Reverse Cycle Walking and its Applications

Sarah Miracle and Scott Yilek

University of St. Thomas

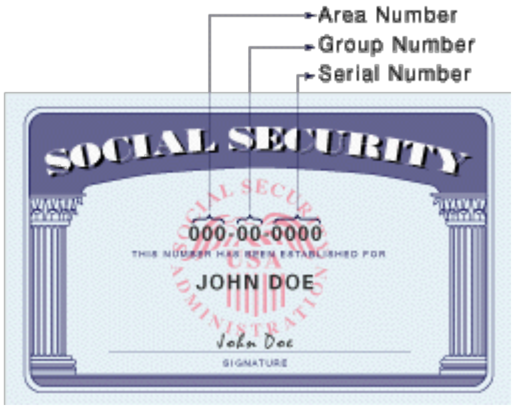
UNIVERSITY OF St.Thomas

Format Preserving Encryption

Example:

Existing database with millions of US social security numbers

- 9 digit numbers
- First 3 digits can’t be 666
- And more . . .

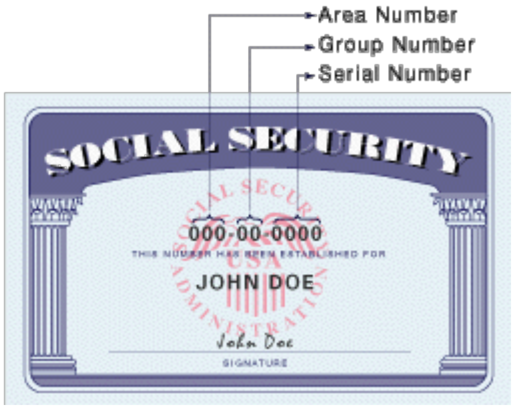


Format Preserving Encryption

<u>Example:</u>

Existing database with millions of US social security numbers

- 9 digit numbers
- First 3 digits can't be 666
- And more . . .



How to add encryption?

Format Preserving Encryption

Example:

Existing database with millions of US social security numbers

How to add encryption?

Format Preserving Encryption

<u>Example:</u>

Existing database with millions of US social security numbers

How to add encryption?

- Represent SSN as 30-bit numbers
- Pad with zeros
- Encrypt using a standard block cipher

  (e.g. AES)

Format Preserving Encryption

<u>Example:</u>

Existing database with millions of US social security numbers

How to add encryption?

- Represent SSN as 30-bit numbers
- Pad with zeros
- Encrypt using a standard block cipher (e.g. AES)

Encrypted numbers have a significantly different format!

Format Preserving Encryption

<u>Format Preserve Encryption schemes:</u>

Encryption schemes in which ciphertexts have the same format as plaintexts.

Name	SSN
Alan Johnson	845-10-3814
Brianna Smith	143-66-9875
Catherine Jones	333-79-0123
Douglas Wilson	593-58-2057

Encrypted SSN
942-54-3210
771-98-1094
574-30-9214
889-01-4884

Talk Outline

- Background and Previous Work
- Our Algorithm
- Proof Outline

Background

Background

- Small-domain block ciphers for bitstrings or integers up to N

Background

- Small-domain block ciphers for bitstrings or integers up to N

  [Hoang, Morris, Rogaway ’12], [Ristenpart, Yilek ‘13], [Morris,Rogaway ’14] . . . .

Background

- Small-domain block ciphers for bitstrings or integers up to N

  [Hoang, Morris, Rogaway ’12], [Ristenpart, Yilek ‘13],

  [Morris,Rogaway ’14] . . . .
- If the target set S has an efficient way to rank/unrank then you can use a cipher on $\{0,\ldots, |S| - 1\}$

Background

- Small-domain block ciphers for bitstrings or integers up to N

    [Hoang, Morris, Rogaway ’12], [Ristenpart, Yilek ‘13],

    [Morris,Rogaway ’14] . . . .
- If the target set S has an efficient way to rank/unrank then you can use a cipher on $\{0,\ldots, |S| - 1\}$

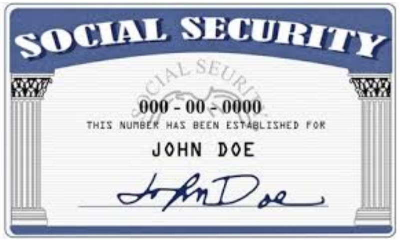
We’ll only assume we can test membership in our target domain set S

General Approach

1. Find a cipher on a larger set T
2. Transform it to a cipher on a smaller set S

<u>Example: Social Security Numbers</u>

- Let T be the set of 30-bit strings ($10^9 < 2^{30}$)
- There are many block ciphers to encipher 30-bit strings



Cycle Walking

1. Find a cipher on a larger set T
2. Transform it to a cipher on a smaller set S

Cycle Walking

1. Find a cipher on a larger set T
2. Transform it to a cipher on a smaller set S

```
Algorithm CW_π(x):
do
      x ← π(x)
while(x ∉ S)
return x
```

Cycle Walking - Example

$T = \{0,\ldots,9\}$

$S = \{0,2,4,6,8\}$

```
Algorithm CW_π(x):
do
      x←π(x)
while(x∉S)
return x
```

Consider the cycle structure:

$(9\ 4\ 6\ 5\ 1\ 0)\ (3\ 2\ 7\ 8)$ ← Permutation on T

↓

$(\cancel{9}\ 4\ 6\ \cancel{5}\ \cancel{1}\ 0)\ (\cancel{3}\ 2\ \cancel{7}\ 8)$

↓

$(4\ 6\ 0)\ (2\ 8)$ ← Permutation on S

Running Time of Cycle Walking

Formally analyzed by Black and Rogaway [CT-RSA 2002]

Running Time of Cycle Walking

Formally analyzed by Black and Rogaway [CT-RSA 2002]

- Small expected running time – O(1)
  (assuming $|S|$ is a constant fraction of $|T|$)

Running Time of Cycle Walking

Formally analyzed by Black and Rogaway [CT-RSA 2002]

- Small expected running time – O(1)
  (assuming $|S|$ is a constant fraction of $|T|$)
- Worst case running time of $\Theta(|T|)$

Running Time of Cycle Walking

Formally analyzed by Black and Rogaway [CT-RSA 2002]

- Small expected running time – O(1)
  (assuming $|S|$ is a constant fraction of $|T|$)
- Worst case running time of $\Theta(|T|)$
- Different run times can leak timing information
  - If the adversary has access to ciphertexts, # cycle-walking steps then not damaging
    [Bellare, Ristenpart, Rogaway, Stegers '09]
  - In general?

Running Time of Cycle Walking

Formally analyzed by Black and Rogaway [CT-RSA 2002]

- Small expected running time – O(1)
  (assuming $|S|$ is a constant fraction of $|T|$)
- → Worst case running time of $\Theta(|T|)$
- → Different run times can leak timing information
  - If the adversary has access to ciphertexts, # cycle-walking steps then not damaging
    [Bellare, Ristenpart, Rogaway, Stegers '09]
  - In general?

Can we do better?

Talk Outline

- Background and Previous Work
- Our Algorithm – an alternative to cycle walking
- Proof Outline

First Approach

$T = \{0,\ldots,9\}$

$S = \{0,2,4,6,8\}$

Consider the cycle structure:

$(4\ 1\ 3\ 5\ 7\ 0\ 2\ 9\ 6\ 8)$ ← Permutation on T

First Approach

$T = \{0,\ldots,9\}$

$S = \{0,2,4,6,8\}$

Consider the cycle structure:

$(4\ 1\ 3\ 5\ 7\ 0\ 2\ 9\ 6\ 8)$ ← Permutation on T

Idea: Cut-off Cycle Walking Early

Reverse Cycle Walking

T = {0,…,9}

S = {0,2,4,6,8}

Consider the cycle structure:

(4 1 3 5 7 0 2 9 6 8)

↓

(4 ~~1 3 5 7~~ 0 2 ~~9~~ 6 8)

↓

(6 8 4) (0 2)

Our Algorithm: Walk backward

Reverse Cycle Walking

T = {0,…,9}

S = {0,2,4,6,8}

Consider the cycle structure:

(4 1 3 5 7 0 2 9 6 8)

(4 ~~1 3 5 7~~ 0 2 ~~9~~ 6 8)

(6 8 4) (0 2) → (4) (8) (6) (0 2)

Our Algorithm: Walk backward + only consider 2-cycles

Another Example

$T = \{0,\ldots,9\}$

$S = \{0,2,4,6,8\}$

Consider the cycle structure:

$(9\ 4\ 6\ 5\ 1\ 0)\ (3\ 2\ 7\ 8)$ ← Permutation on T

↓

(~~9~~ 4 6 ~~5 1~~ 0) (~~3~~ 2 ~~7~~ 8)

↓

$(4\ 6)\ (0)\ (2)\ (8)$ ← Permutation on S

Reverse Cycle Walking

```
Algorithm RCW_{π,B}(x):
________________________________
y ← π(x); z ← π^{-1}(x)
if y ∈ S and z ∉ S and π(y) ∉ S:
    b ← B(x)
    if b = 1 return y else return x
else if y ∉ S and z ∈ S and π^{-1}(z) ∉ S:
    b ← B(z)
    if b = 1 return z else return x
else
    return x
```

Running Time of RCW

Running Time of RCW

- 1 Step of RCW takes $O(1)$ time

Running Time of RCW

- 1 Step of RCW takes $O(1)$ time
- But, even if π is random, RCW_{π} is NOT random

Running Time of RCW

- 1 Step of RCW takes $O(1)$ time
- But, even if π is random, RCW_π is NOT random

How many rounds of RCW are needed before the resulting permutation on S is close to random?

Running Time of RCW

- 1 Step of RCW takes $O(1)$ time
- But, even if π is random, RCW_π is NOT random

How many rounds of RCW are needed before the resulting permutation on S is close to random?

This is a Markov chain!

Running Time of RCW

- 1 Step of RCW takes $O(1)$ time
- But, even if π is random, RCW_{π} is NOT random

How many rounds of RCW are needed before the resulting permutation on S is close to random?

This is a Markov chain!

Answer: $O(\log |T|)$

Advantages of RCW

Advantages of RCW

- Lower worst case running time - **$O(n)$ to $O(\log n)$**

Advantages of RCW

- Lower worst case running time - **$O(n)$ to $O(\log n)$**
- No leaked timing information

Advantages of RCW

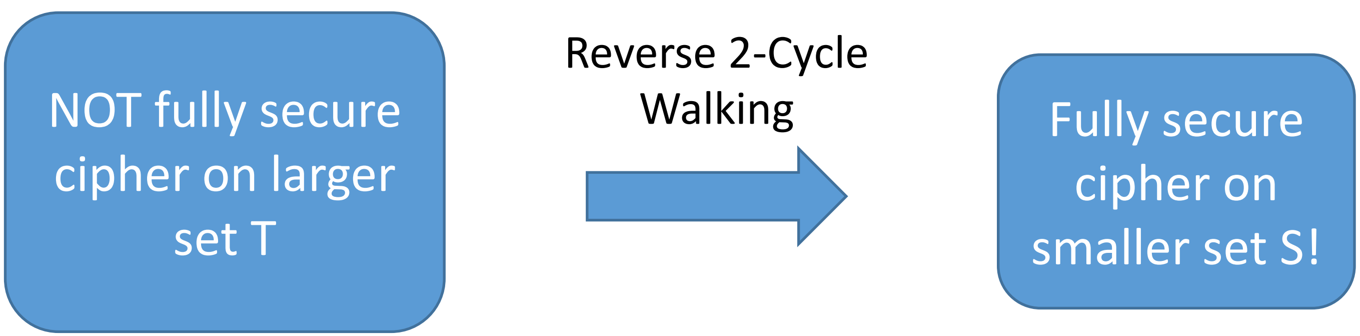
- Lower worst case running time - $\mathbf{O(n)}$ **to** $\mathbf{O(\log n)}$
- No leaked timing information
- Can trade-off security and running time

Advantages of RCW

- Lower worst case running time - **$O(n)$ to $O(\log n)$**
- No leaked timing information
- Can trade-off security and running time
- Increases the level of security

Advantages of RCW

- Lower worst case running time - **$O(n)$ to $O(\log n)$**
- No leaked timing information
- Can trade-off security and running time
- Increases the level of security



under certain circumstances . . .

Talk Outline

- Background and Previous Work
- Reverse Cycle Walking
- Proof Outline – analyzing the mixing time of RCW

Mixing Time

Definition: The total variation distance is

$$|| P^t, \pi || = \max_{x \in \Omega} \tfrac{1}{2} \sum_{y \in \Omega} |P^t(x,y) - \pi(y)|.$$

Definition: Given ε, the mixing time is

$$\tau(\varepsilon) = \min \{t : ||P^{t'}, \pi|| < \boldsymbol{\varepsilon}, \quad \forall t' \geq t\}.$$

Bounding the Mixing Time of RCW

Bounding the Mixing Time of RCW

1. Show that RCW yields a “matching exchange process”

Bounding the Mixing Time of RCW

1. Show that RCW yields a “matching exchange process”
2. Defined and analyzed by Czumaj and Kutylowski [RSA ’00]

Bounding the Mixing Time of RCW

1. Show that RCW yields a “matching exchange process”
2. Defined and analyzed by Czumaj and Kutylowski [RSA ’00]
3. Use same techniques but . . .
    - Give explicit constants for RCW algorithm
    - Reprove several key lemmas

Matching Exchange Process

<u>Matching Exchange:</u>

Repeat:

1. Choose a number κ according to some distribution.
2. Pick a matching M of size κ uniformly at random
3. For each pair in the matching,
   - transpose the two points with prob. ½
   - otherwise, do nothing

Matching Exchange Process

Matching Exchange:

Repeat:

1. Choose a number κ according to some distribution.
2. Pick a matching M of size κ uniformly at random
3. For each pair in the matching,
   - transpose the two points with prob. ½
   - otherwise, do nothing

Theorem [Czumaj, Kutylowski]: If $\mathrm{E}(\kappa)$ is $\Theta(\mathrm{n})$ then a matching exchange process mixes in time $\mathrm{O}(\log \mathrm{n})$.

Path Coupling Approach

[Bubley,Dyer,Greenhill’ 97-8]

Path Coupling Approach

[Bubley,Dyer,Greenhill' 97-8]

- Consider 2 configurations that differ by a single transposition (u,v)

Path Coupling Approach

[Bubley,Dyer,Greenhill’ 97-8]

- Consider 2 configurations that differ by a single transposition (u,v)
- If the first matching contains the pair (u,v) then we can couple the processes after a single step

Path Coupling Approach

[Bubley,Dyer,Greenhill' 97-8]

- Consider 2 configurations that differ by a single transposition (u,v)
- If the first matching contains the pair (u,v) then we can couple the processes after a single step
- But, this only happens with probably $O(1/n)$

Path Coupling Approach

[Bubley,Dyer,Greenhill' 97-8]

- Consider 2 configurations that differ by a single transposition (u,v)
- If the first matching contains the pair (u,v) then we can couple the processes after a single step
- But, this only happens with probably $O(1/n)$

$$\Longrightarrow \quad O(n \log n)$$

Analyzing a Matching Exchange

Analyzing a Matching Exchange

<u>High-level Approach:</u>

Analyzing a Matching Exchange

High-level Approach:

- Look at what happens over O(log(n)) steps.

  Delayed Path Coupling [Czumaj, et al.]

Analyzing a Matching Exchange

High-level Approach:

- Look at what happens over O(log(n)) steps.

  Delayed Path Coupling [Czumaj, et al.]

- Use a non-Markovian coupling

Analyzing a Matching Exchange

High-level Approach:

- Look at what happens over O(log(n)) steps.
  Delayed Path Coupling [Czumaj, et al.]
- Use a non-Markovian coupling
- Let $M_1 \ldots M_t$ be the matchings for process X and $N_1 \ldots N_t$ be the matchings for process Y.

Analyzing a Matching Exchange

High-level Approach:

- Look at what happens over O(log(n)) steps.
  Delayed Path Coupling [Czumaj, et al.]
- Use a non-Markovian coupling
- Let $\mathrm{M}_1 \ldots \mathrm{M}_\mathrm{t}$ be the matchings for process X and $\mathrm{N}_1 \ldots \mathrm{N}_\mathrm{t}$ be the matchings for process Y.
- Choose $\mathrm{M}_1 \ldots \mathrm{M}_\mathrm{t}$ randomly – according to the alg.

Key Idea

Key Idea

- Assume X_0 and Y_0 differ by a (u,v) transposition

Key Idea

- Assume X_0 and Y_0 differ by a (u,v) transposition
- Assume M_1 contains (u,z) and (v,w)
  - If you let $N_1 = M_1$ then X_1 and Y_1 differ by a (z,w) trans.
  - If you let $N_1 = M_1 - (u,z) - (v,w) + (u,w) + (v,z)$ then X_1 and Y_1 differ by a (u,v) trans.

Key Idea

- Assume X_0 and Y_0 differ by a (u,v) transposition
- Assume M_1 contains (u,z) and (v,w)
  - If you let $N_1 = M_1$ then X_1 and Y_1 differ by a (z,w) trans.
  - If you let $N_1 = M_1 - (u,z) - (v,w) + (u,w) + (v,z)$ then X_1 and Y_1 differ by a (u,v) trans.
- If M_2 contains (u,v) OR (z,w) then can choose N_2 so that $\Delta(X_1, Y_1) = 0$.

Key Idea

- Assume X_0 and Y_0 differ by a (u,v) transposition
- Assume M_1 contains (u,z) and (v,w)
  - If you let $N_1 = M_1$ then X_1 and Y_1 differ by a (z,w) trans.
  - If you let $N_1 = M_1 - (u,z) - (v,w) + (u,w) + (v,z)$ then X_1 and Y_1 differ by a (u,v) trans.
- If M_2 contains (u,v) OR (z,w) then can choose N_2 so that $\Delta(X_1, Y_1) = 0$.

Call (u,v) and (z,w) "good pairs".

Key Lemmas

Key Lemmas

1. Show that after $\Theta(\log n)$ steps with high probability, the number of good pairs is $\Theta(n)$

Key Lemmas

1. Show that after $\Theta(\log n)$ steps with high probability, the number of good pairs is $\Theta(n)$
2. Show that with high probability, one of the next $\Theta(\log n)$ matchings contains a good pair

Future Directions

Future Directions

- Improve the constants further

Future Directions

- Improve the constants further
- Remove the bit flip

Future Directions

- Improve the constants further
- Remove the bit flip
- Design an alternative algorithm

  Expected O(1) running time of cycle walking is very attractive

Questions?