Chip and Skim: Cloning EMV cards with the pre-play attack

Mike Bond, Omar Choudary, Steven J. Murdoch, Sergei Skorobogatov, Ross Anderson

Computer Laboratory
EMV – leading system for payments across the world

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.
EMV – introduced to remove magstripe counterfeiting

- EMV uses CHIP & PIN
- Should protect against card cloning and abuse
- Should decrease fraud
EMV is not totally secure in practice

- We discovered 2 important flaws in EMV
  - engineering flaw
  - protocol flaw
- In practice these allow same effect as card cloning
  - we can perform a “CHIP & PIN” transaction without the original EMV card

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.
EMV protocol for POS/ATM
EMV protocol – online authorisation

D = \{\text{Amount, Country, Date, UN, …}\}

REQ = \{\text{UN, ATC, IAD, …}\}, \text{ AUTH REQ} = \text{MAC}_K(D, \text{ ATC}, \text{ IAD})

RESP = \{\text{OK/BAD}\}, \text{ AUTH RESP} = \text{MAC}_K(\text{RESP, AUTH REQ, …})

UN = \text{Unpredictable Number}

ATC = \text{Application Transaction Counter}

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.
Evidence from real data: UN is a counter!

<table>
<thead>
<tr>
<th>Time</th>
<th>UN</th>
</tr>
</thead>
<tbody>
<tr>
<td>10:37:24</td>
<td>F1246E04</td>
</tr>
<tr>
<td>10:37:59</td>
<td>F1241354</td>
</tr>
<tr>
<td>10:38:34</td>
<td>F1244328</td>
</tr>
<tr>
<td>10:39:08</td>
<td>F1247348</td>
</tr>
</tbody>
</table>

- 17 bits fixed
- 15 bits seem to follow a linear counter

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.
Evidence from real data: UN is a counter!

<table>
<thead>
<tr>
<th>Time</th>
<th>UN</th>
</tr>
</thead>
<tbody>
<tr>
<td>10:37:24</td>
<td>F1246E04</td>
</tr>
<tr>
<td>10:37:59</td>
<td>F1241354</td>
</tr>
<tr>
<td>10:38:34</td>
<td>F1244328</td>
</tr>
<tr>
<td>10:39:08</td>
<td>F1247348</td>
</tr>
</tbody>
</table>

- 17 bits fixed
- 15 bits seem to follow a linear counter

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.
No terminal ID

\[ D = \{ \text{Amount, Country, Date, UN, } \ldots \} \]

\[ \text{REQ} = \{ \text{UN, ATC, IAD, } \ldots \}, \ \text{AUTH REQ} = \text{MAC}_K(D, \text{ATC, IAD}) \]

\[ \text{RESP} = \{ \text{OK/BAD} \}, \ \text{AUTH RESP} = \text{MAC}_K(\text{RESP, AUTH REQ, } \ldots) \]
Pre-play attack: exploit predictable UN

Step 1: Skim PIN & data for set of UNs

<table>
<thead>
<tr>
<th>ID</th>
<th>UN</th>
<th>AUTH REQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>xx</td>
<td>aa</td>
</tr>
<tr>
<td>2</td>
<td>yy</td>
<td>bb</td>
</tr>
</tbody>
</table>

\[ D_1 = \{\text{Amount, Country, Date, UN}_1, \ldots\} \]

\[ \text{AUTH REQ}_1 \]

\[ D_2 = \{\text{Amount, Country, Date, UN}_2, \ldots\} \]

\[ \text{AUTH REQ}_2 \]

\[ \vdots \]
Pre-play attack: exploit predictable UN

Step 2: replay data to get diamond

D={Amount, Country, Date, UN, …}

Replay from table of skimmed data

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.
Can we find weak RNGs?

- Previous EMV specs only required 4 consecutive UNs to be different
  - a counter would work better than a secure TRNG
- We decided to find out …
Searching for weak RNG: using ATM logger
Searching for weak RNG: using ATM logger

Microchip PIC18F24K22 0.5mm UQFN
Searching for weak RNG: using ATM logger
Searching for weak RNG: using ATM logger
Searching for weak RNG: using ATM logger

Ready to go

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.
Searching for weak RNG: using ATM logger

<table>
<thead>
<tr>
<th>Characteristic C (5 bits fixed):</th>
<th>ATM1</th>
<th>ATM2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Third nibble is 0</td>
<td>690d4df2</td>
<td>6f0c2d04</td>
</tr>
<tr>
<td>First bit is 0</td>
<td>69053549</td>
<td>580fc7d6</td>
</tr>
<tr>
<td>11 ATMs had same output</td>
<td>660341c7</td>
<td>4906e840</td>
</tr>
<tr>
<td>Possibly due to common lib</td>
<td>5e0fc8f2</td>
<td>46099187</td>
</tr>
</tbody>
</table>

Table II

<table>
<thead>
<tr>
<th>ATMs</th>
<th>Stronger RNGs</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATM1</td>
<td>49868033</td>
</tr>
<tr>
<td>ATM1</td>
<td>293FBA89</td>
</tr>
<tr>
<td>ATM1</td>
<td>39EB1E19</td>
</tr>
<tr>
<td>ATM1</td>
<td>2A26982F</td>
</tr>
<tr>
<td>ATM1</td>
<td>7C0AF071</td>
</tr>
<tr>
<td>ATM1</td>
<td>650155D7</td>
</tr>
<tr>
<td>ATM2</td>
<td>4906e840</td>
</tr>
<tr>
<td>ATM2</td>
<td>46099187</td>
</tr>
<tr>
<td>ATM2</td>
<td>4906e840</td>
</tr>
<tr>
<td>ATM2</td>
<td>46099187</td>
</tr>
<tr>
<td>ATM2</td>
<td>46099187</td>
</tr>
</tbody>
</table>

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.
Searching for weak RNG: using SmartCard Detective
Searching for weak RNG: using SmartCard Detective

- Results from local POS
- First bit still 0, but otherwise could not find clear pattern

<table>
<thead>
<tr>
<th>Stronger RNGs</th>
</tr>
</thead>
<tbody>
<tr>
<td>POS1 013A8CE2</td>
</tr>
<tr>
<td>POS1 01FB2C16</td>
</tr>
<tr>
<td>POS1 2A26982F</td>
</tr>
<tr>
<td>POS1 39EB1E19</td>
</tr>
<tr>
<td>POS1 293FBA89</td>
</tr>
<tr>
<td>POS1 49868033</td>
</tr>
</tbody>
</table>

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.
The deeper problem: We can use our own UN!

UN generated by Terminal (POS, ATM), not issuer!

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.
The pre-play attack by tampering UN

Step 1: get PIN & data for a chosen UN

D={Amount, Country, Date, UN, ...}

\[ \text{AUTH REQ} = \text{MAC}_K(D, ATC, IAD) \]
The pre-play attack by tampering UN

Step 2: replay data & tamper UN to get diamond

D'={Amount, Country, Date, UN', …}

AUTH REQ=MAC_k(UN, …)

Evil link

D'={…, UN', …}, AUTH REQ

RESP, AUTH RESP

D={…, UN, …}, AUTH REQ

RESP, AUTH RESP

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.
Can we actually modify the UN sent by the terminal?

Likely. It depends on bank, country, regulator, etc.

syntax, semantic: ISO 8583, ISO 20022, ...
transport: AS2, AS3, SWIFT, FTP, IFX, ...

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.
Can we actually modify the UN sent by the terminal?

Likely. It depends on bank, country, regulator, etc.
Can we actually modify the UN sent by the terminal?

Likely. It depends on bank, country, regulator, etc.
Can we actually modify the UN sent by the terminal?

… emergence of new functionality such as authentication methods …

[VISA "Transactions Acceptance Device Guide" 2013]

Practical example: Maxwell Parsons in UK

- injected data into the bank system (reverse transactions), stealing £2,560,000 in 7 months
Can we actually modify the UN sent by the terminal?

- Even if authentication is enabled, there are options:
  - Malware infection of POS/ATM
  - Supply chain attacks (react on covert signal)
  - Collusive or dishonest merchant
It is a protocol problem

- Issuer relies on fresh UN for transaction
- But UN generated by terminal
- Terminal might not have incentive to cooperate
Card authentication via DDA does not help

Start transaction

Card data records

Signature over data records

Same UN for both DDA and ARQC => skim signature as well

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.
PIN verification does not help either.

Simply skim PIN during step (1) of attack, or lie [Oakland ’10]
Blocking a pre-play attack using the Transaction Certificate (TC)

\[ D = \{ \text{Amount, Country, Date, } \text{UN}, \ldots \} \]

\[ \text{REQ} = \{ \text{UN, ATC, IAD}, \ldots \} \]
\[ \text{AUTH REQ} = \text{MAC}_K(D, \text{ATC}, \text{IAD}) \]

\[ \text{RESP} = \{ \text{OK/BAD} \} \]
\[ \text{AUTH RESP} = \text{MAC}_K(\text{RESP, AUTH REQ}, \ldots) \]
Blocking a pre-play attack using the Transaction Certificate (TC)

**External Authenticate**

- **D** = \{Amount, Country, Date, UN, \ldots\}
- **REQ** = \{UN, ATC, IAD, \ldots\}, **AUTH REQ** = \text{MAC}_K(D, ATC, IAD)
- **RESP** = \{OK/BAD\}, **AUTH RESP** = \text{MAC}_K(RESP, AUTH REQ, \ldots)

**Final exchange**

- **TC** = \text{MAC}_K(D', ATC, IAD)
- **RESP**, **AUTH RESP**

---

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.
Importance of TC not taken into consideration

• Problem 1: TC not routinely kept
  • not needed for clearance, may be discarded
  • only needed to ensure that card does not need to go online (issuer) at next transaction and to provide liability protection to acquirer

• Problem 2: TC may be sent within 24 hours
  • good: send daily TC batches to reduce #messages
  • bad: this leaves system open to pre-play attack
What could EMV do

- Fix RNG everywhere
- Mandatory authentication between all parties
- Request terminal to keep log of UNs for disputes
- Mandatory check or at least storage of TC for every transaction
  - **TC should be the only probative evidence** in case of disputes
- For high-value transactions, check TC before customer leaves the shop!
Conclusions

• We discovered a deep and important flaw in the EMV implementation, indistinguishable from card cloning

• Issuer relies on freshness, but this is generated by another party
  • Changing the protocol is unlikely to happen
  • Practical solution is mandatory use or retention of TC

• Lack of understanding and deliberate overstatement of security may lead to customers being defrauded

• Bank regulators should prohibit EMV liability shift
Questions?

Speaker: Omar Choudary
Co-authors: Mike Bond, Steven Murdoch, Sergei Skorobogatov, Ross Anderson
E-mail: firstname.lastname@cl.cam.ac.uk

Security Group, Computer Laboratory

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.
Industry response

• RNG attack disclosed in early 2012

• Banks and payment switches acknowledge receipt

• April 2012 EMVCo publishes update on RNG

• However, ATMs and terminals still vulnerable to malware

  • industry insider mentioned Malta’s case may involve ATM malware

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.
ATM reverse engineering

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.
Bank losses by kind

Fraud levels on UK-issued payments cards

Chip and Skim. Bond, Choudary, Murdoch, Skorobogatov, Anderson.