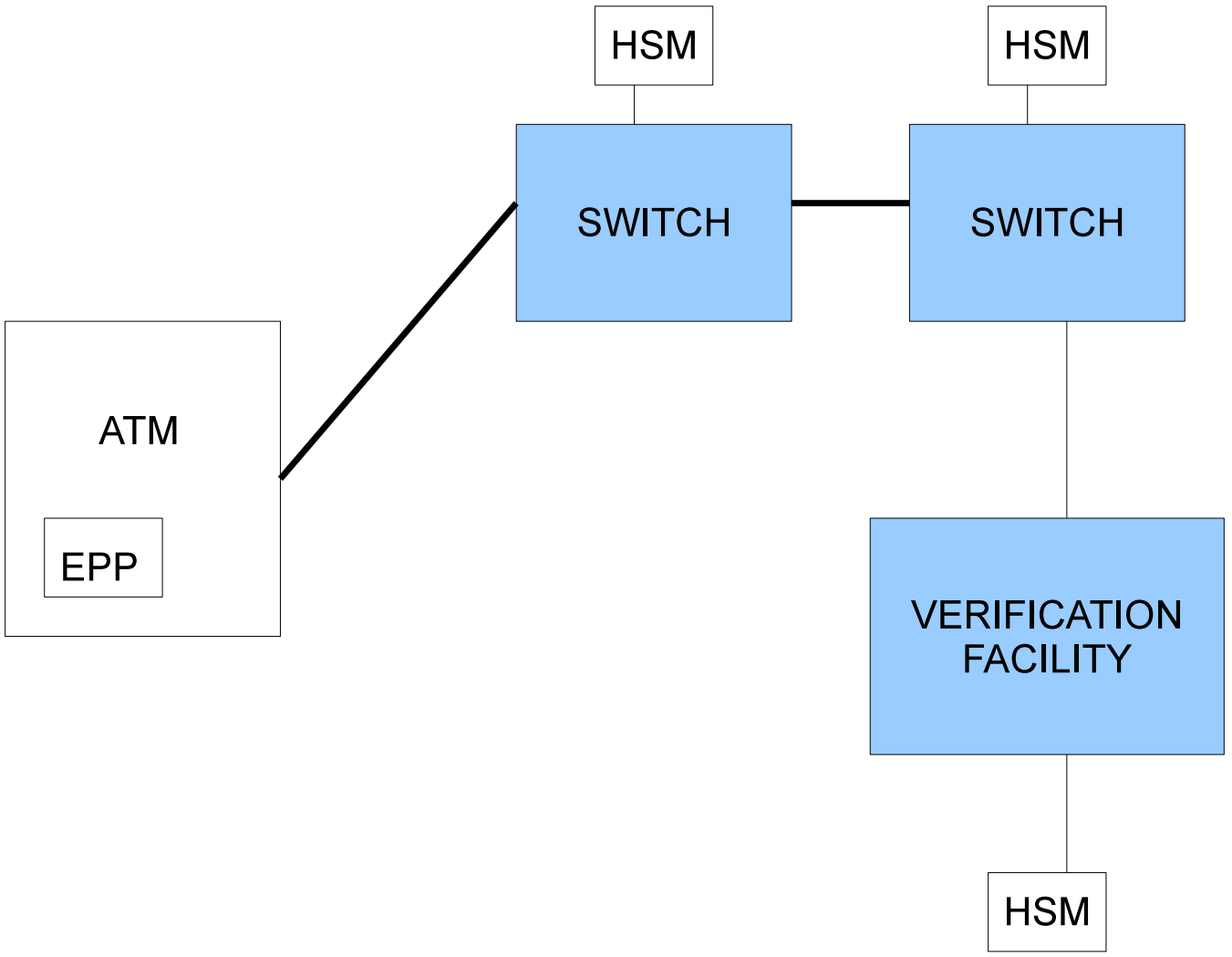
Improving PIN Processing API Security

Riccardo Focardi, Flaminia Luccio and Graham Steel

ASA-3 Long Island, NY

PIN Verification in ATM Networks



IBM 3624 PIN Verification

$$\mathrm{PIN} = f(\{g(\mathrm{PAN})\}_{\mathrm{PDK}})$$

g scheme-specific

f choose leftmost 4 hex digits, decimalise

IBM 3624 PIN Verification

$$\mathsf{PIN} = f(\{g(\mathsf{PAN})\}_{\mathsf{PDK}})$$

g scheme-specific

f choose leftmost 4 hex digits, decimalise

DecTab

```
0123456789ABCDEF
0123456789012345
```

IBM 3624 PIN Verification

$$\mathsf{PIN} = f(\{g(\mathsf{PAN})\}_{\mathsf{PDK}})$$

g scheme-specific

f choose leftmost 4 hex digits, decimalise

DecTab

```
0123456789ABCDEF
0123456789012345
```

$\{\mathsf{PIN}\}_{\mathsf{K}}, \mathsf{PAN}, \mathsf{Dectab} \quad \rightarrow$

yes/no $\quad \leftarrow$

K, PDK

IBM 3624 PIN Verification

$$\mathsf{PIN} = f(\{g(\mathsf{PAN})\}_{\mathsf{PDK}})$$

g scheme-specific

f choose leftmost 4 hex digits, decimalise

DecTab

```
0123456789ABCDEF
0123456789012345
```

$\{\mathsf{PIN}\}_{\mathsf{K}}, \mathsf{PAN}, \mathsf{Dectab} \quad \rightarrow$

K, PDK

yes/no $\quad \leftarrow$

```
0123456789ABCDEF
1123456789112345
```

IBM 3624 PIN Verification with Offset

$$\text{PIN} = f(\{g(\text{PAN})\}_{\text{PDK}}) \oplus OFFSET$$

OFFSET accommodates user chosen PIN

IBM 3624 PIN Verification with Offset

$$\text{PIN} = f(\{g(\text{PAN})\}_{\text{PDK}}) \oplus OFFSET$$

OFFSET accommodates user chosen PIN

Change DecTab at position i

Cycle through offsets until PIN verifies OK.

Identify location of digits i in PIN

IBM 3624 PIN Verification with Offset

$$\text{PIN} = f(\{g(\text{PAN})\}_{\text{PDK}}) \oplus OFFSET$$

OFFSET accommodates user chosen PIN

Change DecTab at position i

Cycle through offsets until PIN verifies OK.

Identify location of digits i in PIN

On average attack requires 16.145 API calls

PIN Cracking Attacks

- Dectab attacks
- Reformatting attacks
- Check value attack
- Calculate offset attack
- Competing verification algorithms attack

All require attacker to make ‘tweaked’ queries to HSM

History of HSM PIN Cracking

- Clulow, Prism TR 2001, RSA Europe 2002
- Bond, U. Cambridge TR 2002
- Anderson talk at Security Protocols Workshop, 2003
- Steel, TCS 2006
- Berkman and Ostrovsky, FC 2007
- Mannan, ASA-2 and FC 2008

History of HSM PIN Cracking

- Clulow, Prism TR 2001, RSA Europe 2002
- Bond, U. Cambridge TR 2002
- Anderson talk at Security Protocols Workshop, 2003
- Steel, TCS 2006
- Berkman and Ostrovsky, FC 2007
- Mannan, ASA-2 and FC 2008
- Sartin, Verizon Data Breach Report and Wired ‘Threat Level’ Blog Interview, 2009:

  “The most common method Sartin says criminals are using to get the PINs is to fool the application programming interface (or API) of the hardware security module in to helping them”

Theory Behind Fix

Language based security

Theory Behind Fix

Language based security

- Multilevel view - high and low security

Theory Behind Fix

Language based security

- Multilevel view - high and low security
- Non-interference - no ‘flow’ from high to low

Theory Behind Fix

Language based security

- Multilevel view - high and low security
- Non-interference - no ‘flow’ from high to low
- Declassification - wrt a policy

Theory Behind Fix

Language based security

- Multilevel view - high and low security
- Non-interference - no ‘flow’ from high to low
- Declassification - wrt a policy
- Robustness - introduces integrity

Theory Behind Fix

Language based security

- Multilevel view - high and low security
- Non-interference - no ‘flow’ from high to low
- Declassification - wrt a policy
- Robustness - introduces integrity
- Endorsement - allows integrity to be raised

Theory Behind Fix

Language based security

- Multilevel view - high and low security
- Non-interference - no ‘flow’ from high to low
- Declassification - wrt a policy
- Robustness - introduces integrity
- Endorsement - allows integrity to be raised

We introduce cryptographically assured endorsement (ESORICS 2009)

Verification Function

PIN_V($PAN, EPB, len, offset, vdata, dectab$) {

$x_1 := \mathsf{enc}_{pdk}(vdata)$;

$x_2 := \mathsf{left}(len, x_1)$;

$x_3 := \mathsf{decimalize}(dectab, x_2)$;

$x_4 := \mathsf{sum\_mod10}(x_3, offset)$;

$x_5 := \mathsf{dec}_k(EPB)$;

$x_6 := \mathsf{fcheck}(x_5)$;

if $(x_6 == \ "FAIL")$ then return$("format\ \ error")$;

if $(x_4 == x_6)$ then return$("PIN\ \ is\ \ correct")$;
else return$("PIN\ \ is\ \ wrong")$;

}

Fixed Function

```
PIN_V_M(PAN,EPB,len,offset,vdata,dectab,MAC) {
    // checking the MAC
    if  ( mac_ak(PAN,EPB,len,offset,vdata,dectab)
                                          == MAC )
```

$EPB' := EPB$; $len' := len$; $offset' := offset$;

$vdata' := vdata$; $dectab' := dectab$;

then return$(PIN_V(PAN, EPB', len', offset', vdata', dectab'))$;

```
    else
```

return$(\text{"}integrity\ \ violation\text{"})$;

```
}
```

Existing MAC

CVV/CVC - Card Verification Value(/Code)

Effectively a MAC of PAN, expiry date, some other data

Made with secret key

5 decimal digits

Written to magstripe, does not appear e.g. on POS receipts

Designed to make construction of fake cards more difficult

CVV Format

PAN	Exp date	Service code	0 pad
16 digits max	4 digits	3 digits	9 digits max
Block B1	Block B2		

2-part DES key K1, K2.

$$CVV_{hex} \quad := \quad enc(K1, dec(K2, enc(K1, (enc(K1, B1) \oplus B2))))$$

CVV’

Dectab	Offset/PVV	original CVV	0 pad
16 digits	4 digits	5 digits	7 digits
Block B1’	Block B2’		

Operation of Scheme

CVV’ is written onto card at issue time

CVV’ is sent along with trial PIN from each ATM transaction

Intermediate switches simply pass along the CVV’

At the verification facility, the supplied CVV’ is checked against the true derived value instead of full MAC

Evaluation - Advantages

- CVV' can be calculated in advance

  - can be written to magstripe track 2, just like CVV

- Existing infrastructure already passes track 2 through network

  - no need for costly changes to infrastructure

- Institutions can choose to upgrade individually

  - no need to await standardization

Evaluation - Disadvantages

- Low entropy of MAC allows brute force attack

  - though overhead for PIN cracking attacks considerably increased

- Does not address translation command attacks

  - that would require point to point MACs, bigger overhead

- Change needed to HSM software

  - maybe not a big deal

Evaluation - Disadvantages

- Low entropy of MAC allows brute force attack

  - though overhead for PIN cracking attacks considerably increased

- Does not address translation command attacks

  - that would require point to point MACs, bigger overhead

- Change needed to HSM software

  - maybe not a big deal

http://www.lsv.ens-cachan.fr/~steel/improving-PIN-security/