Rerandomizable RCCA Encryption

Manoj Prabhakaran & Mike Rosulek

ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

CRYPTO 2007 – August 23, 2007

Thanks to Qualcomm for travel support

Unlinkable Blind Copying

Imagine encryption scheme with following feature

- Alice sends $C = \mathrm{Enc}(m)$ to Bob

Unlinkable Blind Copying

Imagine encryption scheme with following feature

- Alice sends $C = \mathrm{Enc}(m)$ to Bob
- Carl sees C, generates a "copy" C' which also decrypts to m.

Unlinkable Blind Copying

Imagine encryption scheme with following feature

- Alice sends $C = \text{Enc}(m)$ to Bob
- Carl sees C, generates a “copy” C' which also decrypts to m.
- Eve can’t tell whether C' was a “copy” or fresh encryption
- “Unlinkable blind copy”

Unlinkable Blind Copying

Imagine encryption scheme with following feature

- Alice sends $C = \mathrm{Enc}(m)$ to Bob
- Carl sees C, generates a "copy" C' which also decrypts to m.
- Eve can't tell whether C' was a "copy" or fresh encryption
- "Unlinkable blind copy"

Malleability is now a concern, but...

Unlinkable Blind Copying

Imagine encryption scheme with following feature

- Alice sends $C = \text{Enc}(m)$ to Bob
- Carl sees C, generates a “copy” C' which also decrypts to m.
- Eve can’t tell whether C' was a “copy” or fresh encryption
- “Unlinkable blind copy”

Malleability is now a concern, but...

- CCA security rules out copying feature!

Unlinkable Blind Copying

Imagine encryption scheme with following feature

- Alice sends $C = \mathrm{Enc}(m)$ to Bob
- Carl sees C, generates a "copy" C' which also decrypts to m.
- Eve can't tell whether C' was a "copy" or fresh encryption
- "Unlinkable blind copy"

Malleability is now a concern, but...

- CCA security rules out copying feature!
- Ideally, want scheme to be "non-malleable except for copying"

Unlinkable Blind Copying

Imagine encryption scheme with following feature

- Alice sends $C = \mathsf{Enc}(m)$ to Bob
- Carl sees C, generates a "copy" C' which also decrypts to m.
- Eve can't tell whether C' was a "copy" or fresh encryption
- "Unlinkable blind copy"

Malleability is now a concern, but...

- CCA security rules out copying feature!
- Ideally, want scheme to be "non-malleable except for copying"

Strong tradeoff between features and non-malleability:

- Copying is allowed, in very robust sense

Unlinkable Blind Copying

Imagine encryption scheme with following feature

- Alice sends $C = \mathsf{Enc}(m)$ to Bob
- Carl sees C, generates a "copy" C' which also decrypts to m.
- Eve can't tell whether C' was a "copy" or fresh encryption
- "Unlinkable blind copy"

Malleability is now a concern, but...

- CCA security rules out copying feature!
- Ideally, want scheme to be "non-malleable except for copying"

Strong tradeoff between features and non-malleability:

- Copying is allowed, in very robust sense
- Everything else is forbidden, in the strongest sense

Formalizing the Problem

Rerandomizability

For any $C \leftarrow \mathsf{Enc}_{PK}(m)$, $\mathsf{Rerand}(C)$ has same distribution as $\mathsf{Enc}_{PK}(m)$.

Formalizing the Problem

Rerandomizability

For any $C \leftarrow \mathrm{Enc}_{PK}(m)$, $\mathrm{Rerand}(C)$ has same distribution as $\mathrm{Enc}_{PK}(m)$.

Replayable-CCA (RCCA) security [Canetti, Krawczyk, Nielsen]

Scheme is CCA secure, except it may be possible to "maul" an encryption of m into another that decrypts to same m.

Formalizing the Problem

Rerandomizability

For any $C \leftarrow \mathsf{Enc}_{PK}(m)$, $\mathsf{Rerand}(C)$ has same distribution as $\mathsf{Enc}_{PK}(m)$.

Replayable-CCA (RCCA) security [Canetti, Krawczyk, Nielsen]

Scheme is CCA secure, except it may be possible to "maul" an encryption of m into another that decrypts to same m.

Question [Canetti, Krawczyk, Nielsen]

Are there rerandomizable, RCCA-secure encryption schemes?

Related work

Canetti, Krawczyk, Nielsen [*CRYPTO*'03]

- Defined RCCA, posed problem.

Related work

Canetti, Krawczyk, Nielsen [*CRYPTO*'03]

- Defined RCCA, posed problem.

Golle, Jakobsson, Juels, Syverson [*RSA*'04]

- Applications of rerandomizability for *anonymous* schemes
- Rerandomizable, CPA-secure scheme

Related work

Canetti, Krawczyk, Nielsen [*CRYPTO*'03]

- Defined RCCA, posed problem.

Golle, Jakobsson, Juels, Syverson [*RSA*'04]

- Applications of rerandomizability for *anonymous* schemes
- Rerandomizable, CPA-secure scheme

Gröth [*TCC*'04] gives two rerandomizable schemes:

- Achieves weaker variant of RCCA security
- Achieves full RCCA in *generic group* model

Our Results

First rerandomizable RCCA scheme in standard model.

- Security proven under DDH assumption.
- More efficient than Gröth's schemes.

Our Results

First rerandomizable RCCA scheme in standard model.

- Security proven under DDH assumption.
- More efficient than Gröth's schemes.

Natural security characterization in the UC framework.

- Justifies choice of security definitions
- Positive result for sophisticated functionality in standard UC model

Our Results

First rerandomizable RCCA scheme in standard model.

- Security proven under DDH assumption.
- More efficient than Gröth's schemes.

Natural security characterization in the UC framework.

- Justifies choice of security definitions
- Positive result for sophisticated functionality in standard UC model

Full version (and this talk) has improved construction

- http://eprint.iacr.org/2007/119

Recipe

Key Idea #1

Use two “strands” of Cramer-Shoup encryption

- First strand is usual encryption
- Second strand helps with rerandomization

Recipe

Key Idea #1

Use two “strands” of Cramer-Shoup encryption

- First strand is usual encryption
- Second strand helps with rerandomization

Key Idea #2

“Tie strands together” with correlated randomness

- The 2 strands useful only when combined with each other

Recipe

Key Idea #1

Use two “strands” of Cramer-Shoup encryption

- First strand is usual encryption
- Second strand helps with rerandomization

Key Idea #2

“Tie strands together” with correlated randomness

- The 2 strands useful only when combined with each other

Key Idea #3

“Twist first strand” (technical)

- Make the 2 strands of different type
- Avoid bad ways of combining the 2 strands together

Double-Strand Idea

Key Idea #1

Use two "strands" of Cramer-Shoup encryption

- First strand is usual encryption
- Second strand helps with rerandomization

Double-Strand Idea

Key Idea #1

Use two "strands" of Cramer-Shoup encryption

- First strand is usual encryption
- Second strand helps with rerandomization

Golle et al. [*RSA*'04] use a double-strand idea with ElGamal.

- In their case, second strand carries public key data needed to rerandomize ElGamal.

Double-Strand Idea

Key Idea #1

Use two "strands" of Cramer-Shoup encryption

- First strand is usual encryption
- Second strand helps with rerandomization

Golle et al. [*RSA*'04] use a double-strand idea with ElGamal.

- In their case, second strand carries public key data needed to rerandomize ElGamal.
- In our case, second strand carries message data needed for rerandomize Cramer-Shoup.

Double-Strand ElGamal

Double-strand idea illustrated with ElGamal [Golle et al]:

- Normal ElGamal encryption of m (public key is g^a):

$$g^x, m\,(g^a)^x \qquad \text{for random } x$$

Double-Strand ElGamal

Double-strand idea illustrated with ElGamal [Golle et al]:

- Normal ElGamal encryption of m (public key is g^a):

$$g^x, m\,(g^a)^x \qquad \text{for random } x$$

- "Double-Strand" ElGamal encryption of m:

$$g^x, m\,g^{ax}, g^y, g^{ay} \qquad \text{for random } x, y$$

Double-Strand ElGamal

Double-strand idea illustrated with ElGamal [Golle et al]:

- Normal ElGamal encryption of m (public key is g^a):

$$g^x, m\,(g^a)^x \qquad \text{for random } x$$

- "Double-Strand" ElGamal encryption of m:

$$g^x, m\,g^{ax}, g^y, g^{ay} \qquad \text{for random } x, y$$

- Rerandomize x additively

$$\begin{aligned} & g^x(g^y)^s, \quad m\,g^{ax}(g^{ay})^s, \qquad \text{for rand } s \\ = \quad & g^{x+ys}, \quad m\,g^{a(x+ys)}, \end{aligned}$$

Double-Strand ElGamal

Double-strand idea illustrated with ElGamal [Golle et al]:

- Normal ElGamal encryption of m (public key is g^a):

$$g^x, m\,(g^a)^x \qquad \text{for random } x$$

- "Double-Strand" ElGamal encryption of m:

$$g^x, m\,g^{ax}, g^y, g^{ay} \qquad \text{for random } x, y$$

- Rerandomize x additively, y multiplicatively

$$\begin{array}{lllll} & g^x(g^y)^s, & m\,g^{ax}(g^{ay})^s, & (g^y)^t, & (g^{ay})^t \quad \text{for rand } s,\ t \\ = & g^{x+ys}, & m\,g^{a(x+ys)}, & g^{yt}, & g^{a(yt)} \end{array}$$

Double-Strand ElGamal

Double-strand idea illustrated with ElGamal [Golle et al]:

- Normal ElGamal encryption of m (public key is g^a):

$$g^x, m\,(g^a)^x \qquad \text{for random } x$$

- "Double-Strand" ElGamal encryption of m:

$$g^x, m\,g^{ax}, g^y, g^{ay} \qquad \text{for random } x, y$$

- Rerandomize x additively, y multiplicatively

$$\begin{array}{rllllr} & g^x(g^y)^s, & m\,g^{ax}(g^{ay})^s, & (g^y)^t, & (g^{ay})^t & \text{for rand } s,\ t \\ = & g^{x+ys}, & m\,g^{a(x+ys)}, & g^{yt}, & g^{a(yt)} & \end{array}$$

Result is distributed as fresh double-strand encryption

Double-Strands for Cramer-Shoup

We apply double-strand idea to Cramer-Shoup:

- Normal Cramer-Shoup: (μ is hash of first 3 components)

$$g_1^x, g_2^x, m\, B^x, (CD^\mu)^x \qquad \text{for random } x$$

Double-Strands for Cramer-Shoup

We apply double-strand idea to Cramer-Shoup:

- Normal Cramer-Shoup: (μ is hash of first 3 components)

$$g_1^x, g_2^x, m\,B^x, (CD^\mu)^x \qquad \text{for random } x$$

- Double-strand idea applied: (μ is encoding of message)

$$g_1^x, g_2^x, m\,B^x, (CD^\mu)^x, g_1^y, g_2^y, B^y, (CD^\mu)^y \qquad \text{for rand } x, y$$

Double-Strands for Cramer-Shoup

We apply double-strand idea to Cramer-Shoup:

- Normal Cramer-Shoup: (μ is hash of first 3 components)

$$g_1^x, g_2^x, m\,B^x, (CD^\mu)^x \qquad \text{for random } x$$

- Double-strand idea applied: (μ is encoding of message)

$$g_1^x, g_2^x, m\,B^x, (CD^\mu)^x, g_1^y, g_2^y, B^y, (CD^\mu)^y \qquad \text{for rand } x, y$$

Possible Attack!

Can "mix-and-match" strands from 2 independent ciphertexts, if they carry the same message (μ).

Tie the Knot

Key Idea #2

"Tie strands together" with shared randomness u; give an additional encryption of u

$$(g_1^x)^u, (g_2^x)^u, mB^x, (CD^\mu)^x, (g_1^y)^u, (g_2^y)^u, B^y, (CD^\mu)^y, \mathsf{Enc}(u)$$

Tie the Knot

Key Idea #2

"Tie strands together" with shared randomness u; give an additional encryption of u

$$(g_1^x)^u, (g_2^x)^u, mB^x, (CD^\mu)^x, (g_1^y)^u, (g_2^y)^u, B^y, (CD^\mu)^y, \mathsf{Enc}(u)$$

Thwarts "mix-and-match" attack:

- Strands only combine if they share common u
- Receiver must remove u to decrypt

Tie the Knot

Key Idea #2

"Tie strands together" with shared randomness u; give an additional encryption of u

$$(g_1^x)^u, (g_2^x)^u, mB^x, (CD^\mu)^x, (g_1^y)^u, (g_2^y)^u, B^y, (CD^\mu)^y, \mathsf{Enc}(u)$$

Thwarts "mix-and-match" attack:

- Strands only combine if they share common u
- Receiver must remove u to decrypt

Encryption of u must be:

- malleable, to allow rerandomization of u
- rerandomizable, for randomness within Enc

Tie the Knot

Key Idea #2

"Tie strands together" with shared randomness u; give an additional encryption of u

$$(g_1^x)^u, (g_2^x)^u, mB^x, (CD^\mu)^x, (g_1^y)^u, (g_2^y)^u, B^y, (CD^\mu)^y, \mathrm{Enc}(u)$$

Thwarts "mix-and-match" attack:

- Strands only combine if they share common u
- Receiver must remove u to decrypt

Encryption of u must be:

- malleable, to allow rerandomization of u
- rerandomizable, for randomness within Enc

Double-strand "Cramer-Shoup lite" has these properties.

Twist the First Strand

$$g_1^{xu}, g_2^{xu}, mB^x, (CD^\mu)^x, g_1^{yu}, g_2^{yu}, B^y, (CD^\mu)^y, \mathsf{Enc}(u)$$

Possible Attack

Can rerandomize first strand multiplicatively, if plaintext is known.

Twist the First Strand

$$g_1^{xu}, g_2^{xu}, mB^x, (CD^\mu)^x, g_1^{yu}, g_2^{yu}, B^y, (CD^\mu)^y, \mathsf{Enc}(u)$$

Possible Attack

Can rerandomize first strand multiplicatively, if plaintext is known.

Key Idea #3

Make first strand fundamentally different, so it can only be rerandomized in the prescribed way

Twist the First Strand

$$g_1^{xu}, g_2^{xu}, mB^x, (CD^\mu)^x, g_1^{yu}, g_2^{yu}, B^y, (CD^\mu)^y, \mathsf{Enc}(u)$$

Possible Attack

Can rerandomize first strand multiplicatively, if plaintext is known.

Key Idea #3

Make first strand fundamentally different, so it can only be rerandomized in the prescribed way

Analysis uses linear-algebraic interpretation:

- First strand's randomness x is perturbed by fixed vector
- More components $(g_1, \ldots, g_4)$ needed for linear independence.

Security of our Scheme

Our scheme satisfies:

- RCCA security (Canetti, Krawczyk, Nielsen [*CRYPTO*'03])
- Perfect rerandomizability
- Several correctness properties

Security of our Scheme

Our scheme satisfies:

- RCCA security (Canetti, Krawczyk, Nielsen [*CRYPTO*'03])
- Perfect rerandomizability
- Several correctness properties

Proof uses:

- DDH assumption in 2 groups (for CS and CS-lite)
  - Requires 3 large primes of special form (Cunningham chain)
- Linear algebra interpretation of our scheme

On Security Definitions

Some philosophical thoughts:

- Want scheme that is non-malleable except for unlinkable copying feature

On Security Definitions

Some philosophical thoughts:

- Want scheme that is non-malleable except for unlinkable copying feature
- Can also imagine different tradeoff (e.g, non-malleable other than some homomorphic operation)

On Security Definitions

Some philosophical thoughts:

- Want scheme that is non-malleable except for unlinkable copying feature
- Can also imagine different tradeoff (e.g, non-malleable other than some homomorphic operation)
- What are the "right" security, correctness definitions?

On Security Definitions

Some philosophical thoughts:

- Want scheme that is non-malleable except for unlinkable copying feature
- Can also imagine different tradeoff (e.g, non-malleable other than some homomorphic operation)
- What are the "right" security, correctness definitions?
- Details motivated by natural UC formulation...

Universal Composition (UC)

Universal Composition (UC) framework for security definitions

- Proposed by Canetti [*FOCS*'01]
- Simulation-based security with arbitrary interactive environment
- "Natural" formulations of security via ideal functionalities

Universal Composition (UC)

Universal Composition (UC) framework for security definitions

- Proposed by Canetti [*FOCS*'01]
- Simulation-based security with arbitrary interactive environment
- "Natural" formulations of security via ideal functionalities
- No secure protocols for most tasks (unless model is significantly weakened)!

Our Results

We define a powerful new UC functionality.

- Users send private messages to each other
- Anyone can cause previous message to be re-sent

Our Results

We define a powerful new UC functionality.

- Users send private messages to each other
- Anyone can cause previous message to be re-sent

Theorem

Any rerandomizable, RCCA-secure scheme (with our correctness properties) is a secure realization of this functionality.

Our Results

We define a powerful new UC functionality.

- Users send private messages to each other
- Anyone can cause previous message to be re-sent

Theorem

Any rerandomizable, RCCA-secure scheme (with our correctness properties) is a secure realization of this functionality.

- Justifies our security definitions, correctness properties
- Positive UC result in standard model.
- Can easily extend to add features

Our Results

First rerandomizable, RCCA-secure encryption scheme under standard assumption

- Can make unlinkable "copies" of ciphertexts
- Scheme is otherwise totally non-malleable

Our Results

First rerandomizable, RCCA-secure encryption scheme under standard assumption

- Can make unlinkable "copies" of ciphertexts
- Scheme is otherwise totally non-malleable

Natural characterization in UC framework

- Justifies choice of security definitions
- Sophisticated positive result for standard UC model

Open Problems

Anonymous, rerandomizable RCCA scheme?

- Adversary cannot tell which public key used to generate ciphertext
- Most interesting applications of rerandomizability also require anonymity

Open Problems

Anonymous, rerandomizable RCCA scheme?

- Adversary cannot tell which public key used to generate ciphertext
- Most interesting applications of rerandomizability also require anonymity

Other tradeoffs between features and non-malleability in encryption schemes

- Features can be performed in an unlinkable way
- Scheme is non-malleable otherwise

Thanks for your attention!

fin.

Carefully tying the knot

When rerandomizing u value (say, $u \rightsquigarrow ru$):

Carefully tying the knot

When rerandomizing u value (say, $u \rightsquigarrow ru$):

- Multiplication in exponent of Cramer-Shoup group (e.g, $\mathbb{Z}_p^*$):

$$g_1^{y\,u} \rightsquigarrow g_1^{(ty)(ru)}$$

Carefully tying the knot

When rerandomizing u value (say, $u \rightsquigarrow ru$):

- Multiplication in exponent of Cramer-Shoup group (e.g, $\mathbb{Z}_p^*$):

$$g_1^{yu} \rightsquigarrow g_1^{(ty)(ru)}$$

- ... and as the malleable operation of CS-lite:

$$\mathrm{Enc}(u) \rightsquigarrow \mathrm{Enc}(ru)$$

Carefully tying the knot

When rerandomizing u value (say, $u \rightsquigarrow ru$):

- Multiplication in exponent of Cramer-Shoup group (e.g, $\mathbb{Z}_p^*$):

$$g_1^{yu} \rightsquigarrow g_1^{(ty)(ru)}$$

- ... and as the malleable operation of CS-lite:

$$\mathsf{Enc}(u) \rightsquigarrow \mathsf{Enc}(ru)$$

For 2 operations to coincide, must have:

- CS-lite group is subgroup of $\mathbb{Z}_p^*$.
- DDH in both groups (that of CS and CS-lite)

Is it an unreasonable relationship between 2 groups?

Cunningham Chains

Definition

A Cunningham chain is 3 primes of the form $q, 2q+1, 4q+3$.

Cunningham Chains

Definition

A Cunningham chain is 3 primes of the form $q, 2q+1, 4q+3$.

- $\mathbb{QR}^*_{4q+3}$ and $\mathbb{QR}^*_{2q+1}$ have desired relationship
- DDH believed to hold ($4q+3$ and $2q+1$ are *safe primes*).
- Cunningham chains known to exist for $q \sim 2^{20,000}$.

Security Proof Outline

What is adversary's advantage in RCCA experiment?

normal encryption ?
normal decryption ?

Security Proof Outline

Define an "alternate encryption."

normal encryption normal decryption	$\approx$	alternate encryption normal decryption

Lemma

Alternate encryption indistinguishable from normal encryption (DDH assumption).

Security Proof Outline

normal encryption normal decryption	$\approx$	alternate encryption ✓ normal decryption

Lemma

Alternate encryption independent of choice of plaintext.

Security Proof Outline

Decryption answers might leak information about private key!

normal encryption normal decryption	$\approx$	alternate encryption ✓ normal decryption ?

Security Proof Outline

Define a (computationally unbounded) “alternate decryption.”

normal encryption / normal decryption	$\approx$	alternate encryption ✓ / normal decryption	$\approx$	alternate encryption ✓ / alternate decryption

Lemma

Alternate decryption indistinguishable from normal decryption (linear algebra analysis).

Security Proof Outline

normal encryption normal decryption	$\approx$	alternate encryption ✓ normal decryption	$\approx$	alternate encryption ✓ alternate decryption ✓

Lemma

Alternate decryption computed using only public key and challenge ciphertext.

Adversary's entire view independent of choice of plaintext!