LibFTE: A User-Friendly Toolkit for Constructing Practical Format-Abiding Encryption Schemes

Daniel Luchaup, University of Wisconsin-Madison
Kevin Dyer, Portland State University
Somesh Jha, University of Wisconsin-Madison
Thomas Ristenpart, University of Wisconsin-Madison
Thomas Shrimpton, Portland State University
Encryption
Drawback: traditional encryption does not preserve any format
Format Preserving Encryption

(Bellare et. al., 2009)

formatted data

encrypted & formatted data

FPE: plain text and cipher text have similar format
Format Preserving Encryption

(Bellare et. al., 2009)

formatted data

John Doe 1234567890
Jane Doe 2345678909

encrypted & formatted data

Abcd Efg 7865409889
Hlmn Opl 8099087217

Mike Kay 900-88-7777
Paul Kim 800-77-5555

FPE: plain text and cipher text have similar format
Format Transforming Encryption

(Dyer et. al., 2013)

formatted data

John Doe 1234567890
Jane Doe 2345678909

John Doe 987-65-4321
Jane Doe 876-54-3210

encrypted & formatted data

Abcd Efg 7865409889
Hlmn Opl 8099087217

Can we change the format?

Mike Kay 900-88-7777
Paul Kim 800-77-5555
Format Transforming Encryption

(Dyer et. al., 2013)

formatted data

John Doe 1234567890
Jane Doe 2345678909

John Doe 987-65-4321
Jane Doe 876-54-3210

encrypted & formatted data

Mike Kay 900-88-7777
Paul Kim 800-77-5555

Abcd Efg 7865409889
Hlmn Opl 8099087217

FTE: plain text and cipher text have different formats
FTE: Censorship Circumvention

*(Dyer et. al., 2013)*

Censor filters out TOR traffic but allows HTTP
FTE: Censorship Circumvention

(Dyer et. al., 2013)

Censor Circumvented: TOR traffic passed through encrypted as HTTP
Format

• A format is simply a language $L$
  – regular languages $L$ defined by regular expressions
    • $[a-zA-Z]^+$
    • $[a-zA-Z]+\backslash [a-zA-Z]^+\backslash [0-9]\{9\}$
  – Finite limits specified by the problem
    • $w \in L([a-zA-Z]^*)$, $|w| < 20$
Rank And Encipher: FPE

- Ranking Scheme:
  - rank: $L \rightarrow \mathbb{Z}_{|L|}$
  - unrank: $\mathbb{Z}_{|L|} \rightarrow L$

- Integer encryption:
  - encrypt $E: \mathbb{Z}_{|L|} \rightarrow \mathbb{Z}_{|L|}$
  - decrypt $D: \mathbb{Z}_{|L|} \rightarrow \mathbb{Z}_{|L|}$

- FPE
  - encryption of $x$:
    $y = \text{unrank}(\text{encrypt}(\text{rank}(x)))$
  - decryption of $y$:
    $x = \text{rank}(\text{decrypt}(\text{unrank}(y)))$
Rank And Encipher: FPE

- Ranking Scheme:
  - rank: \( L \rightarrow \mathbb{Z}_{|L|} \)
  - unrank: \( \mathbb{Z}_{|L|} \rightarrow L \)

- Integer encryption:
  - encrypt \( E: \mathbb{Z}_{|L|} \rightarrow \mathbb{Z}_{|L|} \)
  - decrypt \( D: \mathbb{Z}_{|L|} \rightarrow \mathbb{Z}_{|L|} \)

- FPE
  - encryption of \( x \):
    \[ y = \text{unrank}(\text{encrypt}(\text{rank}(x))) \]
  - decryption of \( y \):
    \[ x = \text{rank}(\text{decrypt}(\text{unrank}(y))) \]
Rank And Encipher: FTE

What about $|L|$ versus $|L'|$?
- $|L| > |L'|$ cannot encrypt
- $|L| \leq |L'|$ ok
Formatted Encryption

- Specification of language/formats L, L’
  - finite vs. infinite
- Aware of sizes |L|, |L’|
- Efficient rank/unrank
- Efficient integer encryption
Limitations of prior work

• Limited Ranking/Unranking
  – DFA based: only work with simple regex
  – NFA ranking thought impossible
• No public implementation
• Awkward format specification
  – theoretically specified for fixed slices of regular lang.
• No performance analysis
• No configuration
  – Input/Output language selection
  – Reasoning about language sizes.
• Need: generic framework, simple specification, yet fast
New Work: LibFTE *(Luchaup et. al., 2013)*

- Public implementation
- Generic framework, simple specification
  - regular expression, size ranges
- Fast
  - Improved DFA ranking
  - NFA ranking
  - Choice of DFA/NFA ranking transparent to user
- Configuration
  - Input/Output language selection
  - Tool to help user reasoning about configuration choices
- Performance analysis
- Applications:
  - In browser encryption
  - DB encryption and compression
LibFTE: NFA-based ranking

DFA:
• $D = (Q, \Sigma, \delta, q_0, F)$
  – $\delta$ is deterministic
  – count accepting paths
  – unique accepting paths
  large

NFA:
• $N = (Q, \Sigma, \delta, q_0, F)$
  – $\delta$ is not deterministic
  – count accepting paths
  – multiple accepting paths possible
  small
LibFTE: NFA-based ranking

DFA:
- $D = (Q, \Sigma, \delta, q_0, F)$
  - $\delta$ is deterministic
  - count accepting paths
    rank/unrank bijection
    - $\text{unrank} (\text{rank}(x)) = x$
    - $\text{rank} (\text{unrank}(n)) = n$

NFA:
- $D = (Q, \Sigma, \delta, q_0, F)$
  - $\delta$ is not deterministic
  - count accepting paths
    Rank injective/Unrank
    - $\text{Rank}(\text{Unrank}(n)) = n$ may not hold

$L_0 \rightarrow 1 \rightarrow 2 \rightarrow |L|-1$

$p \geq |L| - 1$
Cycle Walking

NFA:
- \( D = (Q, \Sigma, \delta, q_0, F) \)
  - \( \delta \) is not deterministic
  - count accepting paths
    - Rank injective/ Unrank
    - \( \text{Rank}(\text{Unrank}(x)) = x \)
    - \( \text{Rank}(%Unrank(n)) \) may not hold

r = rank(x)
e1 = encrypt(r)
\[ \text{Rank}(\text{Unrank}(e1)) \neq e1 \]
e2 = encrypt(e1)
\[ \text{Rank}(\text{Unrank}(e2)) = e2 \]
Rank And Encipher with Injective Ranking

Adaptations:

• Cycle walk
• Nondeterministic encryption
• Language sizes are relevant

• Details in the paper *(Luchaup et. al., 2013)*
New Work: LibFTE

- Public implementation
- Generic framework, simple specification
  - regular expression, size ranges
- Fast
  - Improved DFA ranking
  - Relaxed ranking
  - NFA ranking
  - Choice of DFA/NFA ranking transparent to user
- Configuration
  - Input/Output language selection
  - Tool to help user reasoning about configuration choices
- Performance analysis
- Applications:
  - In browser encryption
  - DB encryption and compression
QUESTIONS?

luchaup@cs.wisc.edu