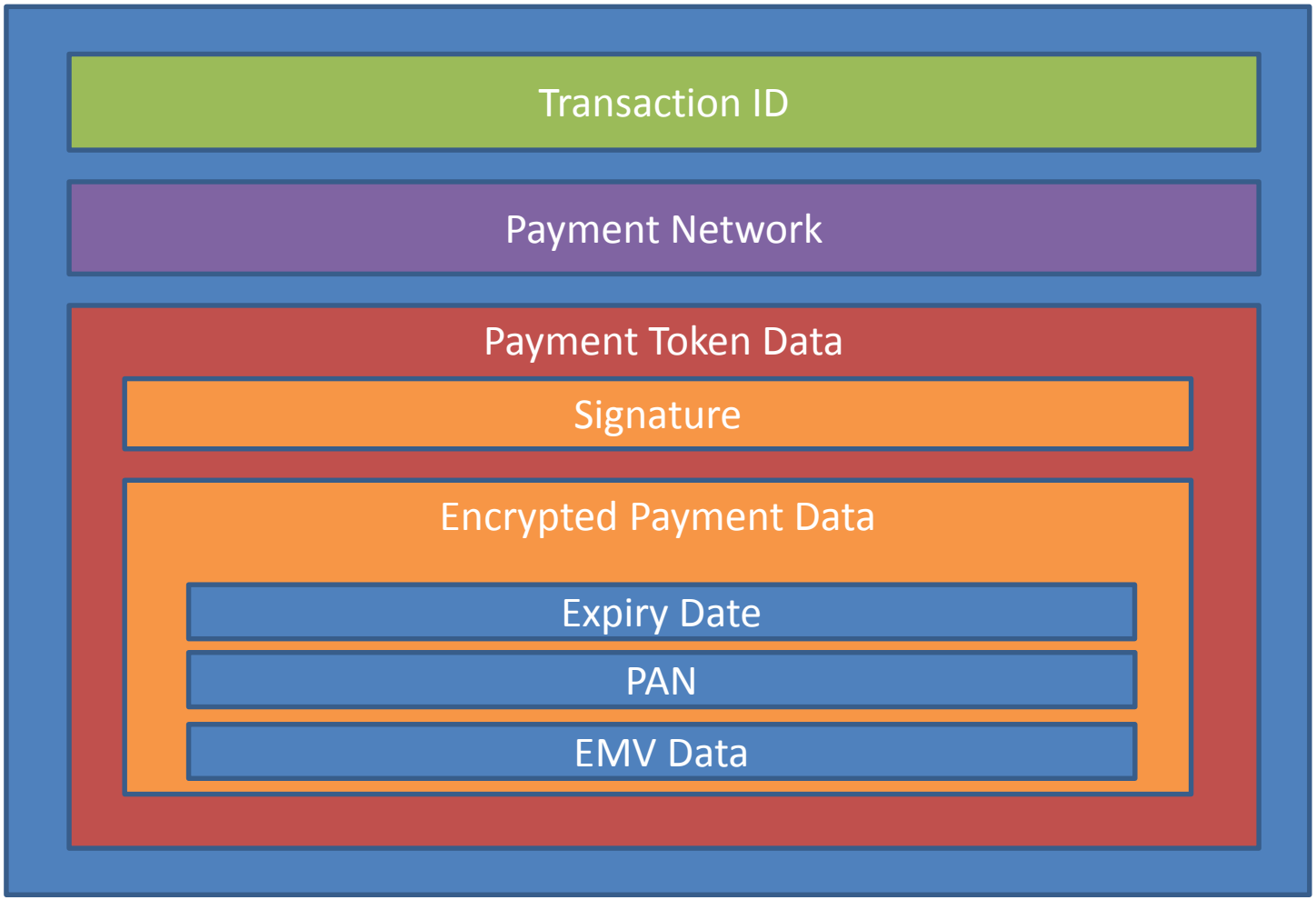
Secure Element

- Holds the token, keys, certificates and commands needed to perform a transaction
- Can be managed remotely by the issuer
- Loaded over a remote connection by the card issuer

PN548 Controller

- Handles the rest of the transaction
- Interfaces with the secure element to perform the transaction
- Performs the transaction by itself.
- Returns the necessary values back as EMV data to the Nearfield.Framework to form the payment token.

PKPaymentToken



https://github.com/beatty/applepay_crypto_demo

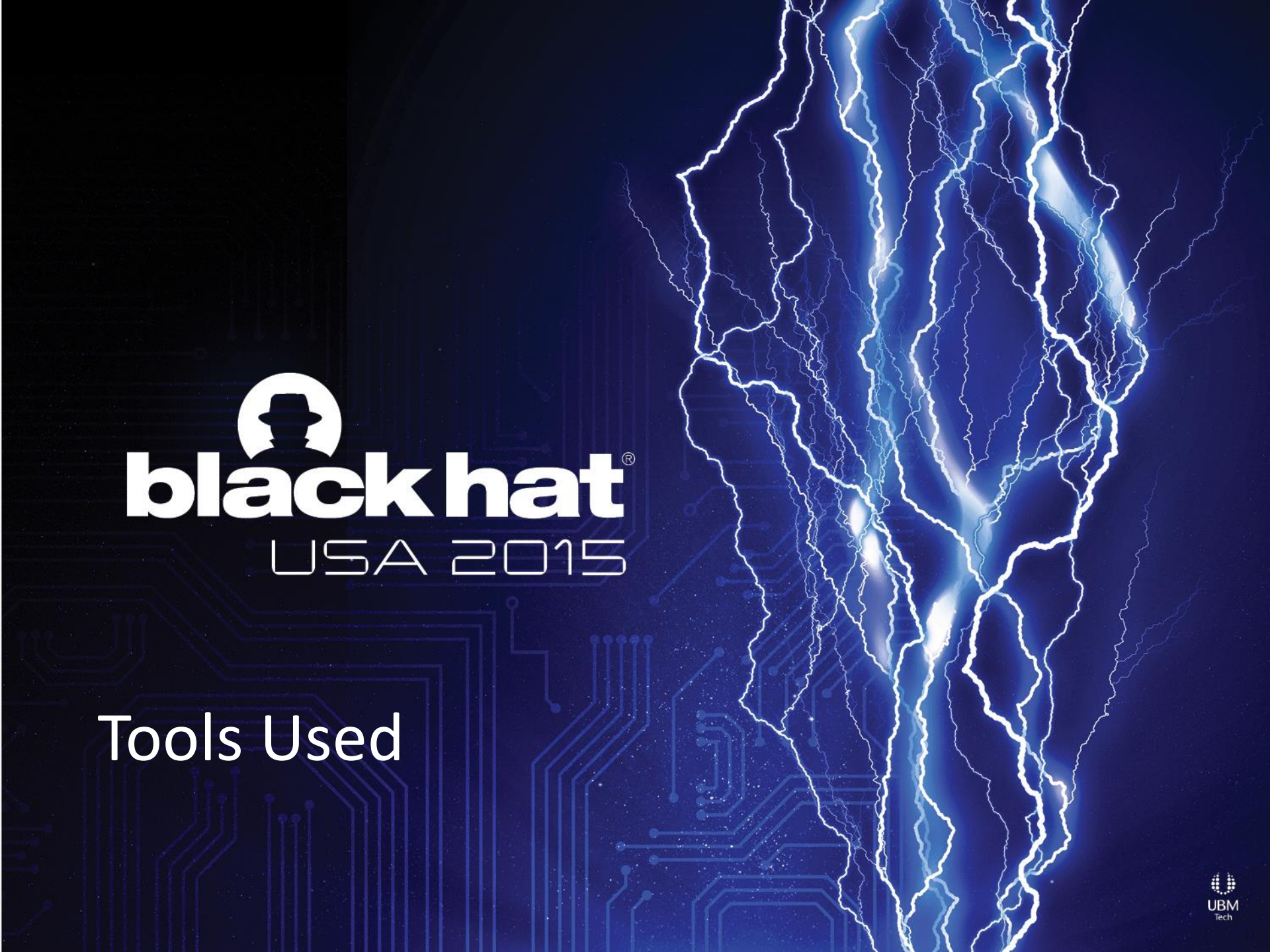
Dump Approved AIDs

Peters-iPhone:~ root# ps aux | grep "passd“
mobile 284 … /System/Library/Frameworks/PassKit.framework/passd
Peters-iPhone:~ root# cycript -p 284
cy# mySE = [[PDSecureElement alloc] init]
#"<PDSecureElement: 0x13f6894d0>“
cy# mySE.secureElementCards
@[#"<NFCard: 0x13f681700> { aid=A0000000041010010000001
family=0x0(UNKNOWN) lifecycle=0x7(selectable) activation=0x80(non-activatable) authTransient=YES }",#"<NFCard: 0x13f646d40> { aid=A0000000031010010000001 family=0x0(UNKNOWN) lifecycle=0x7(selectable) activation=0x0(deactivated) authTransient=YES }",#"<NFCard: 0x13f59c9f0> { aid=A0000000250109010000001
family=0x0(UNKNOWN) lifecycle=0x7(selectable) activation=0x0(deactivated) authTransient=YES }"]

Is it vulnerable?

- Yes!
- The NFC controller handles all the transaction when enabled.
- However either the user has to authorize the payment with touch ID or passcode
- Or using a jailbroken device that has malware that has enabled the transaction
- Additionally any purchase over the contactless limit will be verified through “Consumer Device Cardholder Verification Method” (CDCVM)

https://support.apple.com/en-au/HT202527



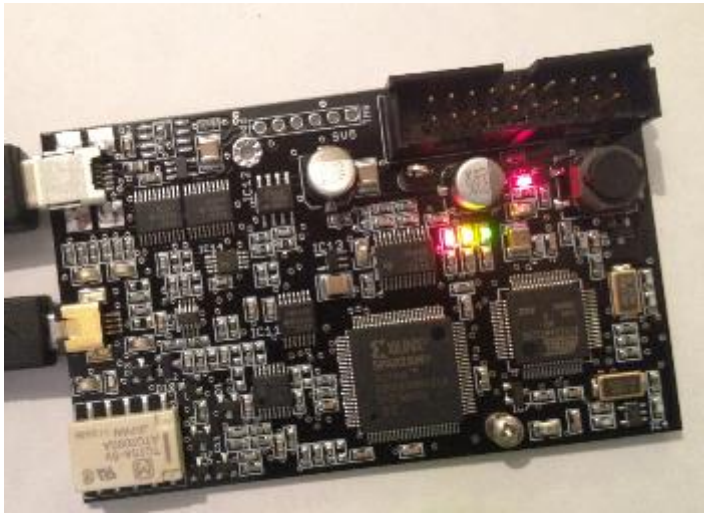
Tools Used

UBM Tech

ACR-122U

Around AUD$60 off ebay
Reads lost of stuff.
Fickle – loves to crash, horrible drivers
Can be made to support card emulation
Good to get started understanding stuff
Lots of limitations – like limited APDU length(~260 bytes),
Stuck with what the interface chip gives you.
No command chaining support (at least in RFIDIOT)

Proxmark 3



- Available for around US$200
- Supports 125/134KHz, 13.56MHz.
- FPGA handles raw signals,
- ARM higher protocol stuff
- Super powerful/Super Painful
- API is a bit hairy
- Lots of bugs! But good development community.

My fork of proxmark to handle EMV Stuff

https://github.com/peterfillmore/proxmark3

Android Phones with NFC

- Prior to 4.4.4 (KitKat) Card Emulation not officially supported. But Cyanogen mod lets you.
- NXP chip supports emulation but not in official AOSP ☹, watch out for pre 2013 android NFC phones
- Broadcom chip does, which was added in Nexus 4, Samsung Galaxy S4 etc.
- Better then ACR-122U cos its less buggy – but limited to chip support stuff – can't spoof UID – limited by internal buffer lengths (2472 in Nexus4).


black hat
USA 2015
Software I've developed
UBM Tech

Additions to RFIDiot

https://github.com/peterfillmore/RFIDIOt

Added on scripts to perform contactless transactions for MasterCard and VISA cards

MasterCard:

MagStripe	`$python ChAP-paypass.py -dv -C MSR`
M/Chip	`$python ChAP-paypass.py -dv -C MCHIP`

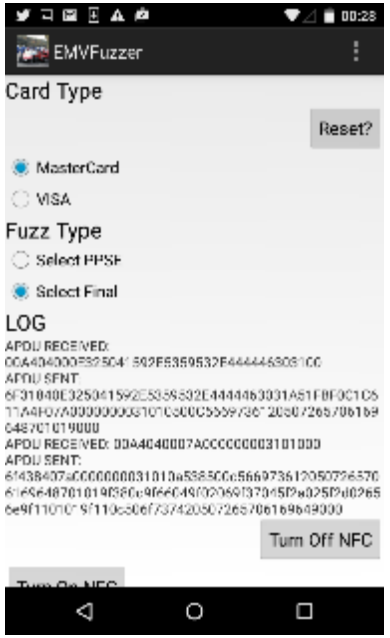
VISA:

dCVV	`$python ChAP-paywave.py -dv -C dCVV`
CVN17	`$python ChAP-paywave.py -dv -C CVN17`
fDDA0	`$python ChAP-paywave.py -dv -C fDDA0`
EMV	`$python ChAP-paywave.py -dv -C EMV`

NFC Fuzzing

https://github.com/peterfillmore/EMVFuzzer

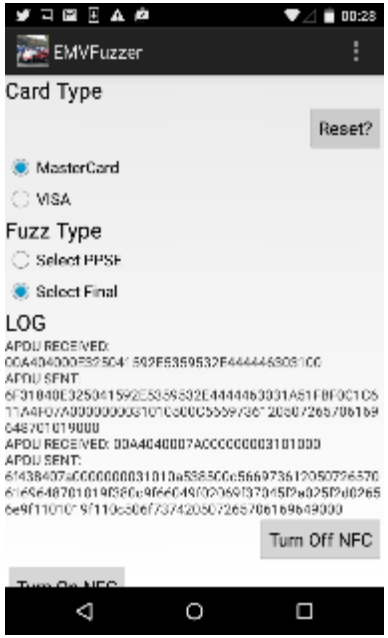
- Work in progress
- Uses Sully generated text files as input
- Requires a rooted phone – need to programmatically power cycle the NFC from the command line.
- I want to try and incorporate this into a Better solution – feel free to fork!

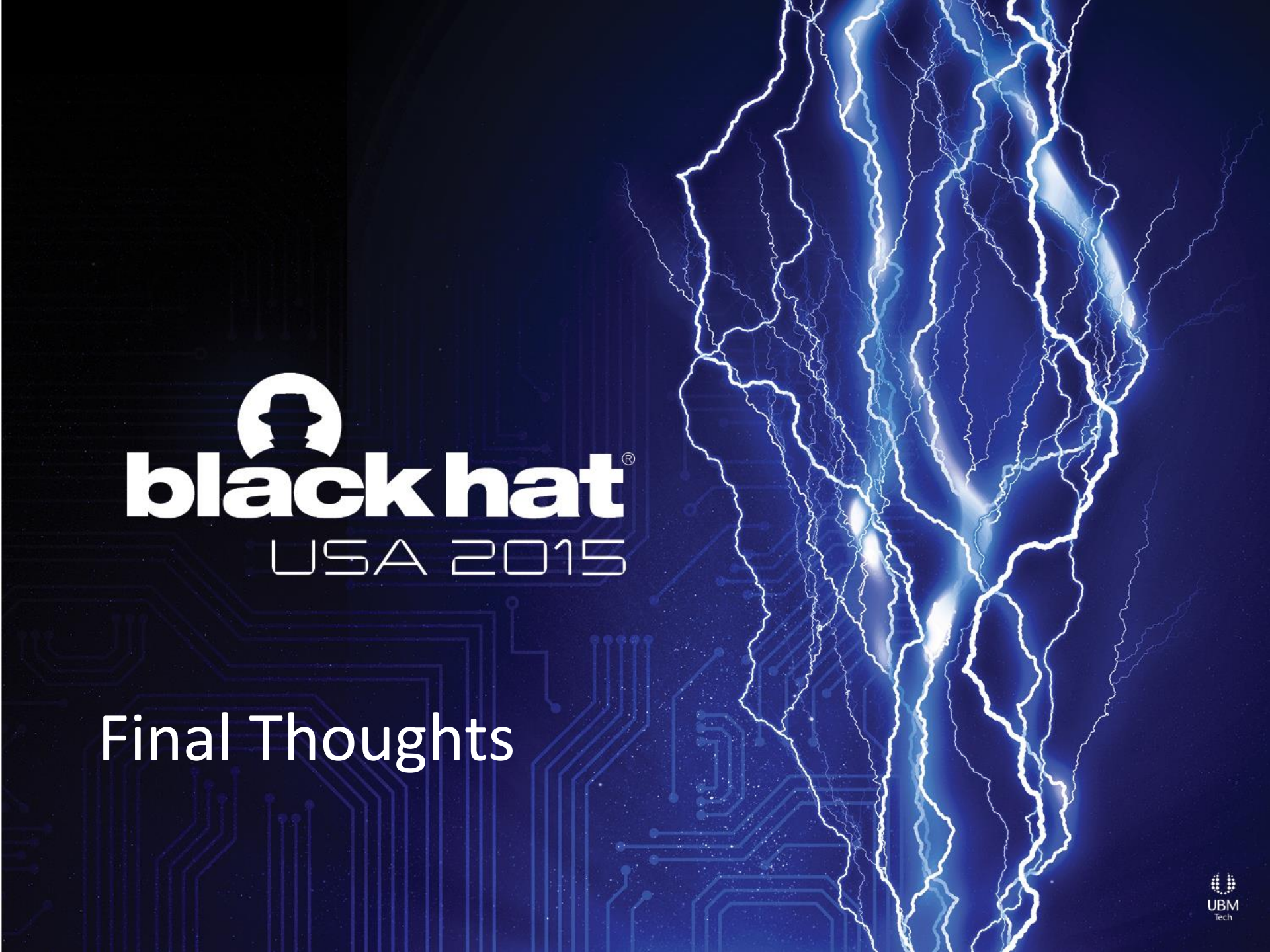


NFC Fuzzing

https://github.com/peterfillmore/EMVFuzzer

- Work in progress
- Uses Sully generated text files as input
- Requires a rooted phone – need to programmatically power cycle the NFC from the command line.
- I want to try and incorporate this into a better solution
- Feel free to fork!





Final Thoughts

Thanks

Pwpiwi@proxmark3 – Putting up with my complaining and fixing the Proxmark3 code
Adam Laurie for writing the RFIDidiot Tool – major help in learning this stuff.
Android team for adding HCE and allowing developers to access NFC Hardware
iOS hackers for developing awesome tools – you know who they are.

Credits and References

“Don’t Stand So Close To Me, An analysis of the NFC attack surface” – Charlie Miller 2012
“PinPadPwn” – Nils & Rafael Dominguez Vega Pin Pads, 2012
“Credit Card Fraud - The Contactless Generation” Kristian Paget, 2012
“Mission Mpossible” –Nils and Jon Butler 2013
“Cloning Credit Cards: A combined pre-play and downgrade attack on EMV Contactless” - Michael Roland 2013
Standards - http://www.emvco.com
Utilities - http://www.emvlab.org/tlvutils/
http://www.cl.cam.ac.uk/research/security/

Key Takeaways

- You can't clone cards (economically)
- You can clone transactions
- Legacy support reduces EMV security
- Random numbers aint random.
- Current standards do not mitigate these attacks sufficiently
- EMV Terminals and software are a huge worry
- ApplePay is a solid implementation of existing technologies.