Overview of Contactless Payment Cards

Peter Fillmore

July 20, 2015

Blackhat USA 2015

Introduction

Contactless payments have exploded in popularity over the last 10 years with various schemes being popular in many domestic markets. Internationally the dominate contactless payments standard has have been from the traditional major international credit card companies and are based around the EMV chip card protocols. Utilisation of these protocols has allowed for the cards and terminals to support contactless payment technologies with minimal changes to existing devices in the field. However due to these cards utilising the EMV protocols; they also carry many of the existing flaws in the existing systems.

Key Identified flaws in EMV systems include:

- Ability to downgrade authorization method. [1]
- Insufficient replay prevention. [3, 10]
- Lack of Man In The Middle protection. [7]
- No protection against relay attacks. [4, 6]
- Insecure generation of random numbers. [5]
- Plaintext transmission of sensitive data. [2]

Additionally EMV software used in commercial products has been shown to be vulnerable to basic logical attacks. [9, 8]

Many of these flaws can be prevented through prevention on the terminal side; proper monitoring by payment processors to ensure that transactions are using secure defaults and have not been tampered with.

Contactless implementations of the EMV standard frequently support simplified processing flows to ensure compatibility with older systems and reduce the time it takes to perform a transaction. These alterations affect the ability of existing protections to prevent fraud on cards.

Additionally contactless payment cards use altered EMV processing flows compared to the contact variety. This is due to the need for compatibility with older payment networks and for
transactions to be completed quickly. This need introduces other vulnerabilities to the system in which they are used.

An overview of the EMV standards

The EMV standard is a publicly available set of documents available at [http://www.emvco.com](http://www.emvco.com). These standards cover the physical and logical properties of the cards, terminals and protocols used in the EMV world. Cards and terminals are certified for use by EMVco to ensure they interoperate with each other safely and reliably. Certification is performed through independent laboratories internationally. Terminals and Cards operate in a “Master/Session” configuration; where the Terminal controls all steps of the transaction with the card providing responses or calculations as necessary.

Contactless Banking Cards

The major standard used for contactless banking cards is ISO14443. This standard provides how cards physically and logically communicate with a reader. It is separated into two card types - Type A and Type B cards. This is due card technologies being initially developed in private companies which have been since incorporated into the ISO standardization process.

Type A and B are separated with how the card physically communicates (different modulation schemes and binary encodings) and initialization processes. The higher level data is formatted the same.

The ISO14443 standard is separated into 4 sections:
1. Physical Characteristics
2. Radio frequency power and signal interface
3. Initialization and anticollision
4. Transmission protocol

Mastercard Paypass

Mastercards implementation of contactless payments is branded “PayPass”. It is separated into two modes M/Chip and MagStripe.
M/Chip

This is MasterCard’s contactless EMV implementation. It follows the full EMV standard fairly closely - emitting the PIN capture phase.

MagStripe

This is the legacy mode provided for use in environments that cannot yet support the additional messaging EMV transactions requires. It utilises the “Compute Cryptographic Checksum” command to generate dynamic Card Verification Codes (CVCs) for returned tracks. The terminal will then construct a transaction specific track from the returned data and send it to the payment network for processing.

VISA Paywave

VISA’s implementation of contactless payment is branded “Paywave”. It is separated into four operating modes - VSDC, qVSDC, CVN17 and dCVV.
VSDC

Visa Smart Debit Credit is a full implementation of the Combined DDA and TC Authorization (CDA) EMV standard. It is not widely supported by current cards due to customers having to keep the card in the contactless field for the complete transaction.

qVSDC

Quick Visa Smart Debit Credit is a cut-down version of the EMV protocol for use in contactless environments. It removes the “Generate AC” step from the EMV protocol to allow for customers to remove their cards from the field prior to authentication from the payment network.
CVN17

Card Verification Number 17 (CVN17) is a mode provided for use in environments that do not yet support EMV messaging. It relies on the “Get Processing Options” command to generate a cryptogram which is transmitted with the track data to the payment network. This mode is a replacement for the dCVV method previously used for older environments.

Figure 5: VISA CVN17 Transaction Flow

dCVV

Dynamic CVV mode is an obsolete mode for use in environments which does not support EMV messaging. dCVV mode utilises the “Read Record” function to generate and return a dynamic CVV in the track data. However many cards will implement a static value for the CVV (which is permitted by the VISA standards).

Figure 6: VISA dCVV Transaction Flow

ApplePay

ApplePay was announced with the release of the iPhone 6 device. This has been developed in conjunction with the major card brands to provide payments utilising special hardware on the device. It can be separated into two methods - Card Not Present (Online) and Card Present (Contactless) transactions.

Accessing List of approved cards in a secure element

```
ps aux | grep "passd" // get PID of the passd daemon
cycript -p {PID of passd}
mySE = [[PDSecureElement alloc] init] // initialize a "SecureElement" object
mySE.secureElementCards // dump the contents
```
References

References


Appendix A - EMV Cryptographic Keys

EMV Cards contain a number of private and public cryptographic keys. Cards and terminals use Triple-DES and RSA ciphers for encryption and signing as well as the SHA-1 standard for hashing.

Symmetric Keys:

<table>
<thead>
<tr>
<th>Key</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KDcvc3</td>
<td>ICC Derived Key for CVC3 Generation</td>
<td>Symmetric Key used for generating the CVC3</td>
</tr>
<tr>
<td>MKac</td>
<td>ICC Application Cryptogram Master Key</td>
<td>Symmetric Key used to derive the session key for generation of the Application Cryptogram</td>
</tr>
<tr>
<td>SKac</td>
<td>ICC Application Cryptogram Session Key</td>
<td>Symmetric Key used to generate the Application Cryptogram</td>
</tr>
</tbody>
</table>

Table 1: Symmetric Keys in an EMV card
### RSA Keys

<table>
<thead>
<tr>
<th>Key</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pi</td>
<td>Issuer Public Key</td>
<td>Used to verify signature on static card data.</td>
</tr>
<tr>
<td>Sic</td>
<td>ICC Private Key</td>
<td>Generates signature on dynamic data</td>
</tr>
<tr>
<td>Pic</td>
<td>ICC Public Key</td>
<td>Used by Terminal for verification of cards signature on dynamic data</td>
</tr>
</tbody>
</table>

Table 2: RSA Keys present in an EMV card

### Appendix B - BER-TLV Formatting

BER-TLV is the formatting used for EMV commands and data. It is defined in ISO7816.

**Tag**

Tag indicates the what the data represents in the Value field.

- The top 2 bits indicate the class of the tag - Universal, Application, Context-Specific and Private.
- Bit 6 determines XXX while the lower 5 bits are reserved for the tag number.
- If the Tag Number is 31 (i.e all 1s) then the tag number is stored in subsequent bytes after the initial byte.
- If the high bit is set in the subsequent bytes - then we keep reading bytes as needed.
- All tags used for financial messaging will be at most 2 bytes long

**Length**

Here we tell the parser how long the data is. It can take two types - a short form and a long form.

- The short form means the length is 1 byte long - and can represent a maximum value length of 127 bytes. The long form is used for data of greater then 127 bytes - with no upper limit of the length.

**Value**

Value is just this - there is no limit as to the values that can be represented here.

**TLV Examples**

- a TLV of “8F0108” is a Tag of 8F, Length of 1, Value of “08”
- a TLV of “9F320103” is a Tag of 9F32, Length of 1, Value of “03”
- a TLV of “7081C95F20...” is a Tag of 70, 1 Length byte, Length of C9, and value of “5F20..”
Appendix C - Application Protocol Data Unit (APDU)

This is how a command to the card and responses to the terminal are formatted. Commands are sent from the terminal using the Command APDU format and responses are sent using the Response APDU format. Response codes are formatted in the last two bytes of the Response APDU (SW1 and SW2); a response of “9000” indicates successful processing while anything other then this value is an error.

<table>
<thead>
<tr>
<th>Byte</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>&lt;VAR&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Desc</td>
<td>CLA</td>
<td>INS</td>
<td>P1</td>
<td>P2</td>
<td>Le</td>
<td>data</td>
</tr>
<tr>
<td>Class</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instruction</td>
<td>Parameter Byte 1</td>
<td>Parameter Byte 2</td>
<td>Data Length</td>
<td>Expected Response Length (0 for contactless)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Command APDU format

<table>
<thead>
<tr>
<th>Byte</th>
<th>&lt;VAR&gt;</th>
<th>&lt;VAR&gt;+1</th>
<th>&lt;VAR&gt;+2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Response Data</td>
<td>SW1</td>
<td>SW2</td>
</tr>
</tbody>
</table>

Table 4: Response APDU Format

Appendix D - Common EMV Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>CLA</th>
<th>INS</th>
<th>P1</th>
<th>P2</th>
<th>Le</th>
<th>Data</th>
<th>Le</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>00</td>
<td>A4</td>
<td>04</td>
<td>00/02</td>
<td>05-10</td>
<td>AID</td>
<td>00</td>
</tr>
<tr>
<td>Get Processing Options</td>
<td>80</td>
<td>A8</td>
<td>00</td>
<td>00</td>
<td>&lt;VAR&gt;</td>
<td>PDOL data</td>
<td>00</td>
</tr>
<tr>
<td>Read Record</td>
<td>00</td>
<td>B2</td>
<td>Record Number</td>
<td>SFI</td>
<td>X</td>
<td>X</td>
<td>00</td>
</tr>
<tr>
<td>Compute Cryptographic Checksum</td>
<td>80</td>
<td>2A</td>
<td>8E</td>
<td>80</td>
<td>&lt;VAR&gt;</td>
<td>UDOL data</td>
<td>00</td>
</tr>
<tr>
<td>Generate Application Cryptogram</td>
<td>80</td>
<td>AE</td>
<td>Control Parameter</td>
<td>00</td>
<td>&lt;VAR&gt;</td>
<td>CDOL data</td>
<td>00</td>
</tr>
</tbody>
</table>

Table 5: Common EMV commands
Select
Used select the payment application stored on the card.

Get Processing Options
Initiates the transaction within the selected card.

Compute Cryptographic Checksum
Used to generate the dynamic CVC codes in MasterCard transactions. Also returns the Application Transaction Counter(ATC) and increments it.

Generate Application Cryptogram
Takes in various fields to identify the terminal and transaction; and returns a signature of these values (and internal ones) for verification by the issuer.

Read Record
Returns data stored in the card according to its record number and “Short File Indicator”(SFI).