

Quantitative Analysis of Security APIs

Graham Steel



School of
informatics

PIN Blocks

64-bit strings to encode a guessed PIN for encryption

e.g.

VISA format 3

```
PPPPFFFFFFFFFFFFFF
```

ISO 9564 format 0

```
04PPPPFFFFFFFFFFFF
```

```
0000AAAAAAAAAAAAAA
```

ISO-0 Reformatting attack

(Clulow, 2003)

04PPPPFFFFFFFFFFFF

0000AAAAAAAAAAAAAA

Error check ($0 \leq P \leq 9$) leaks information

ISO-0 Reformatting attack

(Clulow, 2003)

```
04PPPPFFFFFFFFFFFF
```

```
0000AAAAAAAAAAAA
```

Error check ($0 \leq P \leq 9$) leaks information

Extend:

Masquerade ISO-0 as VISA format 3

```
0604PPPPFFFFFFFFFFFF
```

Can now uniquely determine digits

Dectab Attacks

Controls decimatisation of $\{ \text{pan} \}_{\text{pdk}}$

Cleartext input to verify commands

Dectab Attacks

Controls decimalisation of $\{ \text{pan} \}_{\text{pdk}}$

Cleartext input to verify commands

Standard

0123456789ABCDEF

0123456789012345

Dectab Attacks

Controls decimalisation of $\{ \text{pan} \}_{\text{pdk}}$

Cleartext input to verify commands

Standard

0123456789ABCDEF

0123456789012345

Attack 1

0123456789ABCDEF

1123456789112345

Dectab Attacks

Controls decimalisation of $\{ \text{pan} \}_{\text{pdk}}$

Cleartext input to verify commands

Standard

0123456789ABCDEF

0123456789012345

Attack 1

0123456789ABCDEF

1123456789112345

Alter offset to establish position of digits (Bond + Zieliński, 2003)

Formal Modelling

Take a customer configuration and an API spec. as input

Formal Modelling

Take a customer configuration and an API spec. as input

Using CLP, generate tree of all possible attacks

Formal Modelling

Take a customer configuration and an API spec. as input

Using CLP, generate tree of all possible attacks

Meta-logical predicates allow us to calculate transition probabilities

Formal Modelling

Take a customer configuration and an API spec. as input

Using CLP, generate tree of all possible attacks

Meta-logical predicates allow us to calculate transition probabilities

Apply PRISM (Kwiatkowska et. al, 2004)

Get minimum expected number of steps to determine PIN

Formal Modelling

Take a customer configuration and an API spec. as input

Using CLP, generate tree of all possible attacks

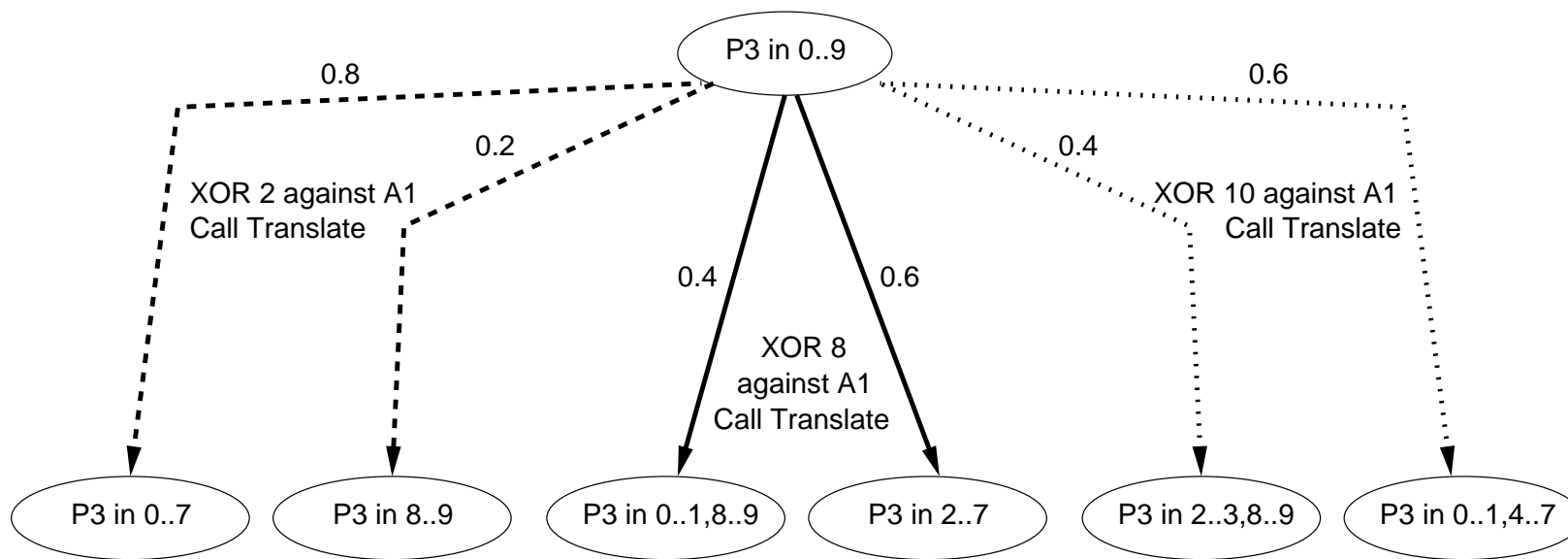
Meta-logical predicates allow us to calculate transition probabilities

Apply PRISM (Kwiatkowska et. al, 2004)

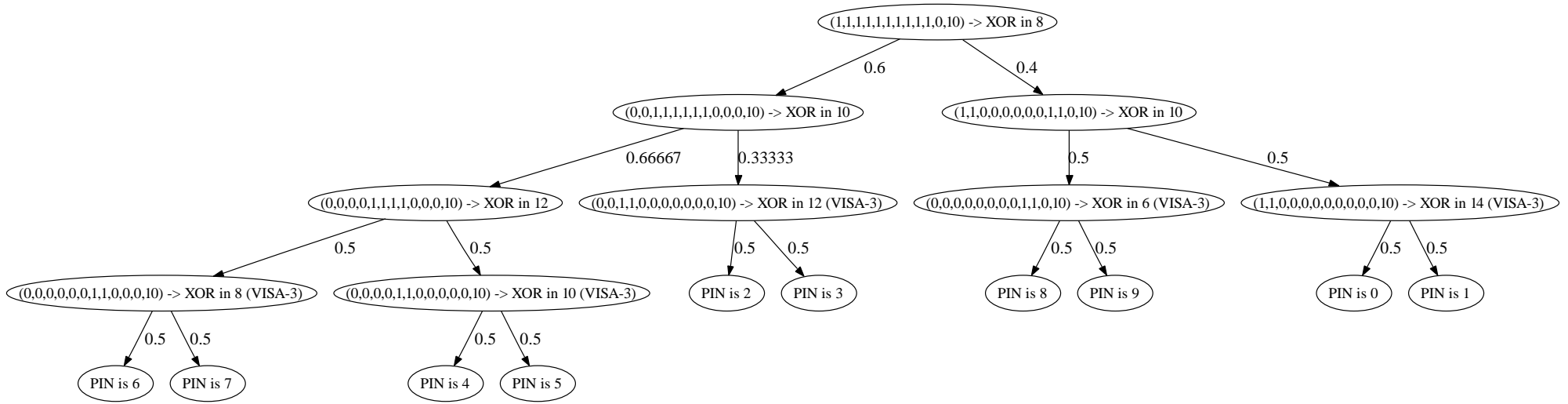
Get minimum expected number of steps to determine PIN

Generate tree for best attack

Attack Trees



Optimised Attack

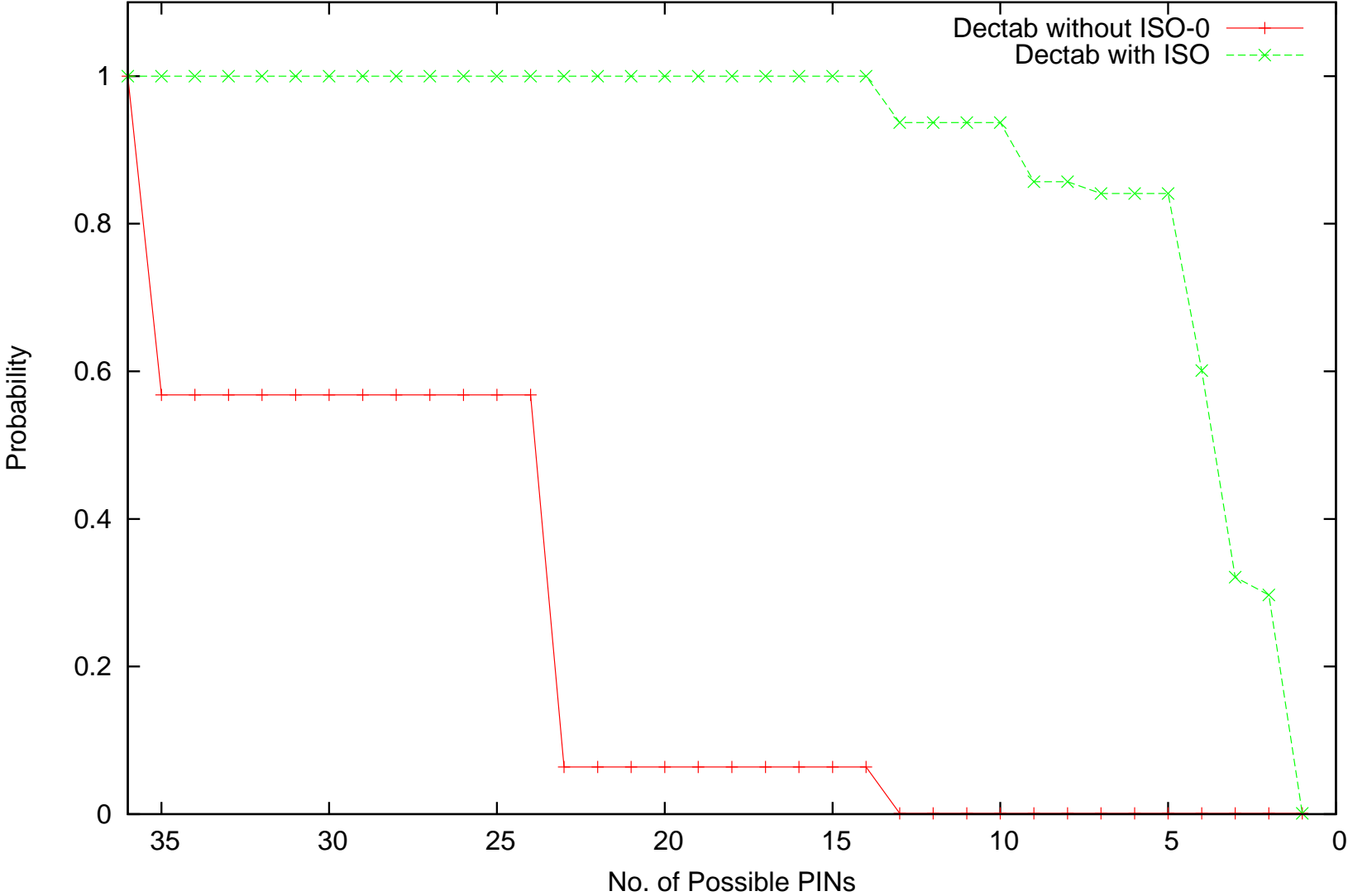


Results from Analysis (Generic API)

No.	Attack	$P(determined)$	$E(Steps)$
(1)	ISO-0 (extended)	1	13.6
(2)	Dectab	1	16.145
(3)	Dectab & ISO (restricted)	1	15.275

No.	Attack	$k = 400$	$k = 36$	$k = 24$	$k = 14$	$k = 1$
(4)	ISO-0 (restricted)	1	0	0	0	0
(5)	Dectab no offset	1	1	0.568	0.064	0.001
(6)	Dectab no offset & ISO-0 (restricted)	1	1	1	1	0.001

Performance of Dectab attack without offset



Parallel Key Search

Key test:

$$\{k\}_{km \oplus TYPE}, TYPE \rightarrow \{00000000\}_k$$

Parallel Key Search

Key test:

$$\{k\}_{km \oplus TYPE}, TYPE \rightarrow \{00000000\}_k$$

Obtain test values for large number of k

Parallel Key Search

Key test:

$$\{k\}_{km \oplus TYPE}, TYPE \rightarrow \{00000000\}_k$$

Obtain test values for large number of k

Offline, encrypt 00000000 under random r until collision occurs

Parallel Key Search

Key Generate:

$$y \xrightarrow{\text{new } n} \{n\}_{km \oplus y}$$

Parallel Key Search

Key Generate:

$$y \xrightarrow{\text{new } n} \{n\}_{km \oplus y}$$

Key test:

$$x, y \rightarrow \{0\}_{\text{dec}(x, km \oplus y)}$$

Parallel Key Search - 2

Offline generation:

new n
→ n

Parallel Key Search - 2

Offline generation:

$$\begin{array}{c} \text{new } n \\ \rightarrow \\ n \end{array}$$

Collision:

$$n, f(n), f(g(n')) \rightarrow g(n')$$

Example Parallel Key Search

Key Generate:

$$\text{exp} \xrightarrow{\text{new } n_1} \{n_1\}_{km \oplus \text{exp}}$$

Example Parallel Key Search

Key Generate:

$$\text{exp} \xrightarrow{\text{new } n_1} \{n_1\}_{km \oplus \text{exp}}$$

Key test:

$$\{n_1\}_{km \oplus \text{exp}}, \text{exp} \rightarrow \{0\}_{n_1}$$

Example Parallel Key Search - 2

Offline generation and encryption:

$$\begin{array}{ccc} & \xrightarrow{\text{new } n_2} & n_2 \\ n_2, 0 & \longrightarrow & \{0\}_{n_2} \end{array}$$

Example Parallel Key Search - 2

Offline generation and encryption:

$$\begin{array}{ccc} & \xrightarrow{\text{new } n_2} & n_2 \\ n_2, 0 & \longrightarrow & \{0\}_{n_2} \end{array}$$

Collision:

$$n_2, \{0\}_{n_2}, \{0\}_{n_1} \longrightarrow n_1$$

What is the cost of each n_1 ?

Quantitative Analysis

Previous work (Cerversato, Adao et. al..) assigned costs to rules

We assign costs to terms, cost functions to rules

Quantitative Analysis

Previous work (Cerversato, Adao et. al..) assigned costs to rules

We assign costs to terms, cost functions to rules

$x, y \rightarrow \{x\}_y$ encryption

$$C(\{x\}_y) = C(x) + C(y) + 1$$

$x, y \rightarrow \text{dec}(x, y)$ decryption

$$C(\text{dec}(x, y)) = C(x) + C(y) + 1$$

Quantitative Analysis

Previous work (Cerversato, Adao et. al..) assigned costs to rules

We assign costs to terms, cost functions to rules

$x, y \rightarrow \{x\}_y$ encryption

$$C(\{x\}_y) = C(x) + C(y) + 1$$

$x, y \rightarrow \text{dec}(x, y)$ decryption

$$C(\text{dec}(x, y)) = C(x) + C(y) + 1$$

$n_1, f(n_1), f(g(n_2)) \rightarrow g(n_2)$

$$C(n_2) = 2^{14}C(f(g(n_2))) + \frac{-\ln(0.5)}{2^{-42}}(C(f(n_1)))$$

Thoughts...

- Cryptographic foundations?
- HSM as encryption oracle for parameterised analysis?
- How to search in such a formalism?
- Unify with previous treatment?