Attacks in the multi-user setting: Discrete logarithm, Even-Mansour and PRINCE

Pierre-Alain Fouque\textsuperscript{1}, Antoine Joux\textsuperscript{2}, Chrysanthi Mavromati\textsuperscript{3}

\textsuperscript{1} Université Rennes 1 and Institut Universitaire de France  
\textsuperscript{2} CryptoExperts and Chaire de Cryptologie de la Fondation de l’UPMC  
\textsuperscript{3} Sogeti/ESEC R&D Lab and UVSQ Laboratoire PRiSM

12 June 2014
The multi-user setting

Cryptographers prove the security of their schemes in a single-user model.

**In real world:** There are many users, each with a different key, sending each other encrypted data.
Main ideas

- Graph of key relations

- New variant of memory-less collision attacks
Generic discrete logarithm

- Single-user discrete log: time $\sqrt{N}$ (generic group)

- Multi-user discrete log ($L$ logs):
  - studied by Kuhn and Struik
  - use of the parallel version of the Pollard rho technique with distinguished points
  - time $\sqrt{NL}$, $L \leq N^{1/4}$
Distinguished points for discrete logarithms

- Define a random function $f : \mathcal{G} \rightarrow \mathcal{G}$
  
  $$f(z) = \begin{cases} 
  z^2 & \text{if } z \in \mathcal{G}_1, \\
  gz & \text{if } z \in \mathcal{G}_2,
  \end{cases}$$

  where $\mathcal{G}_1 \cup \mathcal{G}_2 = \mathcal{G}$.

- Define a distinguished subset $S_0$

- Build chains from random startpoints: $z_{i+1} = f(z_i)$

- Stop chain when $z_\ell = d \in S_0$

\[ g^{x_1} = y_1 \xrightarrow{f} y_2 \xrightarrow{f} y_3 \xrightarrow{f} y_4 \xrightarrow{f} \quad \log g d = A x_1 + B \]

\[ g^{x_1'} = y_1' \xrightarrow{f} y_2' \xrightarrow{f} y_3' \xrightarrow{f} y_4' \xrightarrow{f} \quad \log g d' = A' x_1' + B' \]
New method

\[ g^{x(0)} = y_0^{(0)} \xrightarrow{f} \ldots \xrightarrow{f} \ldots \]

\[ g^{x(1)} = y_0^{(1)} \xrightarrow{f} \ldots \xrightarrow{f} \ldots \]

\[ g^{x(L)} = y_0^{(L)} \xrightarrow{f} \ldots \xrightarrow{f} \ldots \]

linear relation between \(x^{(i)}\) and \(x^{(j)}\)
New method - Construct the graph

\[ \mathcal{L}_{x(i), x(j)} \]
New method - Construct the graph

\[ y_i, y_j \rightarrow \text{learn all keys in connected component} \]
Description of Even-Mansour

Introduced by Even and Mansour at [Asiacrypt '91].

- motivated by the DESX construction [Rivest, 1984]

![Diagram of Even-Mansour]

DES key $k$, whitening keys $k_1, k_2$
Description of Even-Mansour

Introduced by Even and Mansour at [Asiacrypt ’91].

- motivated by the DESX construction [Rivest, 1984]

\[ m \rightarrow DES \rightarrow c \]

DES key \( k \), whitening keys \( k_1, k_2 \)

- minimal construction of a blockcipher

\[
\Pi_{K_1,K_2}(m) = \pi(m \oplus K_1) \oplus K_2
\]

\[ m \rightarrow \pi \rightarrow \Pi(m) \]

- keyed permutation family \( \Pi_{K_1,K_2} \)
- \( \pi \) is a public permutation on \( n \)-bit values \((N = 2^n)\)
- two whitening keys \( K_1, K_2 \) of \( n \)-bits
Known results in the single-user model

**Main result:** Any attack with $D$ queries to $\Pi$ and $T$ off-line computation (queries to the public permutation $\pi$) has an upper bound of $O(DT/2^n)$ on probability of success.

**Single-Key EM:** Proved secure with the same bound [Dunkelman et al.]

P.-A. Fouque, A. Joux and Ch. Mavromati

Attacks on the Even-Mansour scheme
Slidex attack - Single key case

[Dunkelman et al., 2012]

Assume that two plaintexts \((P, P')\) satisfy \(P \oplus P' = K\) (slid pair).
Slidex attack - Single key case

[Dunkelman et al., 2012]

Assume that two plaintexts \((P, P')\) satisfy \(P \oplus P' = K\) (slid pair).

Apply the Davies-Meyer construction to \(\Pi\) and \(\pi\):

\[
F(P) = \Pi(P) \oplus P \quad \text{and} \quad f(P) = \pi(P) \oplus P
\]

\[
F(P') = \Pi(P') \oplus P' = \Pi(P \oplus K) \oplus P \oplus K
\]

\[
= \pi(P \oplus K \oplus K) \oplus K \oplus P \oplus K
\]

\[
= \pi(P) \oplus P = f(P)
\]

\[\Rightarrow F(P') = f(P)\]
Slidex attack - Single key case

[Dunkelman et al., 2012]

Assume that two plaintexts \((P, P')\) satisfy \(P \oplus P' = K\) (slid pair).

Apply the Davies-Meyer construction to \(\Pi\) and \(\pi\):

\[
F(P) = \Pi(P) \oplus P \quad \text{and} \quad f(P) = \pi(P) \oplus P
\]

\[
F(P') = \Pi(P') \oplus P' = \Pi(P \oplus K) \oplus P \oplus K
\]

\[
= \pi(P \oplus K \oplus K) \oplus K \oplus P \oplus K
\]

\[
= \pi(P) \oplus P = f(P)
\]

\[
\Rightarrow F(P') = f(P)
\]

Find a collision,

\[
\pi(P) \oplus P = \Pi(P') \oplus P'
\]

Then, \(P \oplus P'\) is a good candidate for \(K\).
Fix $\delta \in \{0, 1\}^n$

Assume that two plaintexts $(P, P')$ satisfy:
\[
P \oplus P' = K_1 \text{ or } P \oplus P' = K_1 \oplus \delta.
\]

Then, $F(P) = \Pi(P) \oplus \Pi(P \oplus \delta)$ and $f(P) = \pi(P) \oplus \pi(P \oplus \delta)$

\[
\Rightarrow F(P') = f(P) \quad \text{and} \quad F(P' \oplus \delta) = f(P)
\]

Find a collision,
\[
\Pi(P') \oplus \Pi(P' \oplus \delta) = \pi(P) \oplus \pi(P \oplus \delta)
\]

Then, $P \oplus P'$ and $P \oplus P' \oplus \delta$ are good candidates for $K_1$. 
The distinguished points method

- Define a function \( f \) on a set \( S \) of size \( N \).
- Define a distinguished subset \( S_0 \) of \( S \).
- Build chains from random startpoints: \( x_{i+1} = f(x_i) \).
- Stop chain when \( x_\ell = d \in S_0 \).
- Store \( (x_0, d, \ell) \).

How do we construct a collision?

How do we recover a chain?
The distinguished points method

- Define a function \( f \) on a set \( S \) of size \( N \).
- Define a distinguished subset \( S_0 \) of \( S \).
- Build chains from random startpoints: \( x_{i+1} = f(x_i) \).
- Stop chain when \( x_\ell = d \in S_0 \).
- Store \((x_0, d, \ell)\).

How do we construct a collision?

How do we recover a chain?
Goal: Find a collision between a set of chains using the public permutation \( \pi \) and a chain obtained from the keyed permutation \( \Pi \)

Define \( F(P) = \Pi(P) \oplus \Pi(P \oplus \delta) \) and \( f(P) = \pi(P) \oplus \pi(P \oplus \delta) \)

\[ \rightarrow \text{These chains can cross but not merge} \]
**Application on Even-Mansour - New idea**

**Define new functions:**

\[ F(P) = P \oplus \Pi(P) \oplus \Pi(P \oplus \delta) \] and
\[ f(P) = P \oplus \pi(P) \oplus \pi(P \oplus \delta) \]

- Assume that two plaintexts \((P, P')\) satisfy:
  \[ P' = P \oplus K_1 \text{ or } P' = P \oplus K_1 \oplus \delta \]
- Then \( F(P') = f(P) \oplus K_1 \) (resp. \( \oplus \delta \))

→ These chains can become parallel
Application on Even-Mansour - New idea

Define new functions:

\[ F(P) = P \oplus \Pi(P) \oplus \Pi(P \oplus \delta) \quad \text{and} \]
\[ f(P) = P \oplus \pi(P) \oplus \pi(P \oplus \delta) \]

- Assume that two plaintexts \((P, P')\) satisfy:
  \[ P' = P \oplus K_1 \quad \text{or} \quad P' = P \oplus K_1 \oplus \delta \]
- Then \(F(P') = f(P) \oplus K_1\) (resp. \(\oplus \delta\))

→ These chains can become parallel
Detection of parallel chains with distinguished points

- For $f$ chains: define a distinguished point $P$ as a point with a value of $\pi(P) \oplus \pi(P \oplus \delta) \in S_0$

- For $F$ chains: define a distinguished point $P'$ as a point with a value of $\Pi(P') \oplus \Pi(P' \oplus \delta) \in S_0$

- If $P' = P \oplus K_1$ and $P$ is a distinguish point in the $f$ chain, then:

$$
\Pi(P') \oplus \Pi(P' \oplus \delta) = \pi(P' \oplus K_1) \oplus K_2 \oplus \pi(P' \oplus K_1 \oplus \delta) \oplus K_2
= \pi(P) \oplus \pi(P \oplus \delta)
$$

and then $P'$ is a distinguished point in the $F$ chain

- $\rightarrow P \oplus P' = K_1$
New attack on Even-Mansour

- Build chains from $f(P) = P \oplus \pi(P) \oplus \pi(P \oplus \delta)$
  - Stop if $\pi(P) \oplus \pi(P \oplus \delta)$ arrives at a distinguished point
- Build chains from $F(P) = P \oplus \Pi(P) \oplus \Pi(P \oplus \delta)$
  - Stop if $\Pi(P) \oplus \Pi(P \oplus \delta)$ arrives at a distinguished point
- These chains cannot merge but can become parallel
  - Assume $P' = P \oplus K_1$ or $P' = P \oplus K_1 \oplus \delta$
  - $\implies F(P') = f(P) \oplus K_1 \oplus \delta$ respectively
- We only need to store endpoints (don’t have to recompute chains)
Attack Even-Mansour in the multi-user setting

- Build chains from $f$ of length $2^{n/3}$
- Build chains from $F$ of length $2^{n/3}$ for each user
- Construct a graph:
  - Nodes are labelled by the users and the unkeyed user
  - If $F^{(i)} = F^{(j)}$ (for users $(i), (j)$), then add a vertex between the two nodes
  - $\rightarrow K^{(i)}_1 \oplus K^{(j)}_1 (\oplus \delta)$
- If we find a single collision between a user and the unkeyed user, then we learn all keys (in the connected component)

Analysis of the attack:

For $2^{n/3}$ users, $2^{n/3}$ queries/user, $2^{n/3}$ unkeyed queries
$\rightarrow$ recover almost all $2^{n/3}$ keys
Description of PRINCE

PRINCE [Borghoff et al., Asiacrypt 2012]

- 64-bit lightweight block cipher
- 128-bit key $k$ split into equal parts: $k = k_0 \| k_1$
- extension to 192 bit: $k = (k_0 \| k_1) \rightarrow (k_0 \| k'_0 \| k_1)$
- $k'_0$ derived from $k_0$ by using the linear function $L'$:
  $L'(k_0) = (k_0 \gg 1) \oplus (k_0 \gg 63)$
- $\alpha$-reflection property

\[
\forall (k_0 \| k'_0 \| k_1), \quad D_{(k_0 \| k'_0 \| k_1)}(\cdot) = E_{(k'_0 \| k_0 \| k_1 \oplus \alpha)}(\cdot)
\]

\[
E_k(m) = k'_0 \oplus P_{\text{core}}(m \oplus k_0)
\]
Attacks on PRINCE in the single and multi-user setting

Attack in the multi-user setting

**Total cost** $2^{65}$ operations for deducing $k_0$ and $k_1$ of 2 users in a set of $2^{32}$.

Attack in the single-user setting

$$T_{\text{off}} = 2^{96}, T_{\text{on}} = 2^{32}, M = 2^{32}$$

$$D T_{\text{off}} = 2^{128}$$
Conclusion

• Propose two new algorithmic ideas to improve collision based attacks

• Application of the first idea to solve the discrete logarithm problem in the multi-user setting

• Application of both ideas to the Even-Mansour scheme

• Propose two new attacks for PRINCE
  • The attacks have been applied to DESX with some differences
Conclusion

- Propose two new algorithmic ideas to improve collision based attacks
- Application of the first idea to solve the discrete logarithm problem in the multi-user setting
- Application of both ideas to the Even-Mansour scheme
- Propose two new attacks for PRINCE
  - The attacks have been applied to DESX with some differences

Thank you for your attention!