Symmetric Encryption via Keyrings and ECC

Ronald L. Rivest

Institute Professor
MIT, Cambridge, MA

ArcticCrypt
2016-07-18

Outline

Motivation—Simplifying Crypto Key Updates

- Keyring (Bag of Words) Model
- Incremental Key Updates
- Keyring Issues

Resilience

- Prior Work—Biometrics, Fuzziness, Quantum
- Resilient Set Vectorization
- Security Analysis

Encrypting with keyrings

- Error-correction
- Keyring encryption details
- Attacks

Discussion

Motivation—Simplifying Key Updates

$\longleftrightarrow$

Updating symmetric crypto keys is hard, because they:

CSAIL

Motivation—Simplifying Key Updates

$\longleftrightarrow$

Updating symmetric crypto keys is hard, because they:

- have **high entropy**

Motivation—Simplifying Key Updates

 $\longleftrightarrow$ 

Updating symmetric crypto keys is hard, because they:

- have **high entropy**
- are **not memorable**, and

Motivation—Simplifying Key Updates

$\longleftrightarrow$

Updating symmetric crypto keys is hard, because they:

- have **high entropy**
- are **not memorable**, and
- are **updated "all-at-once"** instead of incrementally.

CSAIL

Motivation—Simplifying Key Updates

$\longleftrightarrow$

Updating symmetric crypto keys is hard, because they:

- have **high entropy**
- are **not memorable**, and
- are **updated "all-at-once"** instead of incrementally.

Are there better (non-PK) methods?

CSAIL

Keyring (Bag of Words) Model

Main idea: Key is a "bag of words" agreed upon by sender and receiver. (Really "set" not "bag" (multiset).)

Keyrings

- Each word is a **keyword**.

Keyrings

- Each word is a **keyword**.
- Bag is a **keyring**.

Keyrings

- Each word is a **keyword**.
- Bag is a **keyring**.
- Separate keyring for each sender/receiver pair.

Keyrings

- Each word is a **keyword**.
- Bag is a **keyring**.
- Separate keyring for each sender/receiver pair.
- Sender and receiver have identical (*or nearly identical*) keyrings.

CSAIL

Keyrings

- Each word is a **keyword**.
- Bag is a **keyring**.
- Separate keyring for each sender/receiver pair.
- Sender and receiver have identical (*or nearly identical*) keyrings.
- Maybe 10–100 keywords on a keyring.

CSAIL

Incremental Key Updates Are Simple

Alice says privately to Bob:

- Let’s add “**garlic**” to our keyring.

Incremental Key Updates Are Simple

Alice says privately to Bob:

- Let's add "**garlic**" to our keyring.
- Let's delete "**mustard**" from our keyring.

Incremental Key Updates Are Simple

Alice says privately to Bob:

- Let's add "**garlic**" to our keyring.
- Let's delete "**mustard**" from our keyring.
- Let's add all words from your last two tweets.

Incremental Key Updates Are Simple

Alice says privately to Bob:

- Let's add "**garlic**" to our keyring.
- Let's delete "**mustard**" from our keyring.
- Let's add all words from your last two tweets.
- Let's add words of a quote:

Incremental Key Updates Are Simple

Alice says privately to Bob:

- Let's add "**garlic**" to our keyring.
- Let's delete "**mustard**" from our keyring.
- Let's add all words from your last two tweets.
- Let's add words of a quote:

  "It is a miracle that curiosity survives formal education."

  (Albert Einstein)

Incremental Key Updates Are Simple

Alice says privately to Bob:

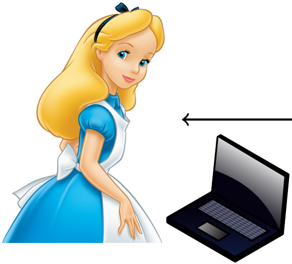
- Let's add "**garlic**" to our keyring.
- Let's delete "**mustard**" from our keyring.
- Let's add all words from your last two tweets.
- Let's add words of a quote:

  "It is a miracle that curiosity survives formal education."

  (Albert Einstein)

- Let's delete all keywords added in 2015.

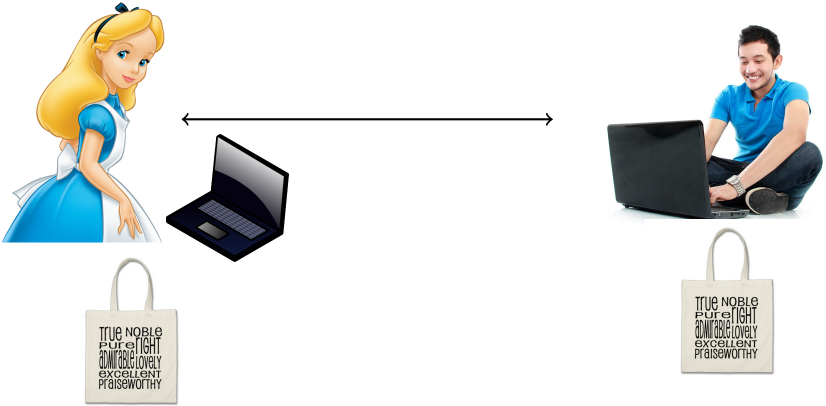
Scenario



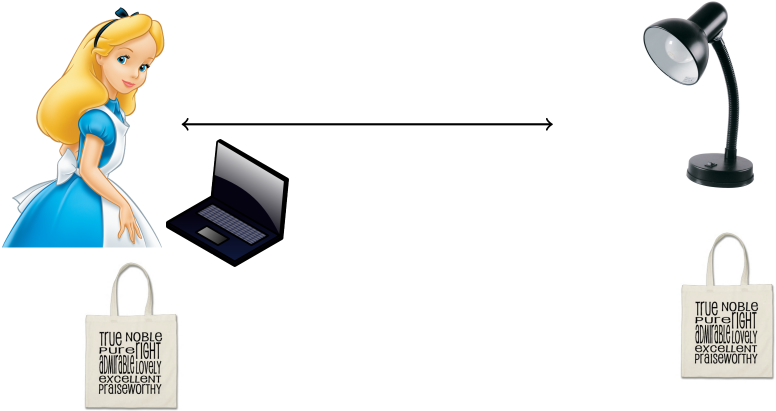
key $=$ `0x47a31...f3`

key $=$ `0x47a31...f3`

Scenario


True Noble
Pure Right
Admirable Lovely
Excellent
Praiseworthy
True Noble
Pure Right
Admirable Lovely
Excellent
Praiseworthy

Scenario



CSAIL

Keyring Issues

- **(Resilience)** How to make encryption work even if Alice and Bob's keyrings are slightly "out of sync"?

Keyring Issues

- **(Resilience)** How to make encryption work even if Alice and Bob's keyrings are slightly "out of sync"?
- **(Keying)** How to use a "bag of words" as a symmetric crypto key?

Keyring Issues

- **(Resilience)** How to make encryption work even if Alice and Bob's keyrings are slightly "out of sync"?
- **(Keying)** How to use a "bag of words" as a symmetric crypto key?
- **(Security)** How to keep adversary from breaking in and then "tracking" keyring evolution?

CSAIL

Resilience

We want that a ciphertext made using keyring A can be decrypted using different keyring B, as long as A and B are “close”.

Resilience

We want that a ciphertext made using keyring A can be decrypted using different keyring B, as long as A and B are "close".

Two metrics of interest:

Resilience

We want that a ciphertext made using keyring A can be decrypted using different keyring B, as long as A and B are "close".
Two metrics of interest:

- **Set distance**. (Relative) size of set difference. That is, $|A \oplus B|$ or $|A \oplus B|/|A \cup B|$.

Resilience

We want that a ciphertext made using keyring A can be decrypted using different keyring B, as long as A and B are "close".
Two metrics of interest:

- **Set distance**. (Relative) size of set difference. That is, $|A \oplus B|$ or $|A \oplus B|/|A \cup B|$.
- **Hamming distance**. (Relative) number of positions in which vectors x and y differ.

Resilience

We want that a ciphertext made using keyring A can be decrypted using different keyring B, as long as A and B are "close".

Two metrics of interest:

- **Set distance**. (Relative) size of set difference. That is, $|A \oplus B|$ or $|A \oplus B|/|A \cup B|$.
- **Hamming distance**. (Relative) number of positions in which vectors x and y differ.

We describe a nice way of converting from the first to the second.

Biometrics: Use a fingerprint as key

Our problem is not particularly new...

Biometrics: Use a fingerprint as key

Our problem is not particularly new...
Similar to the problem of encrypting a key with a biometric; biometric features $\sim$ keywords.

Fuzziness everywhere

- Juels/Wattenberg 1999 *"A Fuzzy Commitment Scheme"*. Introduces "code-offset" construction.

Fuzziness everywhere

- Juels/Wattenberg 1999 *"A Fuzzy Commitment Scheme"*. Introduces "code-offset" construction.
- Juels/Sudan 2006 *"A Fuzzy Vault Scheme"* Based on clever use of interpolation.

Fuzziness everywhere

- Juels/Wattenberg 1999 *"A Fuzzy Commitment Scheme"*. Introduces "code-offset" construction.
- Juels/Sudan 2006 *"A Fuzzy Vault Scheme"* Based on clever use of interpolation.
- Dodis/Ostrovsky/Reyzin/Smith 2004 *"Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data"* Relates 'secure sketches" and fuzzy extractors. (Also: Dodis/Reyzin/Smith 2007 *"Fuzzy Extractors"*)

Fuzziness everywhere

- Juels/Wattenberg 1999 *“A Fuzzy Commitment Scheme”*. Introduces “code-offset” construction.
- Juels/Sudan 2006 *“A Fuzzy Vault Scheme”* Based on clever use of interpolation.
- Dodis/Ostrovsky/Reyzin/Smith 2004 *“Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data”* Relates ‘secure sketches” and fuzzy extractors. (Also: Dodis/Reyzin/Smith 2007 *“Fuzzy Extractors”*)
- Sahai/Waters 2005 *“Fuzzy IBE”*. Fuzzy PK scheme.

PinSketch[DORS04]

- Uses BCH ECC with algorithms that work efficiently on *sparse* vectors.
- Message transmitted has length δ over $GF(2^\alpha)$, where $2^\alpha \geq |\mathcal{U}|$ and $\mathcal{U}$ is universe of keys, and where δ is upper bound on the size of the set difference $A \oplus B$.
- Allows recipient to reconstruct A.

Quantum Key Distribution

- Bennet Brassard 1984
  “Quantum cryptography: Public key distribution and coin tossing”
  Information reconciliation by public discussion over a classical channel.

Resilient Set Vectorization

A **set vectorizer** ϕ takes as input a set A, an integer n, and a nonce N, and produces as output a uniformly chosen (over the choice of nonce) vector from A^n.

A **resilient set vectorizer** is a set vectorizer with the property that for any two sets A and B with $|A \cap B| = p \cdot |A \cup B|$ (for some p, $0 \le p \le 1$), we have

$$d(\phi(A, n, N), \phi(B, n, N)) \sim n - \mathrm{Bin}(n, p) \ .$$

That is, if a fraction p of $A \cup B$ are shared, then the fraction of positions where $\phi(A, n, N)$ and $\phi(B, n, N)$ agree follows the binomial distribution with mean np.

Keyword Matching Game ($\equiv$ RSV with $n = 1$)

- Alice and Bob agree on a strategy.

Keyword Matching Game ($\equiv$ RSV with $n = 1$)

- Alice and Bob agree on a strategy.
- Alice is given an arbitrary keyring A.

Keyword Matching Game ($\equiv$ RSV with $n = 1$)

- Alice and Bob agree on a strategy.
- Alice is given an arbitrary keyring A.
- Bob is given an arbitrary keyring B.

Keyword Matching Game ($\equiv$ RSV with $n = 1$)

- Alice and Bob agree on a strategy.
- Alice is given an arbitrary keyring A.
- Bob is given an arbitrary keyring B.
- They are told sizes of A, B, $A \cap B$, $A \cup B$, $\mathcal{U}$.

Keyword Matching Game ($\equiv$ RSV with $n = 1$)

- Alice and Bob agree on a strategy.
- Alice is given an arbitrary keyring A.
- Bob is given an arbitrary keyring B.
- They are told sizes of A, B, $A \cap B$, $A \cup B$, $\mathcal{U}$.
- They are given the *same* random nonce N.

Keyword Matching Game ($\equiv$ RSV with $n = 1$)

- Alice and Bob agree on a strategy.
- Alice is given an arbitrary keyring A.
- Bob is given an arbitrary keyring B.
- They are told sizes of A, B, $A \cap B$, $A \cup B$, $\mathcal{U}$.
- They are given the *same* random nonce N.
- Alice and Bob separately each pick *one* element from their keyrings.

Keyword Matching Game ($\equiv$ RSV with $n = 1$)

- Alice and Bob agree on a strategy.
- Alice is given an arbitrary keyring A.
- Bob is given an arbitrary keyring B.
- They are told sizes of A, B, $A \cap B$, $A \cup B$, $\mathcal{U}$.
- They are given the *same* random nonce N.
- Alice and Bob separately each pick *one* element from their keyrings.
- *What is the maximum probability that they pick the same element, using optimal strategy?*

CSAIL

Simplest interesting example

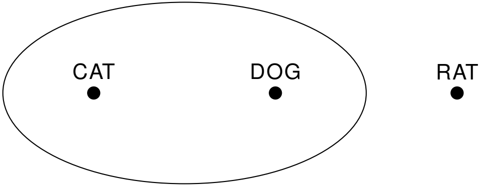
$$|A| = 2 \qquad |A \cap B| = 1 \quad |B| = 2 \qquad |\mathcal{U}| = 3$$

CAT DOG RAT

Simplest interesting example

Alice sees:

$|A| = 2 \quad |A \cap B| = 1 \quad |B| = 2 \quad |\mathcal{U}| = 3$



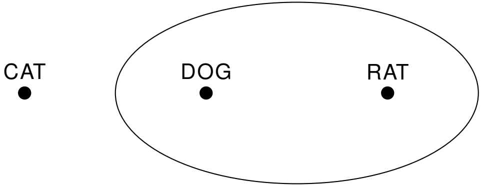
$N = 3762134912$

Should Alice pick CAT or DOG?

Simplest interesting example

Bob sees:

$$|A| = 2 \quad |A \cap B| = 1 \quad |B| = 2 \quad |\mathcal{U}| = 3$$

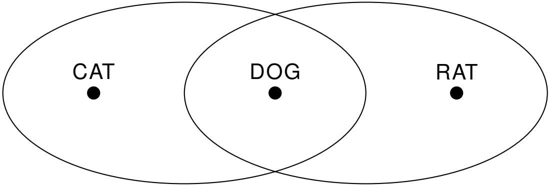


$N = 3762134912$

Should Bob pick DOG or RAT?

Simplest interesting example

$|A| = 2 \quad |A \cap B| = 1 \quad |B| = 2 \quad |\mathcal{U}| = 3$



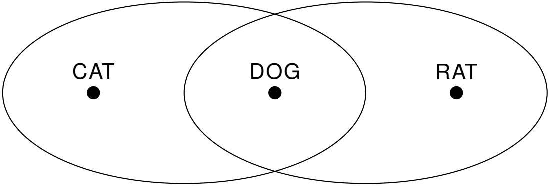
$N = 3762134912$

Should Alice pick CAT or DOG?

Should Bob pick DOG or RAT?

Simplest interesting example

$|A| = 2 \quad |A \cap B| = 1 \quad |B| = 2 \quad |\mathcal{U}| = 3$



$N = 3762134912$

Should Alice pick CAT or DOG?

Should Bob pick DOG or RAT?

Agree with prob 1/4? 1/3? 1/2?...

Keyword Matching Game – Random Strategy

- If Alice and Bob make their choices **independently at random**, then they match with probability

$$|A \cap B| / |A| \, |B| \ .$$

Keyword Matching Game – Random Strategy

- If Alice and Bob make their choices **independently at random**, then they match with probability

$$|A \cap B| / |A|\,|B| \ .$$

- (Pretty small, especially when A and B are large.)

Keyword Matching Game for $|A \cap B| = 1$

Brute-force searches for optimal strategies (surprisingly) suggested the following

Theorem

When $|A \cap B| = 1$ *and* $A \cup B = \mathcal{U}$ *the optimum match probability is at least*

$$1/\max(|A|, |B|)\ .$$

Keyword Matching Game for $|A \cap B| = 1$

Brute-force searches for optimal strategies (surprisingly) suggested the following

Theorem

When $|A \cap B| = 1$ and $A \cup B = \mathcal{U}$ the optimum match probability is at least

$$1/\max(|A|, |B|) \; .$$

Proof: (at end). ∎

Keyword Matching Game for $|A \cap B| = 1$

Brute-force searches for optimal strategies (surprisingly) suggested the following

Theorem

When $|A \cap B| = 1$ and $A \cup B = \mathcal{U}$ the optimum match probability is at least

$$1/\max(|A|, |B|) .$$

Proof: (at end). ∎

Exercise: Find such an optimal strategy for our example that matches with probability $1/2$.

Keyword Matching Game for $|A \cap B| = 1$

Brute-force searches for optimal strategies (surprisingly) suggested the following

Theorem

When $|A \cap B| = 1$ and $A \cup B = \mathcal{U}$ the optimum match probability is at least

$$1/\max(|A|, |B|) .$$

Proof: (at end). ∎

Exercise: Find such an optimal strategy for our example that matches with probability $1/2$.

But $|A \cap B| = 1$ and $A \cup B = \mathcal{U}$ are unrealistic...

Jaccard Index of Similarity

- The **Jaccard similarity coefficient** $J(A, B)$ measures the similarity of two sets A and B:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} .$$

CSAIL

Jaccard Index of Similarity

- The **Jaccard similarity coefficient** $J(A,B)$ measures the similarity of two sets A and B:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} \; .$$

- It can be estimated using the **MinHash** method (Broder 1997): Construct n random hash functions mapping elements to real values. Compute the fraction f of them having the same minimum in A as in B. Then

$$E(f) = J(A,B) \; .$$

Keyword Matching Game via MinHash

Theorem

Alice and Bob can always win with probability at least $p = J(A, B) = |A \cap B| / |A \cup B|$.

Keyword Matching Game via MinHash

Theorem

Alice and Bob can always win with probability at least $p = J(A, B) = |A \cap B| / |A \cup B|$.

Proof.

- Initially, Alice and Bob agree on a random hash function h.
- They each pick their keyword with minimum hash-value.
- They win if one of their shared keywords has the smallest hash value in both sets.

Keyword Matching Game via MinHash

Theorem

Alice and Bob can always win with probability at least $p = J(A, B) = |A \cap B| / |A \cup B|$.

Proof.

- Initially, Alice and Bob agree on a random hash function h.
- They each pick their keyword with minimum hash-value.
- They win if one of their shared keywords has the smallest hash value in both sets.

Conjecture: The MinHash strategy is *optimal* for $|A \cap B| > 1$.

Resilient Set Vectorization (RSV)

Alice iterates the MinHash method (with n random hash functions), to create a **keyword vector**

$$W = \phi(A, n, N) = (W_1, W_2, \ldots, W_n)$$

of some desired length n.

Resilient Set Vectorization (RSV)

Alice iterates the MinHash method (with n random hash functions), to create a **keyword vector**

$$W = \phi(A, n, N) = (W_1, W_2, \ldots, W_n)$$

of some desired length n.

Bob (using same hashes) similarly creates a keyword vector W'.

Resilient Set Vectorization (RSV)

Alice iterates the MinHash method (with n random hash functions), to create a **keyword vector**

$$W = \phi(A, n, N) = (W_1, W_2, \ldots, W_n)$$

of some desired length n.

Bob (using same hashes) similarly creates a keyword vector W'.

Let z denote the number of positions in which W and W' agree, and let $p = J(A, B)$. Then (under ROM)

$$z \sim \mathrm{Bin}(n, p),$$

so $E(z) = np$ and $\sigma(z) = \sqrt{np(1-p)}$.

Security Analysis Setup

Suppose we can arrange things so that Bob *can always decrypt* Alice's ciphertext if

$$z \geq 3n/4 \; .$$

Security Analysis Setup

Suppose we can arrange things so that Bob *can always decrypt* Alice's ciphertext if

$$z \geq 3n/4 \; .$$

Suppose further we can arrange things so that the Adversary *can't decrypt* Alice's ciphertext if the number z' of positions of W it knows (or guesses) correctly satisfies

$$z' < n/2 \; .$$

Analysis–for the good guys

- Suppose Alice and Bob have

$$p = J(A, B) = 0.90 \ .$$

Analysis–for the good guys

- Suppose Alice and Bob have

$$p = J(A, B) = 0.90 \ .$$

- Alice encrypts a message to Bob using $\phi(A, n, N)$ as a key, where $n = 256$.

Analysis–for the good guys

- Suppose Alice and Bob have

$$p = J(A, B) = 0.90 \ .$$

- Alice encrypts a message to Bob using $\phi(A, n, N)$ as a key, where $n = 256$.
- Bob's vector $\phi(B, n, N)$ agrees with $\phi(A, n, N)$ in z positions.

Analysis–for the good guys

- Suppose Alice and Bob have

$$p = J(A, B) = 0.90 \ .$$

- Alice encrypts a message to Bob using $\phi(A, n, N)$ as a key, where $n = 256$.
- Bob's vector $\phi(B, n, N)$ agrees with $\phi(A, n, N)$ in z positions.
- If $z \geq 192$, Bob can decrypt the message.

Analysis–for the good guys

- Suppose Alice and Bob have

$$p = J(A, B) = 0.90 \ .$$

- Alice encrypts a message to Bob using $\phi(A, n, N)$ as a key, where $n = 256$.
- Bob's vector $\phi(B, n, N)$ agrees with $\phi(A, n, N)$ in z positions.
- If $z \geq 192$, Bob can decrypt the message.
- Bob fails to decrypt with near-zero probability:

$$\mathrm{Prob}\,(z < 192) = 1.5 \times 10^{-12} \ .$$

Analysis–for the Adversary

- Suppose Adversary knows (or guesses) Q, a set of $1/4$ of Alice's keyring A, so

$$p' = J(A, Q) = 0.25 \ .$$

Analysis–for the Adversary

- Suppose Adversary knows (or guesses) Q, a set of $1/4$ of Alice's keyring A, so
$$p' = J(A, Q) = 0.25 \; .$$
- Alice encrypts a message to Bob using $\phi(A, n, N)$ as a key; Adversary overhears ciphertext.

Analysis–for the Adversary

- Suppose Adversary knows (or guesses) Q, a set of $1/4$ of Alice's keyring A, so
$$p' = J(A, Q) = 0.25 \ .$$
- Alice encrypts a message to Bob using $\phi(A, n, N)$ as a key; Adversary overhears ciphertext.
- Adversary's vector $\phi(Q, n, N)$ agrees with Alice's in z' positions.

Analysis–for the Adversary

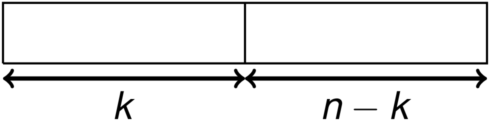
- Suppose Adversary knows (or guesses) Q, a set of $1/4$ of Alice's keyring A, so
$$p' = J(A, Q) = 0.25 \ .$$
- Alice encrypts a message to Bob using $\phi(A, n, N)$ as a key; Adversary overhears ciphertext.
- Adversary's vector $\phi(Q, n, N)$ agrees with Alice's in z' positions.
- If $z' \geq 128$, Adversary can decrypt message.

Analysis–for the Adversary

- Suppose Adversary knows (or guesses) Q, a set of $1/4$ of Alice's keyring A, so
$$p' = J(A, Q) = 0.25 \ .$$
- Alice encrypts a message to Bob using $\phi(A, n, N)$ as a key; Adversary overhears ciphertext.
- Adversary's vector $\phi(Q, n, N)$ agrees with Alice's in z' positions.
- If $z' \geq 128$, Adversary can decrypt message.
- But Adversary fails almost certainly, since
$$\text{Prob}\,(z' \geq 128) = 7.5 \times 10^{-18} \ .$$

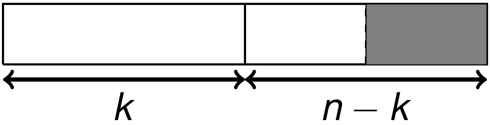
Error Correction

- An (n, k) Reed-Solomon code has k information symbols and codewords of length n.



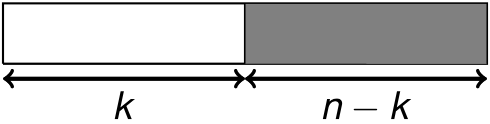
Error Correction

- An (n, k) Reed-Solomon code has k information symbols and codewords of length n.
- Bob can efficiently correct up to $(n - k)/2$ errors and always obtain a **unique decoding**.



Error Correction

- An (n, k) Reed-Solomon code has k information symbols and codewords of length n.
- Bob can efficiently correct up to $(n-k)/2$ errors and always obtain a **unique decoding**.
- With **list decoding** Adversary can efficiently correct up to $(n-k)$ errors (and obtain a small number of possible decodings).



Keyring proposal for encrypting M with keyring A

M

A

Keyring proposal for encrypting M with keyring A

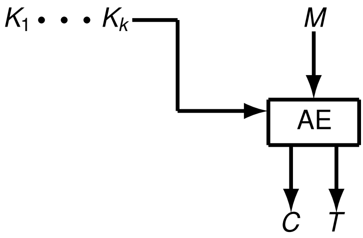
$K_1 \bullet \bullet \bullet K_k$

M

A

Keyring proposal for encrypting M with keyring A

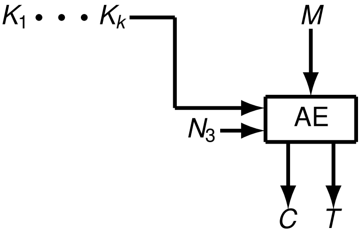
A



Alice sends C, and T.

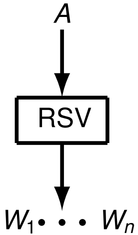
Keyring proposal for encrypting *M* with keyring *A*

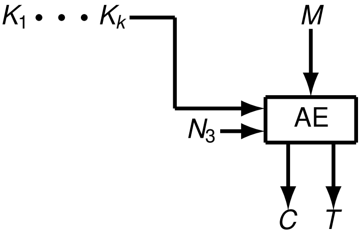
A



Alice sends (N_3), C, and T.

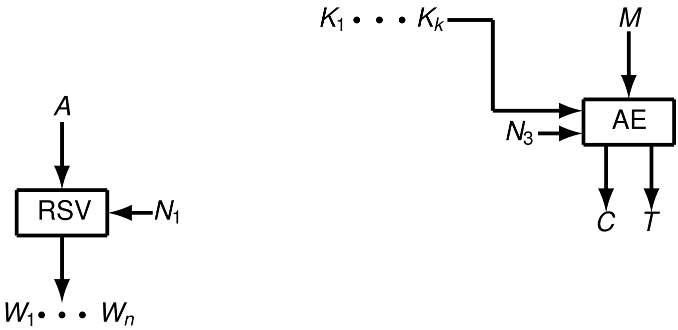
Keyring proposal for encrypting M with keyring A





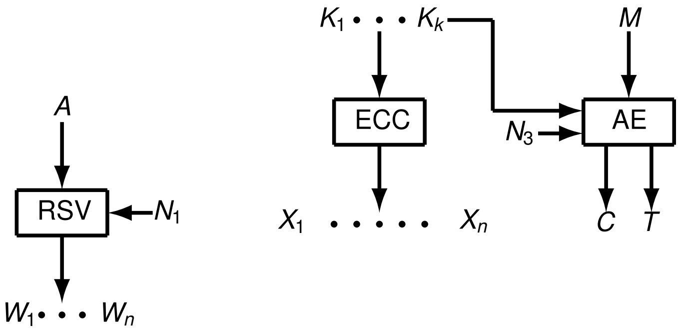
Alice sends (N_3), C, and T.

Keyring proposal for encrypting M with keyring A



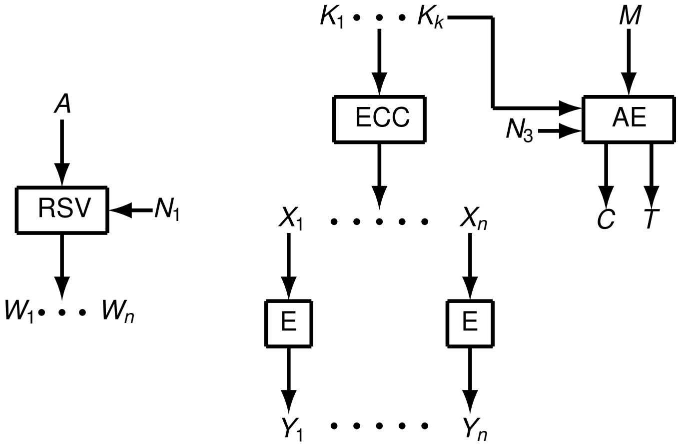
Alice sends (N_1, N_3), C, and T.

Keyring proposal for encrypting M with keyring A



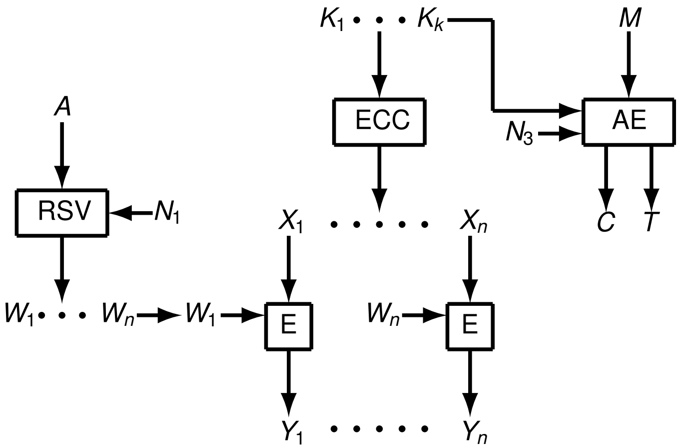
Alice sends (N_1, N_3), C, and T.

Keyring proposal for encrypting M with keyring A



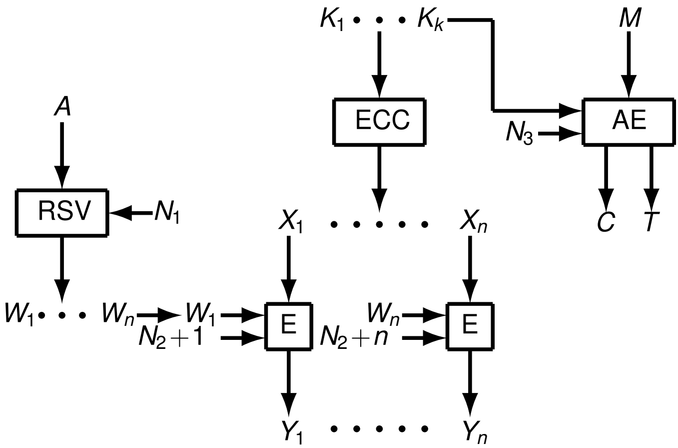
Alice sends (N_1, N_3), Y, C, and T.

Keyring proposal for encrypting M with keyring A



Alice sends (N_1, N_3), Y, C, and T.

Keyring proposal for encrypting M with keyring A



Alice sends (N_1, N_2, N_3), Y, C, and T.

Compute nonces, K, C, T

- Choose random nonces N_1, N_2, N_3.

Compute nonces, K, C, T

- Choose random nonces N_1, N_2, N_3.
- Choose n and k (e.g. $n = 256, k = 128$) and byte size ($GF(2^8)$).

Compute nonces, K, C, T

- Choose random nonces N_1, N_2, N_3.
- Choose n and k (e.g. $n = 256, k = 128$) and byte size ($GF(2^8)$).
- Choose random k-byte message key $K_1, \ldots, K_k$ (aka "vault contents").

Compute nonces, K, C, T

- Choose random nonces N_1, N_2, N_3.
- Choose n and k (e.g. $n = 256, k = 128$) and byte size ($GF(2^8)$).
- Choose random k-byte message key $K_1, \ldots, K_k$ (aka "vault contents").
- Encrypt message M with key K and nonce N_3 using an authenticated encryption method to obtain ciphertext C and authentication tag T.

CSAIL

Compute W, X, and Y

- Compute keyword vector $W = \phi(A, n, N_1)$.

Compute W, X, and Y

- Compute keyword vector $W = \phi(A, n, N_1)$.
- Reed-Solomon-encode key to give n-byte encoded key $X_1, \ldots, X_n$.

CSAIL

Compute W, X, and Y

- Compute keyword vector $W = \phi(A, n, N_1)$.
- Reed-Solomon-encode key to give n-byte encoded key $X_1, \ldots, X_n$.
- Use each keyword vector element W_i as key to encrypt each encoded key byte X_i:

$$Y_i = E(W_i, X_i, N_2 + i)$$

  use small-domain encryption tweakable encryption method like “swap-or-not” (Hoang-Morris-Rogaway14).

Compute W, X, and Y

- Compute keyword vector $W = \phi(A, n, N_1)$.
- Reed-Solomon-encode key to give n-byte encoded key $X_1, \ldots, X_n$.
- Use each keyword vector element W_i as key to encrypt each encoded key byte X_i:

  $$Y_i = E(W_i, X_i, N_2 + i)$$

  use small-domain encryption tweakable encryption method like “swap-or-not” (Hoang-Morris-Rogaway14).
- Send $(N_1, N_2, N_3), Y, C, T$.

Decrypting (N_1, N_2, N_3), Y, C, T with keyring B

B

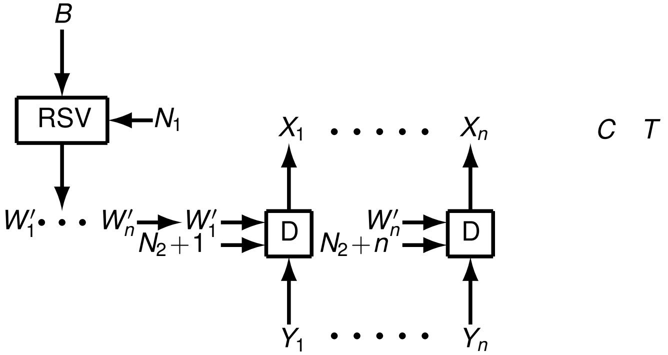
C T

Bob receives (N_1, N_2, N_3), Y, C, and T.

Decrypting (N_1, N_2, N_3), Y, C, T with keyring B

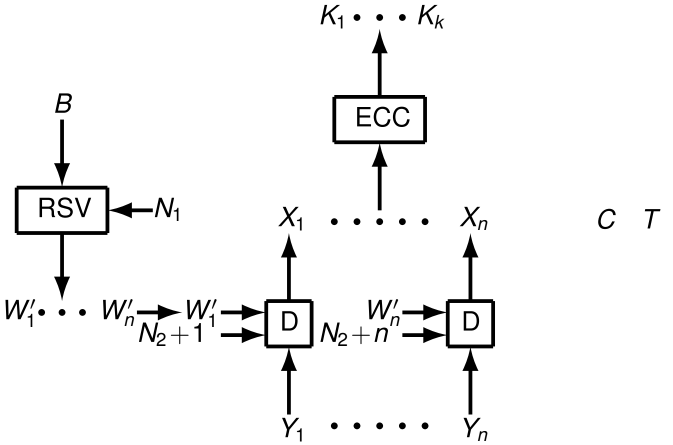
Bob receives (N_1, N_2, N_3), Y, C, and T.

Decrypting (N_1, N_2, N_3), Y, C, T with keyring B



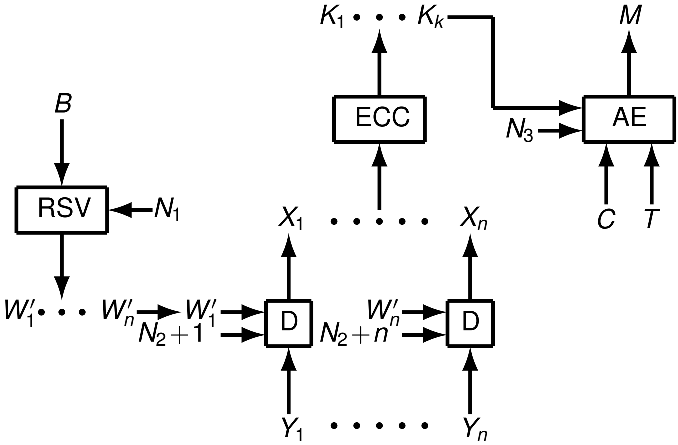
Bob receives (N_1, N_2, N_3), Y, C, and T.

Decrypting (N_1, N_2, N_3), Y, C, T with keyring B



Bob receives (N_1, N_2, N_3), Y, C, and T.

Decrypting (N_1, N_2, N_3), Y, C, T with keyring B



Bob receives (N_1, N_2, N_3), Y, C, and T.

Attack 1: Guessing large subset of A

- Adversary may try to guess a large subset of A.

Attack 1: Guessing large subset of A

- Adversary may try to guess a large subset of A.
- Difficulty depends on A. Even if $|\mathcal{U}| = 4096$ and $|A| = 24$ (chosen uniformly), guessing a 12-word subset of A has chance

$$\binom{24}{12} \Big/ \binom{4096}{12} \approx 2^{-94}$$

of success.

Attack 1: Guessing large subset of A

- Adversary may try to guess a large subset of A.
- Difficulty depends on A. Even if $|\mathcal{U}| = 4096$ and $|A| = 24$ (chosen uniformly), guessing a 12-word subset of A has chance

$$\binom{24}{12} \Big/ \binom{4096}{12} \approx 2^{-94}$$

  of success.
- Using keyrings may invite poor choices (just as passwords tend to be poor). "Biometric" keyrings don't have this problem.

Attack 1: Guessing large subset of A

- Adversary may try to guess a large subset of A.
- Difficulty depends on A. Even if $|\mathcal{U}| = 4096$ and $|A| = 24$ (chosen uniformly), guessing a 12-word subset of A has chance
$$\binom{24}{12} \Big/ \binom{4096}{12} \approx 2^{-94}$$
of success.
- Using keyrings may invite poor choices (just as passwords tend to be poor). "Biometric" keyrings don't have this problem.
- Initial keywords may be high-entropy.

Attack 2: Stealing *A*, then tracking its evolution

- Stealing A, then tracking its evolution if updates are small.

Attack 2: Stealing A, then tracking its evolution

- Stealing A, then tracking its evolution if updates are small.
- Make updates large every once in a while!

Attack 2: Stealing A, then tracking its evolution

- Stealing A, then tracking its evolution if updates are small.
- Make updates large every once in a while!
- Reminiscent of problems of refreshing entropy pool in PRNG. (Ferguson-Schneier-Kohn'10, Dodis-Shamir-StephensDavidowitz-Wich'14).

Attack 3: Playing Matching Game better

- We only *conjectured* that MinHash strategy was best way to play Keyword Matching Game.

Attack 3: Playing Matching Game better

- We only *conjectured* that MinHash strategy was best way to play Keyword Matching Game.
- Perhaps Adversary can play this game better than Bob can, even for a fixed strategy by Alice!

Attack 3: Playing Matching Game better

- We only *conjectured* that MinHash strategy was best way to play Keyword Matching Game.
- Perhaps Adversary can play this game better than Bob can, even for a fixed strategy by Alice!
- We need to prove that MinHash strategy is optimal (for $|A \cap B| > 1$)!

Attack 4: Chosen ciphertext attack

- Given a valid ciphertext, Adversary can use Bob as a pass/fail decryption oracle to do a sensitivity analysis disclosing where he has correct keywords.

Attack 4: Chosen ciphertext attack

- Given a valid ciphertext, Adversary can use Bob as a pass/fail decryption oracle to do a sensitivity analysis disclosing where he has correct keywords.
- Serious! Adversary may compute set of candidate words with small MinHash values in each such position. These are good candidates for being in B.

Attack 4: Chosen ciphertext attack

- Given a valid ciphertext, Adversary can use Bob as a pass/fail decryption oracle to do a sensitivity analysis disclosing where he has correct keywords.
- Serious! Adversary may compute set of candidate words with small MinHash values in each such position. These are good candidates for being in B.
- Encrypt M with AEAD instead of AE, where AD includes Y and nonces. Insecure? (AD and K are related.) Proof needed.

Comparison with PinSketch

- Keyring scheme is not a “sketch”—Bob can’t recover A.

Comparison with PinSketch

- Keyring scheme is not a "sketch"—Bob can't recover A.
- Keyring scheme isn't restricted to certain error codes (e.g. algebraic codes).

Comparison with PinSketch

- Keyring scheme is not a “sketch”—Bob can’t recover A.
- Keyring scheme isn’t restricted to certain error codes (e.g. algebraic codes).
- We don’t require bounded $|\mathcal{U}|$.

Comparison with PinSketch

- Keyring scheme is not a "sketch"—Bob can't recover A.
- Keyring scheme isn't restricted to certain error codes (e.g. algebraic codes).
- We don't require bounded $|\mathcal{U}|$.
- PinSketch messages have size

$$|A \oplus B| \log |\mathcal{U}| \ .$$

Comparison with PinSketch

- Keyring scheme is not a “sketch”—Bob can’t recover A.
- Keyring scheme isn’t restricted to certain error codes (e.g. algebraic codes).
- We don’t require bounded $|\mathcal{U}|$.
- PinSketch messages have size

$$|A \oplus B| \log |\mathcal{U}| \ .$$

- We send $n = 256$ bytes plus nonces.

Comparison with PinSketch

- Keyring scheme is not a “sketch”—Bob can't recover A.
- Keyring scheme isn't restricted to certain error codes (e.g. algebraic codes).
- We don't require bounded $|\mathcal{U}|$.
- PinSketch messages have size

$$|A \oplus B| \log |\mathcal{U}| \ .$$

- We send $n = 256$ bytes plus nonces.
- Bob can decode whp if $p - k/n \geq c\sqrt{np(1-p)}$, which holds for **constant** n if $p > (1+\epsilon)k/n$.

Summary

- New scheme facilitates updates of keys; these updates can now be done incrementally as well as all at once.

Summary

- New scheme facilitates updates of keys; these updates can now be done incrementally as well as all at once.
- New scheme has reduced message size.

Summary

- New scheme facilitates updates of keys; these updates can now be done incrementally as well as all at once.
- New scheme has reduced message size.
- Security is controllable via choices of n and keyring size.

Summary

- New scheme facilitates updates of keys; these updates can now be done incrementally as well as all at once.
- New scheme has reduced message size.
- Security is controllable via choices of *n* and keyring size.
- Keyword Matching Game of possible independent interest.

CSAIL

Summary

- New scheme facilitates updates of keys; these updates can now be done incrementally as well as all at once.
- New scheme has reduced message size.
- Security is controllable via choices of n and keyring size.
- Keyword Matching Game of possible independent interest.
- Open problems include

Summary

- New scheme facilitates updates of keys; these updates can now be done incrementally as well as all at once.
- New scheme has reduced message size.
- Security is controllable via choices of n and keyring size.
- Keyword Matching Game of possible independent interest.
- Open problems include
  - Determining optimal strategy in Keyword Matching Game. (Is it MinHash?)

Summary

- New scheme facilitates updates of keys; these updates can now be done incrementally as well as all at once.
- New scheme has reduced message size.
- Security is controllable via choices of n and keyring size.
- Keyword Matching Game of possible independent interest.
- Open problems include
  - Determining optimal strategy in Keyword Matching Game. (Is it MinHash?)
  - Analyzing security of AEAD variant against CCA.

CSAIL

The End

Strategy for $p = 1/\max(|A|, |B|)$ when $|A \cap B| = 1$, $\mathcal{U} = A \cup B$, $|A| \geq |B|$

- Create bipartite graph whose vertices are all $|A|$-subsets (resp. all $|B|$-subsets) of $\mathcal{U}$ with an (X, Y) edge iff $|X \cap Y| = 1$. The $|A|$-subsets have degree $|A|$; the $|B|$-subsets have degree $|B|$.

Strategy for $p = 1/\max(|A|, |B|)$ when $|A \cap B| = 1$, $\mathcal{U} = A \cup B$, $|A| \geq |B|$

- Create bipartite graph whose vertices are all $|A|$-subsets (resp. all $|B|$-subsets) of $\mathcal{U}$ with an (X, Y) edge iff $|X \cap Y| = 1$. The $|A|$-subsets have degree $|A|$; the $|B|$-subsets have degree $|B|$.
- By Hall's Thm you can find a matching that covers all $|A|$-subsets.

Strategy for $p = 1/\max(|A|, |B|)$ when $|A \cap B| = 1$, $\mathcal{U} = A \cup B$, $|A| \geq |B|$

- Create bipartite graph whose vertices are all $|A|$-subsets (resp. all $|B|$-subsets) of $\mathcal{U}$ with an (X, Y) edge iff $|X \cap Y| = 1$. The $|A|$-subsets have degree $|A|$; the $|B|$-subsets have degree $|B|$.
- By Hall's Thm you can find a matching that covers all $|A|$-subsets.
- Alice and Bob each choose keyword shared with their matched subset (if any).

Strategy for $p = 1/\max(|A|, |B|)$ when $|A \cap B| = 1$, $\mathcal{U} = A \cup B$, $|A| \geq |B|$

- Create bipartite graph whose vertices are all $|A|$-subsets (resp. all $|B|$-subsets) of $\mathcal{U}$ with an (X, Y) edge iff $|X \cap Y| = 1$. The $|A|$-subsets have degree $|A|$; the $|B|$-subsets have degree $|B|$.
- By Hall's Thm you can find a matching that covers all $|A|$-subsets.
- Alice and Bob each choose keyword shared with their matched subset (if any).
- They pick the same keyword with probability $1/|A| = 1/\max(|A|, |B|)$.

Strategy for $p = 1/\max(|A|, |B|)$ when $|A \cap B| = 1$, $\mathcal{U} = A \cup B$, $|A| \geq |B|$

- Create bipartite graph whose vertices are all $|A|$-subsets (resp. all $|B|$-subsets) of $\mathcal{U}$ with an (X, Y) edge iff $|X \cap Y| = 1$. The $|A|$-subsets have degree $|A|$; the $|B|$-subsets have degree $|B|$.
- By Hall's Thm you can find a matching that covers all $|A|$-subsets.
- Alice and Bob each choose keyword shared with their matched subset (if any).
- They pick the same keyword with probability $1/|A| = 1/\max(|A|, |B|)$.
- (This only works for $|A \cap B| = 1$. ☹)