Traceable Anonymous Encryption

Julien Cathalo[1], Malika Izabachène[2], David Pointcheval[2] and Damien Vergnaud[2]

[1] Université Catholique de Louvain, Crypto Group (Belgium)
[2] École Normale Supérieure, C.N.R.S. – I.N.R.I.A. (France)

June 28, 2009
Grenoble

About this talk

- examine **covert channels** in the context of anonymous (traceable) encryption
- introduce a new primitive: mediated anonymous traceable encryption
- give
  - *security definitions* for this new primitive
  - a *construction* meeting the formalized requirements.

- **Caveat emptor**:
  - work in submission
  - somehow marginal to the workshop, but
    - uses techniques which originated in the context of voting protocols
    - can be applied to enforce security of voting protocols

About this talk

- examine **covert channels** in the context of anonymous (traceable) encryption
- introduce a new primitive: mediated anonymous traceable encryption
- give
  - *security definitions* for this new primitive
  - a *construction* meeting the formalized requirements.
- **Caveat emptor**:
  - work in submission
  - somehow marginal to the workshop, but
    - uses techniques which originated in the context of voting protocols
    - can be applied to enforce security of voting protocols

Contents
Traceable Anonymous Encryption

1. Introduction
   - Motivation
   - Model
2. Basic scheme
   - Description
   - Security
3. Extensions and open problems
   - Protection against the Opener
   - Protection Against the Issuer
   - Protection Against Active Adversaries
4. Conclusion

Anonymous encryption

- classical security requirement of an encryption scheme: **privacy of the encrypted data**.

- different (additional) security requirement: key privacy (Bellare, Boldyreva, Desai and Pointcheval – 2001)

  ⇝ provides **privacy of the key under which the encryption was performed**.

- anonymous communication: eavesdroppers are prevented from learning the identities of the communicating parties

  ⇝ a very attractive notion from the user's viewpoint.

- **But**, some organizations and governments are concerned about how anonymity can be abused by criminals.

  ⇝ one should be capable of revoking key privacy when illegal behavior is detected.

⇝ Anonymous Traceable Encryption

Anonymous encryption

- classical security requirement of an encryption scheme: **privacy of the encrypted data**.
- different (additional) security requirement: key privacy (Bellare, Boldyreva, Desai and Pointcheval – 2001)

  $\rightsquigarrow$ provides **privacy of the key under which the encryption was performed**.
- anonymous communication: eavesdroppers are prevented from learning the identities of the communicating parties

  $\rightsquigarrow$ a very attractive notion from the user's viewpoint.
- **But**, some organizations and governments are concerned about how anonymity can be abused by criminals.

  $\rightsquigarrow$ one should be capable of revoking key privacy when illegal behavior is detected.

$\rightsquigarrow$ ANONYMOUS TRACEABLE ENCRYPTION

Anonymous encryption

- classical security requirement of an encryption scheme: **privacy of the encrypted data**.
- different (additional) security requirement: key privacy (Bellare, Boldyreva, Desai and Pointcheval – 2001)

  $\rightsquigarrow$ provides **privacy of the key under which the encryption was performed**.
- anonymous communication: eavesdroppers are prevented from learning the identities of the communicating parties

  $\rightsquigarrow$ a very attractive notion from the user's viewpoint.
- **But**, some organizations and governments are concerned about how anonymity can be abused by criminals.

  $\rightsquigarrow$ one should be capable of revoking key privacy when illegal behavior is detected.

$\rightsquigarrow$ Anonymous Traceable Encryption

Anonymous encryption

- classical security requirement of an encryption scheme: **privacy of the encrypted data**.
- different (additional) security requirement: key privacy (Bellare, Boldyreva, Desai and Pointcheval – 2001)

  $\leadsto$ provides **privacy of the key under which the encryption was performed**.
- anonymous communication: eavesdroppers are prevented from learning the identities of the communicating parties

  $\leadsto$ a very attractive notion from the user's viewpoint.
- **But**, some organizations and governments are concerned about how anonymity can be abused by criminals.

  $\leadsto$ one should be capable of revoking key privacy when illegal behavior is detected.

$\leadsto$ ANONYMOUS TRACEABLE ENCRYPTION

Anonymous Traceable Encryption

- **anonymous traceable encryption**:
  - an adversary cannot determine which user's public key has been used to generate a given ciphertext
  - a trusted third party (given some trapdoor information) is able to revoke anonymity of ciphertexts.
- several works try to achieve this property, e.g.
  - custodian-hiding (verifiable) encryption (Liu, Wei and Wung – 2004)
  - group encryption (Kiayia, Tsiounis and Yung – 2007)
  - group decryption (Qin, Wu, Susilo, Wang – 2007)
- **But** they all miss a critical point: an encryption scheme produces a covert channel

Anonymous Traceable Encryption

- **anonymous traceable encryption**:
  - an adversary cannot determine which user's public key has been used to generate a given ciphertext
  - a trusted third party (given some trapdoor information) is able to revoke anonymity of ciphertexts.
- several works try to achieve this property, e.g.
  - custodian-hiding (verifiable) encryption (Liu, Wei and Wung – 2004)
  - group encryption (Kiayia, Tsiounis and Yung – 2007)
  - group decryption (Qin, Wu, Susilo, Wang – 2007)
- **But** they all miss a critical point: an encryption scheme produces a covert channel

Anonymous Traceable Encryption

- **anonymous traceable encryption**:
  - an adversary cannot determine which user's public key has been used to generate a given ciphertext
  - a trusted third party (given some trapdoor information) is able to revoke anonymity of ciphertexts.
- several works try to achieve this property, e.g.
  - custodian-hiding (verifiable) encryption (Liu, Wei and Wung – 2004)
  - group encryption (Kiayia, Tsiounis and Yung – 2007)
  - group decryption (Qin, Wu, Susilo, Wang – 2007)
- **But** they all miss a critical point: an encryption scheme produces a covert channel

Covert channels

- **covert channel:** a communication channel that exists, contrary to system design.

  aka subliminal channel, steganographic channel.

- to provide semantic security, asymmetric encryption has to be randomized $\leadsto$ covert channel

- malicious users may use the **covert channel** to communicate illegally using ciphertexts that trace back to some honest user.

- More precisely:
  - possible to encrypt a message to a user (the official target recipient for the tracing authority)
  - so that the randomness is used for transmitting information to a third (not traced as a possible recipient).

Covert channels

- **covert channel:** a communication channel that exists, contrary to system design.

  aka subliminal channel, steganographic channel.

- to provide semantic security, asymmetric encryption has to be randomized $\rightsquigarrow$ covert channel

- malicious users may use the **covert channel** to communicate illegally using ciphertexts that trace back to some honest user.

- More precisely:
  - possible to encrypt a message to a user (the official target recipient for the tracing authority)
  - so that the randomness is used for transmitting information to a third (not traced as a possible recipient).

Motivating Example 1: Auction Protocols

- in 2000, Sako proposed a novel approach to achieve bid secrecy in auction protocols.
- consists in expressing each bid as an encryption of a *known* message, with a key corresponding to the value of the bid.
- What needs to be hidden in the ciphertext is **not the message**, but **the key** used to encrypt it;
  $\rightsquigarrow$ anonymous traceable encryption seems very promising for such applications.
- **But**, one major concern in auction protocols is the problem of collusion between bidders
  $\rightsquigarrow$ it is highly desirable to prevent bidders from engaging in such collaborative bidding strategies.
- Unfortunately, no known construction of anonymous traceable encryption is free of covert channels

Motivating Example 1: Auction Protocols

- in 2000, Sako proposed a novel approach to achieve bid secrecy in auction protocols.
- consists in expressing each bid as an encryption of a *known* message, with a key corresponding to the value of the bid.
- What needs to be hidden in the ciphertext is **not the message**, but **the key** used to encrypt it;
  $\rightsquigarrow$ anonymous traceable encryption seems very promising for such applications.
- **But**, one major concern in auction protocols is the problem of collusion between bidders
  $\rightsquigarrow$ it is highly desirable to prevent bidders from engaging in such collaborative bidding strategies.
- Unfortunately, no known construction of anonymous traceable encryption is free of covert channels

Motivating Example 2: Voting protocols

- Kleptography is a way of breaking security of cryptographic systems.

  ⇝ works on code-level: a malicious implementation exploits randomness used in the protocol to build a **trapdoor**

- Randomness in e-voting makes room for a subliminal channel through which may leak:
  - voters' choices;
  - signing key of voting machines;
  - ...

- **Kleptographic attacks** have been presented on several voting protocols:
  - Benaloh-Tuinstra protocol (Borzęcki, Kabarowski, Kubiak, Kutylowski, Zagòrski – 2006))
  - a Mix-net used as a building block of a *Prêt à Voter* e-voting protocol (Kubiak, Kutylowski, Zagòrski – 2007).

- necessity of a subliminal-channel free encryption scheme.

Motivating Example 2: Voting protocols

- Kleptography is a way of breaking security of cryptographic systems.

  ⇝ works on code-level: a malicious implementation exploits randomness used in the protocol to build a **trapdoor**

- Randomness in e-voting makes room for a subliminal channel through which may leak:
  - voters' choices;
  - signing key of voting machines;
  - . . .

- **Kleptographic attacks** have been presented on several voting protocols:
  - Benaloh-Tuinstra protocol (Borzęcki, Kabarowski, Kubiak, Kutylowski, Zagòrski – 2006))
  - a Mix-net used as a building block of a *Prêt à Voter* e-voting protocol (Kubiak, Kutylowski, Zagòrski – 2007).

- necessity of a subliminal-channel free encryption scheme.

Motivating Example 2: Voting protocols

- Kleptography is a way of breaking security of cryptographic systems.

  $\rightsquigarrow$ works on code-level: a malicious implementation exploits randomness used in the protocol to build a **trapdoor**

- Randomness in e-voting makes room for a subliminal channel through which may leak:
  - voters' choices;
  - signing key of voting machines;
  - ...

- **Kleptographic attacks** have been presented on several voting protocols:
  - Benaloh-Tuinstra protocol
    (Borzęcki, Kabarowski, Kubiak, Kutylowski, Zagòrski – 2006))
  - a Mix-net used as a building block of a *Prêt à Voter* e-voting protocol
    (Kubiak, Kutylowski, Zagòrski – 2007).

- necessity of a subliminal-channel free encryption scheme.

Motivating Example 2: Voting protocols

- Kleptography is a way of breaking security of cryptographic systems.

  ⇝ works on code-level: a malicious implementation exploits randomness used in the protocol to build a **trapdoor**

- Randomness in e-voting makes room for a subliminal channel through which may leak:
  - voters' choices;
  - signing key of voting machines;
  - . . .

- **Kleptographic attacks** have been presented on several voting protocols:
  - Benaloh-Tuinstra protocol
    (Borzęcki, Kabarowski, Kubiak, Kutylowski, Zagòrski – 2006))
  - a Mix-net used as a building block of a *Prêt à Voter* e-voting protocol
    (Kubiak, Kutylowski, Zagòrski – 2007).

- necessity of a subliminal-channel free encryption scheme.

Mediated Anonymous Traceable Encryption (MATE)

- introduction of a new primitive: mediated anonymous traceable encryption
- provides **confidentiality** and **anonymity**
- anonymity can be **revoked**
- prevents malicious users to embed **subliminal messages** in ciphertexts

- **Recall:** for semantic security, encryption has to be randomized
  $\rightsquigarrow$ hard to eliminate covert channels
- even impossible without assuming that
  - recipients are securely initialized by a trusted party
  - ciphertexts are modified on-line

- removing the entropy in ciphertexts is impossible
  $\rightsquigarrow$ add more randomness so that any hidden message is smothered.

Mediated Anonymous Traceable Encryption (MATE)

- introduction of a new primitive: mediated anonymous traceable encryption
- provides **confidentiality** and **anonymity**
- anonymity can be **revoked**
- prevents malicious users to embed **subliminal messages** in ciphertexts

- **Recall:** for semantic security, encryption has to be randomized
  $\rightsquigarrow$ hard to eliminate covert channels
- even impossible without assuming that
  - recipients are securely initialized by a trusted party
  - ciphertexts are modified on-line

- removing the entropy in ciphertexts is impossible
  $\rightsquigarrow$ add more randomness so that any hidden message is smothered.

Mediated Anonymous Traceable Encryption (MATE)

- introduction of a new primitive: mediated anonymous traceable encryption
- provides **confidentiality** and **anonymity**
- anonymity can be **revoked**
- prevents malicious users to embed **subliminal messages** in ciphertexts

- **Recall:** for semantic security, encryption has to be randomized
  $\rightsquigarrow$ hard to eliminate covert channels
- even impossible without assuming that
  - recipients are securely initialized by a trusted party
  - ciphertexts are modified on-line
- removing the entropy in ciphertexts is impossible
  $\rightsquigarrow$ add more randomness so that any hidden message is smothered.

Mediated Anonymous Traceable Encryption (MATE)

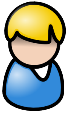
- introduction of a new primitive: mediated anonymous traceable encryption
- provides **confidentiality** and **anonymity**
- anonymity can be **revoked**
- prevents malicious users to embed **subliminal messages** in ciphertexts

- **Recall:** for semantic security, encryption has to be randomized
  $\rightsquigarrow$ hard to eliminate covert channels
- even impossible without assuming that
  - recipients are securely initialized by a trusted party
  - ciphertexts are modified on-line
- removing the entropy in ciphertexts is impossible
  $\rightsquigarrow$ add more randomness so that any hidden message is smothered.

MATE: Cast of characters

(Honest) Users

Alice



Bob

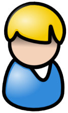
Existence of a PKI

All participants have a certified pair of secret and public keys $(\mathsf{sk}, \mathsf{pk})$.

MATE: Cast of characters

(Honest) Users

Alice



Bob

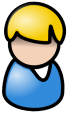
Issuer

Issuer: for adding new members to the system

MATE: Cast of characters

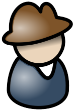
(Honest) Users

Alice



Bob

Issuer



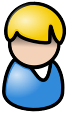
Opener

Opener: for revoking anonymity

MATE: Cast of characters

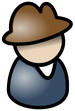
(Honest) Users

Alice



Bob

Issuer



Opener

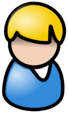
Mediator

Mediator (honest but possibly curious): systematically re-randomizes all its inputs

MATE: Cast of characters

(Honest) Users

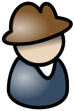
Alice



Bob

Eve

Issuer



Opener

Mediator

Adversary (Eve): confidentiality, anonymity, covert channels

Syntactic definition

Mediated Anonymous Traceable Encryption Schemes

(GSetup, Join, Encrypt, ReRand, Decrypt, Trace, Judge)

- $\mathsf{GSetup}(\lambda) \to (\mathsf{mpk}, \mathsf{msk}, \mathsf{sk}_O, \mathcal{L})$:
    - a group public key mpk,
    - a manager's secret key msk
    - an opening key sk_O;
    - a data structure $\mathcal{L}$ called a *registration list*.
- Join is a polynomial time interactive protocol between a member owning a pair of keys $(\mathsf{sk}, \mathsf{pk})$ and the issuer:
    - $\mathsf{Join.Member}(\mathsf{id}, \mathsf{mpk}, \mathsf{sk}) \to (\mathsf{pk}_{\mathsf{id}}, \mathsf{sk}_{\mathsf{id}})$;
    - $\mathsf{Join.Group}(\mathsf{id}, \mathsf{pk}, \mathsf{msk}) \to \mathcal{L}$.
- $\mathsf{Encrypt}(\mathsf{mpk}, \mathsf{pk}_{\mathsf{id}}, m) \to C$.
- $\mathsf{ReRand}(\mathsf{mpk}, C) \to C'$.
- $\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}}, C) \to m$.
- $\mathsf{Trace}(\mathsf{mpk}, \mathcal{L}, \mathsf{sk}_O, C) \to (\mathsf{id}, \Pi)$.
- $\mathsf{Judge}(\mathsf{mpk}, \mathcal{L}, C, \mathsf{id}, \Pi) \to \{0, 1\}$.

Syntactic definition

Mediated Anonymous Traceable Encryption Schemes

(GSetup, Join, Encrypt, ReRand, Decrypt, Trace, Judge)

- $\mathsf{GSetup}(\lambda) \rightarrow (\mathsf{mpk}, \mathsf{msk}, \mathsf{sk}_O, \mathcal{L})$:
    - a group public key mpk,
    - a manager's secret key msk
    - an opening key sk_O;
    - a data structure $\mathcal{L}$ called a *registration list*.
- Join is a polynomial time interactive protocol between a member owning a pair of keys (sk, pk) and the issuer:
    - $\mathsf{Join.Member}(\mathsf{id}, \mathsf{mpk}, \mathsf{sk}) \rightarrow (\mathsf{pk}_{\mathsf{id}}, \mathsf{sk}_{\mathsf{id}})$;
    - $\mathsf{Join.Group}(\mathsf{id}, \mathsf{pk}, \mathsf{msk}) \rightarrow \mathcal{L}$.
- $\mathsf{Encrypt}(\mathsf{mpk}, \mathsf{pk}_{\mathsf{id}}, m) \rightarrow C$.
- $\mathsf{ReRand}(\mathsf{mpk}, C) \rightarrow C'$.
- $\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}}, C) \rightarrow m$.
- $\mathsf{Trace}(\mathsf{mpk}, \mathcal{L}, \mathsf{sk}_O, C) \rightarrow (\mathsf{id}, \Pi)$.
- $\mathsf{Judge}(\mathsf{mpk}, \mathcal{L}, C, \mathsf{id}, \Pi) \rightarrow \{0, 1\}$.

Syntactic definition

Mediated Anonymous Traceable Encryption Schemes
(GSetup, Join, Encrypt, ReRand, Decrypt, Trace, Judge)

- $\mathsf{GSetup}(\lambda) \rightarrow (\mathsf{mpk}, \mathsf{msk}, \mathsf{sk}_O, \mathcal{L})$:
  - a group public key mpk,
  - a manager's secret key msk
  - an opening key sk_O;
  - a data structure $\mathcal{L}$ called a *registration list*.
- Join is a polynomial time interactive protocol between a member owning a pair of keys (sk, pk) and the issuer:
  - $\mathsf{Join.Member}(\mathsf{id}, \mathsf{mpk}, \mathsf{sk}) \rightarrow (\mathsf{pk}_{\mathsf{id}}, \mathsf{sk}_{\mathsf{id}})$;
  - $\mathsf{Join.Group}(\mathsf{id}, \mathsf{pk}, \mathsf{msk}) \rightarrow \mathcal{L}$.
- $\mathsf{Encrypt}(\mathsf{mpk}, \mathsf{pk}_{\mathsf{id}}, m) \rightarrow C$.
- $\mathsf{ReRand}(\mathsf{mpk}, C) \rightarrow C'$.
- $\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}}, C) \rightarrow m$.
- $\mathsf{Trace}(\mathsf{mpk}, \mathcal{L}, \mathsf{sk}_O, C) \rightarrow (\mathsf{id}, \Pi)$.
- $\mathsf{Judge}(\mathsf{mpk}, \mathcal{L}, C, \mathsf{id}, \Pi) \rightarrow \{0, 1\}$.

Syntactic definition

Mediated Anonymous Traceable Encryption Schemes

(GSetup, Join, Encrypt, ReRand, Decrypt, Trace, Judge)

- $\mathsf{GSetup}(\lambda) \to (\mathsf{mpk}, \mathsf{msk}, \mathsf{sk}_O, \mathcal{L})$:
  - a group public key mpk,
  - a manager's secret key msk
  - an opening key sk_O;
  - a data structure $\mathcal{L}$ called a *registration list*.
- Join is a polynomial time interactive protocol between a member owning a pair of keys $(\mathsf{sk}, \mathsf{pk})$ and the issuer:
  - $\mathsf{Join.Member}(\mathsf{id}, \mathsf{mpk}, \mathsf{sk}) \to (\mathsf{pk}_{\mathsf{id}}, \mathsf{sk}_{\mathsf{id}})$;
  - $\mathsf{Join.Group}(\mathsf{id}, \mathsf{pk}, \mathsf{msk}) \to \mathcal{L}$.
- $\mathsf{Encrypt}(\mathsf{mpk}, \mathsf{pk}_{\mathsf{id}}, m) \to C$.
- $\mathsf{ReRand}(\mathsf{mpk}, C) \to C'$.
- $\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}}, C) \to m$.
- $\mathsf{Trace}(\mathsf{mpk}, \mathcal{L}, \mathsf{sk}_O, C) \to (\mathsf{id}, \Pi)$.
- $\mathsf{Judge}(\mathsf{mpk}, \mathcal{L}, C, \mathsf{id}, \Pi) \to \{0, 1\}$.

Syntactic definition

Mediated Anonymous Traceable Encryption Schemes
(GSetup, Join, Encrypt, ReRand, Decrypt, Trace, Judge)

- $\mathsf{GSetup}(\lambda) \rightarrow (\mathsf{mpk}, \mathsf{msk}, \mathsf{sk}_O, \mathcal{L})$:
  - a group public key mpk,
  - a manager's secret key msk
  - an opening key sk_O;
  - a data structure $\mathcal{L}$ called a *registration list*.
- Join is a polynomial time interactive protocol between a member owning a pair of keys $(\mathsf{sk}, \mathsf{pk})$ and the issuer:
  - $\mathsf{Join.Member}(\mathsf{id}, \mathsf{mpk}, \mathsf{sk}) \rightarrow (\mathsf{pk}_{\mathsf{id}}, \mathsf{sk}_{\mathsf{id}})$;
  - $\mathsf{Join.Group}(\mathsf{id}, \mathsf{pk}, \mathsf{msk}) \rightarrow \mathcal{L}$.
- $\mathsf{Encrypt}(\mathsf{mpk}, \mathsf{pk}_{\mathsf{id}}, m) \rightarrow C$.
- $\mathsf{ReRand}(\mathsf{mpk}, C) \rightarrow C'$.
- $\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}}, C) \rightarrow m$.
- $\mathsf{Trace}(\mathsf{mpk}, \mathcal{L}, \mathsf{sk}_O, C) \rightarrow (\mathsf{id}, \Pi)$.
- $\mathsf{Judge}(\mathsf{mpk}, \mathcal{L}, C, \mathsf{id}, \Pi) \rightarrow \{0, 1\}$.

Syntactic definition

Mediated Anonymous Traceable Encryption Schemes

(GSetup, Join, Encrypt, ReRand, Decrypt, Trace, Judge)

- $\mathsf{GSetup}(\lambda) \rightarrow (\mathsf{mpk}, \mathsf{msk}, \mathsf{sk}_O, \mathcal{L})$:
  - a group public key mpk,
  - a manager's secret key msk
  - an opening key sk_O;
  - a data structure $\mathcal{L}$ called a *registration list*.
- Join is a polynomial time interactive protocol between a member owning a pair of keys $(\mathsf{sk}, \mathsf{pk})$ and the issuer:
  - $\mathsf{Join.Member}(\mathsf{id}, \mathsf{mpk}, \mathsf{sk}) \rightarrow (\mathsf{pk}_{\mathsf{id}}, \mathsf{sk}_{\mathsf{id}})$;
  - $\mathsf{Join.Group}(\mathsf{id}, \mathsf{pk}, \mathsf{msk}) \rightarrow \mathcal{L}$.
- $\mathsf{Encrypt}(\mathsf{mpk}, \mathsf{pk}_{\mathsf{id}}, m) \rightarrow C$.
- $\mathsf{ReRand}(\mathsf{mpk}, C) \rightarrow C'$.
- $\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}}, C) \rightarrow m$.
- $\mathsf{Trace}(\mathsf{mpk}, \mathcal{L}, \mathsf{sk}_O, C) \rightarrow (\mathsf{id}, \Pi)$.
- $\mathsf{Judge}(\mathsf{mpk}, \mathcal{L}, C, \mathsf{id}, \Pi) \rightarrow \{0, 1\}$.

Security Notions

- after a Join protocol execution:
  - the public key of the user, and his identity, are stored/published in the registration list $\mathcal{L}$,
  - the secret key is kept private.
- in our security model, we will exclude collusions of traitors: adversary will be given access to its own private key only, and possibly several public keys.

  $\rightsquigarrow$ in practice: private keys are stored in tamper-proof devices
- we will consider **Chosen-Plaintext Attacks** only (no decryption oracle is available to the adversary).
- Three different notions:
  - **Semantic Security**
  - **Anonymity** (Key privacy)
  - **Subliminal-Channel Freeness**

Security Notions

- after a Join protocol execution:
  - the public key of the user, and his identity, are stored/published in the registration list $\mathcal{L}$,
  - the secret key is kept private.
- in our security model, we will exclude collusions of traitors: adversary will be given access to its own private key only, and possibly several public keys.

  $\rightsquigarrow$ in practice: private keys are stored in tamper-proof devices
- we will consider **Chosen-Plaintext Attacks** only (no decryption oracle is available to the adversary).
- Three different notions:
  - **Semantic Security**
  - **Anonymity** (Key privacy)
  - **Subliminal-Channel Freeness**

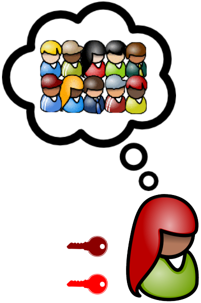
Security Notions

- after a Join protocol execution:
  - the public key of the user, and his identity, are stored/published in the registration list $\mathcal{L}$,
  - the secret key is kept private.
- in our security model, we will exclude collusions of traitors: adversary will be given access to its own private key only, and possibly several public keys.

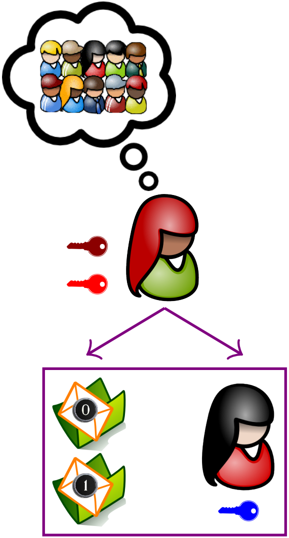
  $\rightsquigarrow$ in practice: private keys are stored in tamper-proof devices
- we will consider **Chosen-Plaintext Attacks** only (no decryption oracle is available to the adversary).
- Three different notions:
  - **Semantic Security**
  - **Anonymity** (Key privacy)
  - **Subliminal-Channel Freeness**

Semantic Security

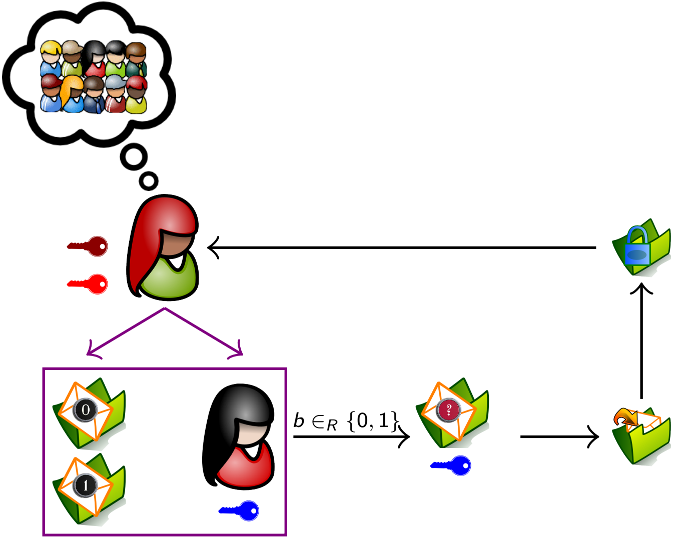
Semantic Security



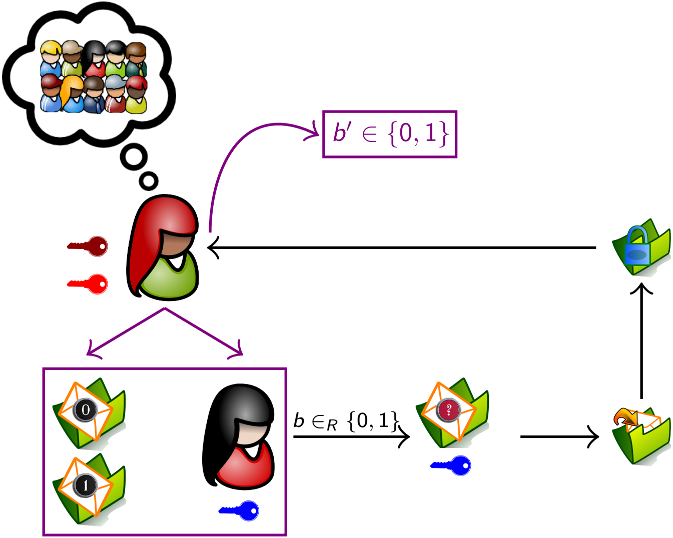
Semantic Security



Semantic Security

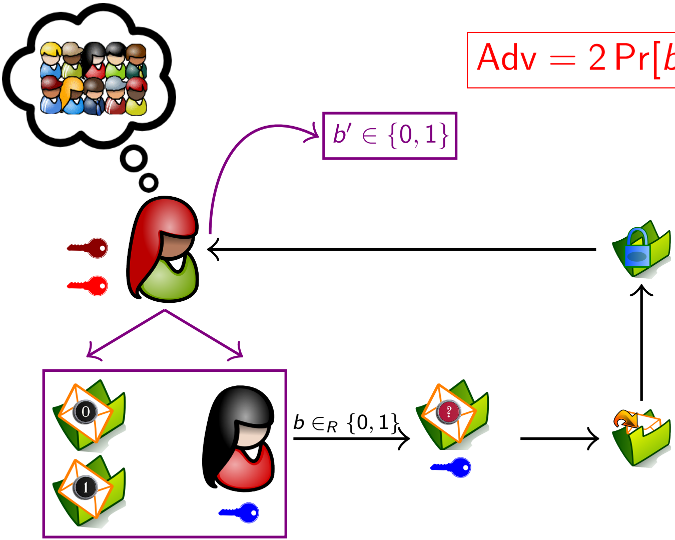


Semantic Security

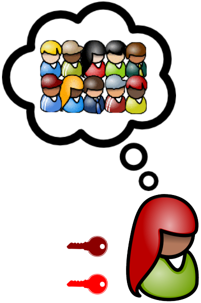


Semantic Security

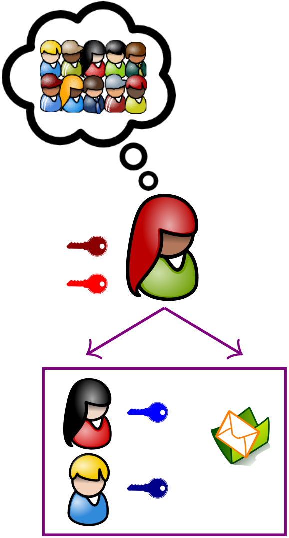
$$\boxed{\mathsf{Adv} = 2\Pr[b' = b] - 1}$$



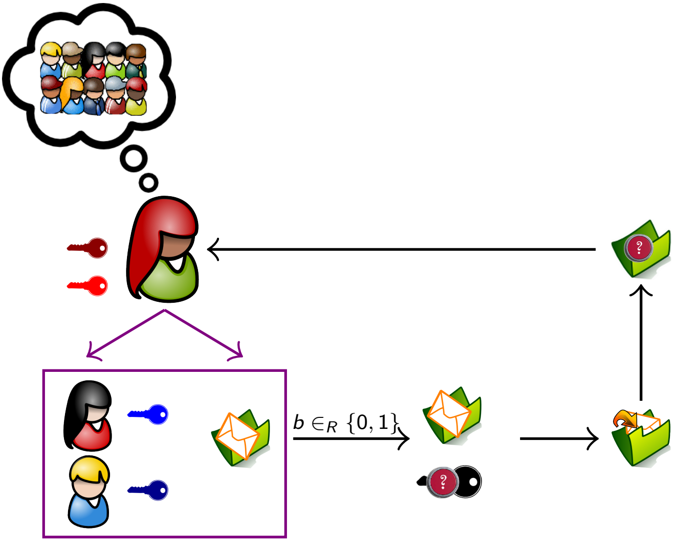
Anonymity



Anonymity

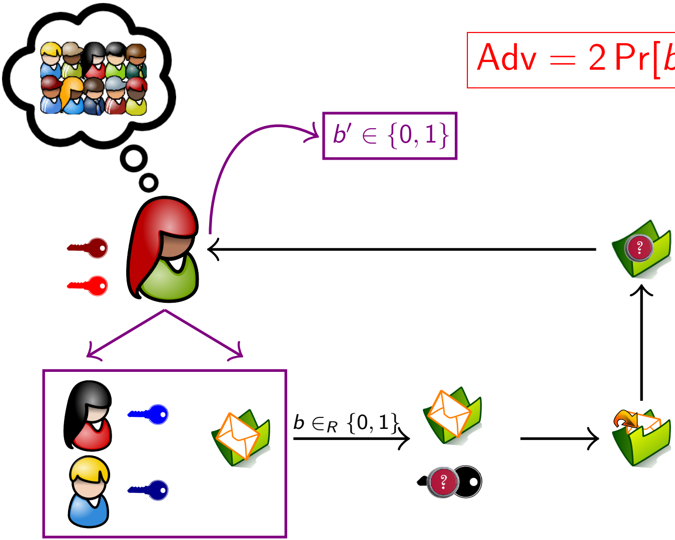


Anonymity



Anonymity

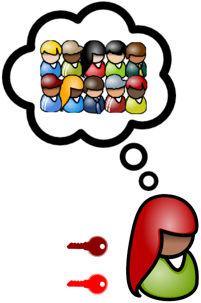
$$\boxed{\mathsf{Adv} = 2\Pr[b' = b] - 1}$$



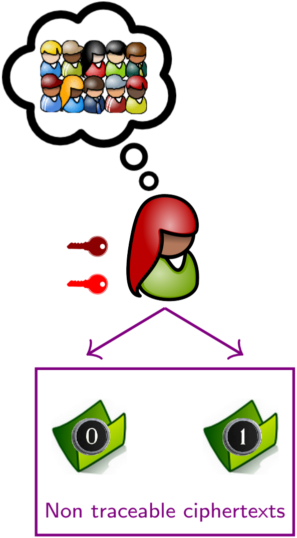
$b' \in \{0,1\}$

$b \in_R \{0,1\}$

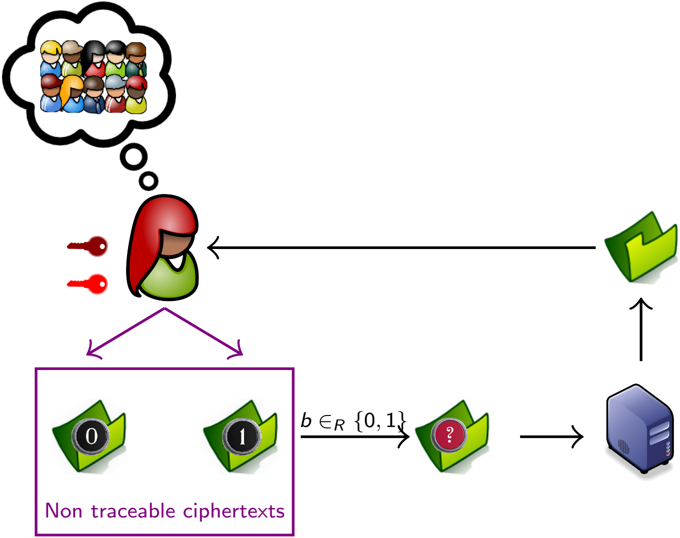
Subliminal Channel Freeness



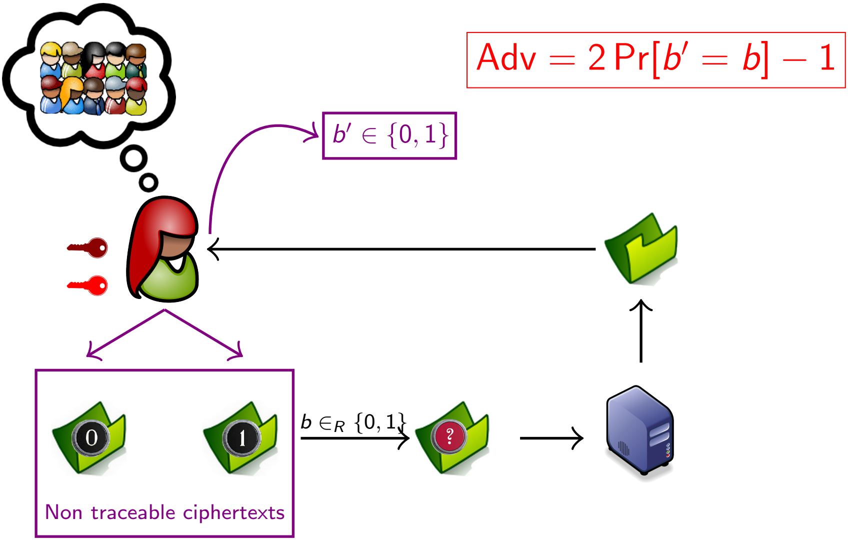
Subliminal Channel Freeness



Subliminal Channel Freeness



Subliminal Channel Freeness



$$\boxed{\mathrm{Adv} = 2\Pr[b' = b] - 1}$$

$b' \in \{0,1\}$

$b \in_R \{0,1\}$

Non traceable ciphertexts

Contents
Traceable Anonymous Encryption

1. Introduction
   - Motivation
   - Model
2. Basic scheme
   - Description
   - Security
3. Extensions and open problems
   - Protection against the Opener
   - Protection Against the Issuer
   - Protection Against Active Adversaries
4. Conclusion

Design principle

- any encryption scheme allowing users to select a private/public key pair where the public key is **unregistered** contains a covert channel:
    - Eve can select such a key pair
    - encrypt a message of her choice under this public key
    - the ciphertext does not trace back to any registered user (but can be decrypted by Eve).

$\rightsquigarrow$ **Idea:** use an encryption scheme with key escrow

- It is easy to design an anonymous encryption scheme that provides traceability or the absence of steganographic channel:
    - in 2007, Borisov and Minami proposed a single-bit re-encryption scheme based on the well-known Goldwasser-Micali scheme
    - it allows the encryption of a single bit to be transformed into another, supposedly completely unlinkable with the original but preserving the bit value.
    - Unfortunately, their scheme is **not anonymous**.

$\rightsquigarrow$ **Idea:** use a universal re-encryption scheme

Design principle

- any encryption scheme allowing users to select a private/public key pair where the public key is **unregistered** contains a covert channel:
  - Eve can select such a key pair
  - encrypt a message of her choice under this public key
  - the ciphertext does not trace back to any registered user (but can be decrypted by Eve).

$\rightsquigarrow$ **Idea:** use an encryption scheme with key escrow

- It is easy to design an anonymous encryption scheme that provides traceability or the absence of steganographic channel:
  - in 2007, Borisov and Minami proposed a single-bit re-encryption scheme based on the well-known Goldwasser-Micali scheme
  - it allows the encryption of a single bit to be transformed into another, supposedly completely unlinkable with the original but preserving the bit value.
  - Unfortunately, their scheme is **not anonymous**.

$\rightsquigarrow$ **Idea:** use a universal re-encryption scheme

Universal Re-encryption

- Re-encrypts the ciphertext without the knowledge of the public key using a random encryption factor.
- Allows external anonymity which provides total privacy protection for data being transmitted.

- Re-encryption is based on a **homomorphic property**.
- Encrypts under the public key and random encryption factor.
- Appends an **identity element** to the ciphertext.
- First decrypts the identity element to confirm the intended message.

Universal Re-encryption

- Re-encrypts the ciphertext without the knowledge of the public key using a random encryption factor.
- Allows external anonymity which provides total privacy protection for data being transmitted.

- Re-encryption is based on a **homomorphic property**.
- Encrypts under the public key and random encryption factor.
- Appends an **identity element** to the ciphertext.
- First decrypts the identity element to confirm the intended message.

ElGamal universal re-encryption

ElGamal re-encryption

- given a cyclic group $\mathbb{G} = \langle g \rangle$ of prime order p. $(\bar{\mathbb{G}} = \mathbb{G} \backslash \{1\})$
- $(\mathsf{pk} = g^x, \mathsf{sk} = x)$
- Encrypt : $m \in \mathbb{G} \longrightarrow C = (c_1, c_2) = (m \cdot y^r, g^r)$
- ReRand : $(c_1, c_2) \in \mathbb{G}^2 \longrightarrow (c_1 \cdot y^s, c_2 \cdot g^s)$

ElGamal universal re-encryption

(Golle, Jakobsson, Juels and Syverson – 2004)

- $(\mathsf{pk} = g^x, \mathsf{sk} = x)$
- Encrypt : $m \in \mathbb{G} \longrightarrow C = (c_1, c_2, c_3, c_4) = (m \cdot y^r, g^r, y^t, g^t)$
  $= (\mathrm{Encrypt}(m), \mathrm{Encrypt}(1_{\mathbb{G}}))$
- ReRand : $(c_1, c_2, c_3, c_4) \in \mathbb{G}^4 \longrightarrow (c_1 \cdot c_3^{s_1}, c_2 \cdot c_4^{s_1}, c_3^{s_2}, c_4^{s_2})$

ElGamal universal re-encryption

ElGamal re-encryption

- given a cyclic group $\mathbb{G} = \langle g \rangle$ of prime order p. $(\bar{\mathbb{G}} = \mathbb{G} \backslash \{1\})$
- $(\mathsf{pk} = g^x, \mathsf{sk} = x)$
- Encrypt : $m \in \mathbb{G} \longrightarrow C = (c_1, c_2) = (m \cdot y^r, g^r)$
- ReRand : $(c_1, c_2) \in \mathbb{G}^2 \longrightarrow (c_1 \cdot y^s, c_2 \cdot g^s)$

ElGamal universal re-encryption

(Golle, Jakobsson, Juels and Syverson – 2004)

- $(\mathsf{pk} = g^x, \mathsf{sk} = x)$
- Encrypt : $m \in \mathbb{G} \longrightarrow C = (c_1, c_2, c_3, c_4) = (m \cdot y^r, g^r, y^t, g^t)$
  $$= (\text{Encrypt}(m), \text{Encrypt}(1_{\mathbb{G}}))$$
- ReRand : $(c_1, c_2, c_3, c_4) \in \mathbb{G}^4 \longrightarrow (c_1 \cdot c_3^{s_1}, c_2 \cdot c_4^{s_1}, c_3^{s_2}, c_4^{s_2})$

Design principle

- Elgamal universal re-encryption is:
  - semantically secure
  - anonymous
  - not subliminal-channel free ! :(

- nothing prevents Eve to use an unregistered public-key

$\rightsquigarrow$ **Idea:** combine with a 1-resilient Identity-Based Encryption scheme

- use a master public key $\mathrm{mpk} = (\Omega_1^{(0)}, \Omega_1^{(1)}, \ldots, \Omega_\ell^{(0)}, \Omega_\ell^{(1)})$, where $\Omega_i^{(b)} = g^{1/x_i^{(b)}}$
- encode identities into ℓ-bits words: $\mathrm{pk}_{\mathrm{id}} = \mathcal{G}(\mathrm{id}) = h_1 h_2 \ldots h_\ell$
- $\mathrm{sk}_{\mathrm{id}} = (x_1^{(h_1)}, \ldots, x_l^{(h_\ell)})$
- use a ℓ-out-of-ℓ secret sharing for $m \in \mathbb{G}$ and encrypt each share with ElGamal universal re-encryption scheme

Design principle

- Elgamal universal re-encryption is:
  - semantically secure
  - anonymous
  - not subliminal-channel free ! :(

- nothing prevents Eve to use an unregistered public-key

⇝ **Idea:** combine with a 1-resilient Identity-Based Encryption scheme

- use a master public key $\mathsf{mpk} = (\Omega_1^{(0)}, \Omega_1^{(1)}, \ldots, \Omega_\ell^{(0)}, \Omega_\ell^{(1)})$, where $\Omega_i^{(b)} = g^{1/x_i^{(b)}}$
- encode identities into ℓ-bits words: $\mathsf{pk}_{\mathsf{id}} = \mathcal{G}(\mathsf{id}) = h_1 h_2 \ldots h_\ell$
- $\mathsf{sk}_{\mathsf{id}} = (x_1^{(h_1)}, \ldots, x_l^{(h_\ell)})$
- use a ℓ-out-of-ℓ secret sharing for $m \in \mathbb{G}$ and encrypt each share with ElGamal universal re-encryption scheme

Design principle

- Elgamal universal re-encryption is:
  - semantically secure
  - anonymous
  - not subliminal-channel free ! :(

- nothing prevents Eve to use an unregistered public-key

⇝ **Idea:** combine with a 1-resilient Identity-Based Encryption scheme

- use a master public key mpk $= (\Omega_1^{(0)}, \Omega_1^{(1)}, \ldots, \Omega_\ell^{(0)}, \Omega_\ell^{(1)})$, where $\Omega_i^{(b)} = g^{1/x_i^{(b)}}$
- encode identities into ℓ-bits words: $\mathrm{pk}_{\mathrm{id}} = \mathcal{G}(\mathrm{id}) = h_1 h_2 \ldots h_\ell$
- $\mathrm{sk}_{\mathrm{id}} = (x_1^{(h_1)}, \ldots, x_l^{(h_\ell)})$
- use a ℓ-out-of-ℓ secret sharing for $m \in \mathbb{G}$ and encrypt each share with ElGamal universal re-encryption scheme

Design principle

- Elgamal universal re-encryption is:
  - semantically secure
  - anonymous
  - not subliminal-channel free ! :(

- nothing prevents Eve to use an unregistered public-key

$\rightsquigarrow$ **Idea:** combine with a 1-resilient Identity-Based Encryption scheme

- use a master public key $\mathrm{mpk} = (\Omega_1^{(0)}, \Omega_1^{(1)}, \ldots, \Omega_\ell^{(0)}, \Omega_\ell^{(1)})$, where $\Omega_i^{(b)} = g^{1/x_i^{(b)}}$
- encode identities into ℓ-bits words: $\mathrm{pk}_{\mathrm{id}} = \mathcal{G}(\mathrm{id}) = h_1 h_2 \ldots h_\ell$
- $\mathrm{sk}_{\mathrm{id}} = (x_1^{(h_1)}, \ldots, x_l^{(h_\ell)})$
- use a ℓ-out-of-ℓ secret sharing for $m \in \mathbb{G}$ and encrypt each share with ElGamal universal re-encryption scheme

Design principle

- Elgamal universal re-encryption is:
  - semantically secure
  - anonymous
  - not subliminal-channel free ! :(

- nothing prevents Eve to use an unregistered public-key

$\rightsquigarrow$ **Idea:** combine with a 1-resilient Identity-Based Encryption scheme

- use a master public key mpk $= (\Omega_1^{(0)}, \Omega_1^{(1)}, \ldots, \Omega_\ell^{(0)}, \Omega_\ell^{(1)})$, where $\Omega_i^{(b)} = g^{1/x_i^{(b)}}$
- encode identities into ℓ-bits words: $\mathrm{pk}_{\mathrm{id}} = \mathcal{G}(\mathrm{id}) = h_1 h_2 \ldots h_\ell$
- $\mathrm{sk}_{\mathrm{id}} = (x_1^{(h_1)}, \ldots, x_l^{(h_\ell)})$
- use a ℓ-out-of-ℓ secret sharing for $m \in \mathbb{G}$ and encrypt each share with ElGamal universal re-encryption scheme

Description

- GSetup(λ):
  - chooses a cyclic group $\mathbb{G} = \langle g \rangle$ of prime order p. ($\bar{\mathbb{G}} = \mathbb{G} \backslash \{1\}$)
  - picks $x_1^{(0)}, x_1^{(1)}, \ldots, x_\ell^{(0)}, x_\ell^{(1)} \xleftarrow{R} \mathbb{Z}_p$, where ℓ is the bit-length of the public keys, whereas the identities are μ-bit long.
  - chooses a code with minimal distance 2, which encodes μ-bits words (the identities) into ℓ-bits words (the public keys) ($\mathsf{pk}_{\mathsf{id}} = \mathcal{G}(\mathsf{id})$).

  $\mathsf{msk} = \mathsf{sk}_O = (x_1^{(0)}, x_1^{(1)}, \ldots, x_\ell^{(0)}, x_\ell^{(1)})$ and $\mathsf{mpk} = (\Omega_1^{(0)}, \Omega_1^{(1)}, \ldots, \Omega_\ell^{(0)}, \Omega_\ell^{(1)})$, where $\Omega_i^{(b)} = g^{1/x_i^{(b)}}$ for $b \in \{0, 1\}$ and $i \in \{1, \ldots, \ell\}$.

- Join:
  - Join.Member(id, mpk, sk): $\mathsf{id} \rightarrow \mathsf{pk}_{\mathsf{id}} = \mathcal{G}(\mathsf{id}) = h_1 h_2 \ldots h_\ell$, where $h_i \in \{0, 1\}$. $\sigma = \mathsf{Sign}(\mathsf{sk}; \mathsf{pk}_{\mathsf{id}})$
  - Join.Group(id, pk, msk): $\mathsf{sk}_{\mathsf{id}} = (x_1^{(h_1)}, \ldots, x_l^{(h_\ell)})$. $\mathcal{L} \leftarrow \mathcal{L} \cup \{(\mathsf{id}, \mathsf{pk}_{\mathsf{id}}, \sigma)\}$.

Description

- Encrypt$(\mathsf{mpk}, \mathsf{pk}_{\mathsf{id}}, m)$ with $\mathsf{pk}_{\mathsf{id}} = \mathcal{G}(\mathsf{id}) = h_1 \dots h_\ell$:
  1. chooses random elements $K_i, U_i \in \mathbb{G}$, for $i = 1, \dots, \ell$, such that
  $$\prod_{i=1}^{\ell} K_i = m \text{ and } \prod_{i=1}^{\ell} U_i = 1;$$
  2. chooses two sequences of random scalars $t_i, s_i \in \mathbb{Z}_p$ for $i = 1, \dots, \ell$;
  3. computes, for $i = 1, \dots, \ell$,
  $$A_i = g^{t_i} \times K_i, \quad B_i = (\Omega_i{}^{(h_i)})^{t_i}, \quad C_i = g^{s_i} \times U_i, \quad D_i = (\Omega_i^{(h_i)})^{s_i}.$$
  If, for some i, $C_1 = 1$ or $D_i = 1$, then one restarts the encryption process.

Description

- ReRand(mpk, C): with $C = (A_i, B_i, C_i, D_i)_{i=1,\ldots,\ell} \in (\mathbb{G}^2 \times \bar{\mathbb{G}}^2)^\ell$,
  1. choose random elements $V_i, W_i \in \mathbb{G}$, for $i = 1, \ldots, \ell$, such that $\prod_{i=1}^{\ell} V_i = \prod_{i=1}^{\ell} W_i = 1$;
  2. choose four sequences of random scalars $r_i^{(0)}, r_i^{(1)}, u_i^{(0)}, u_i^{(1)} \in \mathbb{Z}_p$, for $i = 1, \ldots, \ell$;
  3. choose two random scalars $r, u \in \mathbb{Z}_p$, and compute, for $i = 1, \ldots, \ell$, and for $b = 0, 1$:

$$\begin{array}{rclrcl} A_i^{(b)} & \leftarrow & A_i \times C_i^r \times g^{r_i^{(b)}} \times W_i, & B_i^{(b)} & \leftarrow & B_i \times D_i^r \times (\Omega_i^{(b)})^{r_i^{(b)}} \\ C_i^{(b)} & \leftarrow & C_i^u \times g^{u_i^{(b)}} \times V_i, & D_i^{(b)} & \leftarrow & D_i^u \times (\Omega_i^{(b)})^{u_i^{(b)}}. \end{array}$$

Description

- Decrypt$(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}}, C)$ with $\mathsf{sk}_{\mathsf{id}} = (X_1, \ldots, X_\ell) = (x_1^{(h_1)}, \ldots, x_\ell^{(h_\ell)})$: compute:

$$\prod_{i=1}^{\ell} A_i^{(h_i)} \times (B_i^{(h_i)})^{-X_i}.$$

- Trace$(\mathsf{msk}, \mathcal{L}, \mathsf{sk}_O, C)$:
  1. enumerates all the public keys $\mathsf{pk}_{\mathsf{id}} = h_1 \ldots h_\ell \in \mathcal{L}$ and checks whether

$$\prod_{i=1}^{\ell} C_i^{(h_i)} = \prod_{i=1}^{\ell} (D_i^{(h_i)})^{x_i^{(h_i)}}.$$

  2. When such a $\mathsf{pk}_{\mathsf{id}}$ is found, Π consists of non-interactive zero-knowledge proof of validity of this equality.

- Judge$(\mathsf{mpk}, \mathcal{L}, C, \mathsf{id}, \Pi)$: Check whether the proof Π is valid.

Semantic Security

The following result of the semantic security is similar to the usual ElGamal-like schemes ones.

Theorem

The scheme is semantically secure against chosen-plaintext attacks under the DDH assumption in $\mathbb{G}$.

$$\mathsf{Adv}^{\mathsf{weak-ind}}_{\mathcal{M}\textsc{ates}}(t) \leq 2 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}}(t'),$$

Definition

The DDH problem in basis g in $\mathbb{G}$, denoted $\mathsf{DDH}_{\mathbb{G}}(g)$, consists, given (g^a, g^b, g^c), in deciding whether $c = ab \mod p$.

Anonymity

Anonymity relies on the DLIN assumption introduced in the context of bilinear cryptography:

Definition

The DLIN problem in basis (g, u, v) in $\mathbb{G}$, denoted $\mathsf{DLIN}_{\mathbb{G}}(g, u, v)$, consists, given (u^a, v^b, g^c), in deciding whether $c = a + b \mod p$.

The DLIN assumption is weaker than the DDH assumption.

Theorem

The scheme is anonymous against chosen-plaintext attacks under the $\mathsf{DLIN}_{\mathbb{G}}$ assumption, if the public keys are taken in a code with minimal distance at least 2:

$$\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{M}\mathrm{ATES}}(t) \leq 8\ell^2 \times \mathsf{Adv}^{\mathsf{dlin}}_{\mathbb{G}}(t).$$

Anonymity: Proof (Sketch)

- Given $\mathcal{A}$ an adversary against the anonymity of our scheme, we construct $\mathcal{B}$, that has access to $\mathcal{A}$ in order to break the $\mathsf{DLIN}_{\mathbb{G}}(g, u, v)$ problem.
- $\mathcal{B}$ simulates the GSetup algorithm by using g as the group generator and

$$\begin{array}{rcl} \Omega_i^{(b)} & = & g^{1/x_i^{(b)}}, \quad \text{for } i = 1, \dots, \ell \text{ and } b = 0, 1 \\ \Omega_\gamma^{(\alpha)} & = & u \\ \Omega_\delta^{(\beta)} & = & v \end{array}$$

- $\mathcal{A}$ asks for public keys and one private key $\mathsf{sk}_{\mathsf{id}}$ for itself. With probability $1/4$, this private key does not need to know u nor v discrete log.
- Then, $\mathcal{A}$ outputs a message m and two public keys in $\mathcal{L}$: pk_0 and pk_1. $\mathcal{B}$ picks a random bit B. Since we use a code of minimal distance 2 for generating the public keys pk_β and $\mathsf{pk}_{\mathsf{id}}$ differ in at least 2 positions.

  With probability $\geq 1/\ell^2$, they differ in position γ and δ and the public key pk_B contains the DLIN basis (u, v):
- $\mathcal{B}$'s simulation is then straightforward (but tedious) computation to embed the DLIN instance in the challenge ciphertext.

Anonymity: Proof (Sketch)

- Given $\mathcal{A}$ an adversary against the anonymity of our scheme, we construct $\mathcal{B}$, that has access to $\mathcal{A}$ in order to break the $\mathsf{DLIN}_{\mathbb{G}}(g, u, v)$ problem.
- $\mathcal{B}$ simulates the GSetup algorithm by using g as the group generator and

$$\begin{array}{rcl} \Omega_i^{(b)} & = & g^{1/x_i^{(b)}}, \quad \text{for } i = 1, \ldots, \ell \text{ and } b = 0, 1 \\ \Omega_\gamma^{(\alpha)} & = & u \\ \Omega_\delta^{(\beta)} & = & v \end{array}$$

- $\mathcal{A}$ asks for public keys and one private key $\mathsf{sk}_{\mathsf{id}}$ for itself. With probability $1/4$, this private key does not need to know u nor v discrete log.
- Then, $\mathcal{A}$ outputs a message m and two public keys in $\mathcal{L}$: pk_0 and pk_1. $\mathcal{B}$ picks a random bit B. Since we use a code of minimal distance 2 for generating the public keys pk_β and $\mathsf{pk}_{\mathsf{id}}$ differ in at least 2 positions.

  With probability $\geq 1/\ell^2$, they differ in position γ and δ and the public key pk_B contains the DLIN basis (u, v):
- $\mathcal{B}$'s simulation is then straightforward (but tedious) computation to embed the DLIN instance in the challenge ciphertext.

Anonymity: Proof (Sketch)

- Given $\mathcal{A}$ an adversary against the anonymity of our scheme, we construct $\mathcal{B}$, that has access to $\mathcal{A}$ in order to break the $\mathsf{DLIN}_{\mathbb{G}}(g, u, v)$ problem.
- $\mathcal{B}$ simulates the GSetup algorithm by using g as the group generator and

$$\begin{array}{rcl} \Omega_i^{(b)} & = & g^{1/x_i^{(b)}}, \quad \text{for } i = 1, \ldots, \ell \text{ and } b = 0, 1 \\ \Omega_\gamma^{(\alpha)} & = & u \\ \Omega_\delta^{(\beta)} & = & v \end{array}$$

- $\mathcal{A}$ asks for public keys and one private key $\mathsf{sk}_{\mathsf{id}}$ for itself. With probability $1/4$, this private key does not need to know u nor v discrete log.
- Then, $\mathcal{A}$ outputs a message m and two public keys in $\mathcal{L}$: pk_0 and pk_1. $\mathcal{B}$ picks a random bit B. Since we use a code of minimal distance 2 for generating the public keys pk_β and $\mathsf{pk}_{\mathsf{id}}$ differ in at least 2 positions.

  With probability $\geq 1/\ell^2$, they differ in position γ and δ and the public key pk_B contains the DLIN basis (u, v):
- $\mathcal{B}$'s simulation is then straightforward (but tedious) computation to embed the DLIN instance in the challenge ciphertext.

Anonymity: Proof (Sketch)

- Given $\mathcal{A}$ an adversary against the anonymity of our scheme, we construct $\mathcal{B}$, that has access to $\mathcal{A}$ in order to break the $\mathsf{DLIN}_{\mathbb{G}}(g, u, v)$ problem.
- $\mathcal{B}$ simulates the GSetup algorithm by using g as the group generator and

$$\begin{array}{rcl} \Omega_i^{(b)} & = & g^{1/x_i^{(b)}}, \quad \text{for } i = 1, \ldots, \ell \text{ and } b = 0, 1 \\ \Omega_\gamma^{(\alpha)} & = & u \\ \Omega_\delta^{(\beta)} & = & v \end{array}$$

- $\mathcal{A}$ asks for public keys and one private key $\mathsf{sk_{id}}$ for itself. With probability $1/4$, this private key does not need to know u nor v discrete log.
- Then, $\mathcal{A}$ outputs a message m and two public keys in $\mathcal{L}$: pk_0 and pk_1. $\mathcal{B}$ picks a random bit B. Since we use a code of minimal distance 2 for generating the public keys pk_β and $\mathsf{pk_{id}}$ differ in at least 2 positions.

  With probability $\geq 1/\ell^2$, they differ in position γ and δ and the public key pk_B contains the DLIN basis (u, v):
- $\mathcal{B}$'s simulation is then straightforward (but tedious) computation to embed the DLIN instance in the challenge ciphertext.

Anonymity: Proof (Sketch)

- Given $\mathcal{A}$ an adversary against the anonymity of our scheme, we construct $\mathcal{B}$, that has access to $\mathcal{A}$ in order to break the $\mathsf{DLIN}_{\mathbb{G}}(g, u, v)$ problem.
- $\mathcal{B}$ simulates the GSetup algorithm by using g as the group generator and

$$\begin{array}{rcl} \Omega_i^{(b)} & = & g^{1/x_i^{(b)}}, \quad \text{for } i = 1, \ldots, \ell \text{ and } b = 0, 1 \\ \Omega_\gamma^{(\alpha)} & = & u \\ \Omega_\delta^{(\beta)} & = & v \end{array}$$

- $\mathcal{A}$ asks for public keys and one private key $\mathsf{sk}_{\mathsf{id}}$ for itself. With probability $1/4$, this private key does not need to know u nor v discrete log.
- Then, $\mathcal{A}$ outputs a message m and two public keys in $\mathcal{L}$: pk_0 and pk_1. $\mathcal{B}$ picks a random bit B. Since we use a code of minimal distance 2 for generating the public keys pk_β and $\mathsf{pk}_{\mathsf{id}}$ differ in at least 2 positions.

  With probability $\geq 1/\ell^2$, they differ in position γ and δ and the public key pk_B contains the DLIN basis (u, v):
- $\mathcal{B}$'s simulation is then straightforward (but tedious) computation to embed the DLIN instance in the challenge ciphertext.

Subliminal Channel Freeness

Theorem

The scheme is subliminal-channel free against chosen-plaintext attacks under the DDH assumption:

$$\mathsf{Adv}^{\mathsf{subF}}_{\mathcal{M}\text{ATES}}(t) \leq 4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}}(t'),$$

where t' exceeds t for a few more exponentiations (14ℓ).

Proof (Sketch):

- We consider an adversary $\mathcal{A}$ that owns a key, and that tries to transfer some information in a ciphertext that does not trace back to this key.
- under the DDH assumption, after re-randomization, any ciphertext that does not trace back to its key is **indistinguishable** to a random ciphertext for $\mathcal{A}$.
- therefore the re-randomization of two ciphertexts that **do not trace back to** $\mathcal{A}$ will lead to indistinguishable ciphertexts. □

Subliminal Channel Freeness

Theorem

The scheme is subliminal-channel free against chosen-plaintext attacks under the DDH assumption:

$$\mathsf{Adv}^{\mathsf{subF}}_{\mathcal{M}\text{ATES}}(t) \leq 4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}}(t'),$$

where t' exceeds t for a few more exponentiations (14ℓ).

Proof (Sketch):

- We consider an adversary $\mathcal{A}$ that owns a key, and that tries to transfer some information in a ciphertext that does not trace back to this key.
- under the DDH assumption, after re-randomization, any ciphertext that does not trace back to its key is **indistinguishable** to a random ciphertext for $\mathcal{A}$.
- therefore the re-randomization of two ciphertexts that **do not trace back to** $\mathcal{A}$ will lead to indistinguishable ciphertexts. □

Subliminal Channel Freeness

Theorem

The scheme is subliminal-channel free against chosen-plaintext attacks under the DDH assumption:

$$\mathsf{Adv}^{\mathsf{subF}}_{\mathcal{M}\text{ATES}}(t) \leq 4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}}(t'),$$

where t' exceeds t for a few more exponentiations (14ℓ).

Proof (Sketch):

- We consider an adversary $\mathcal{A}$ that owns a key, and that tries to transfer some information in a ciphertext that does not trace back to this key.
- under the DDH assumption, after re-randomization, any ciphertext that does not trace back to its key is **indistinguishable** to a random ciphertext for $\mathcal{A}$.
- therefore the re-randomization of two ciphertexts that **do not trace back to** $\mathcal{A}$ will lead to indistinguishable ciphertexts. □

Subliminal Channel Freeness

Theorem

The scheme is subliminal-channel free against chosen-plaintext attacks under the DDH assumption:

$$\mathsf{Adv}^{\mathsf{subF}}_{\mathcal{M}\mathrm{ATES}}(t) \leq 4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}}(t'),$$

where t' exceeds t for a few more exponentiations (14ℓ).

Proof (Sketch):

- We consider an adversary $\mathcal{A}$ that owns a key, and that tries to transfer some information in a ciphertext that does not trace back to this key.
- under the DDH assumption, after re-randomization, any ciphertext that does not trace back to its key is **indistinguishable** to a random ciphertext for $\mathcal{A}$.
- therefore the re-randomization of two ciphertexts that **do not trace back to** $\mathcal{A}$ will lead to indistinguishable ciphertexts. □

Contents
Traceable Anonymous Encryption

1. Introduction
   - Motivation
   - Model
2. Basic scheme
   - Description
   - Security
3. Extensions and open problems
   - Protection against the Opener
   - Protection Against the Issuer
   - Protection Against Active Adversaries
4. Conclusion

Protection against the Opener

- Our scheme only provides “weak” semantic security since the issuing key and the opening key are the same

- separation of the authorities ⇝ embed our scheme into a *bilinear setting*.

- Let $(\mathbb{G} = \langle g \rangle, \mathbb{G}_T, e)$ be a pairing friendly structure, i.e.

$$e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$$

  is a (non-degenerate) bilinear map:

  - this structure permits to give group elements $g^{x_i^{(b)}}$ to the opening authority instead of the scalar exponents $x_i^{(b)}$.
  - encryption of the message is done in the group $\mathbb{G}_T$ instead of the group $\mathbb{G}$ (*i.e.*

$$A_i = e(g,g)^{t_i} \times K_i, B_i = e(g,(\Omega_i)^{(h_i)})^{t_i} \ldots$$

  - but the tracing elements $(C_i, D_i) \in \mathbb{G}$ remains unchanged

- we can then achieve the strong notion of **semantic security** (assuming the hardness of DDH in $\mathbb{G}_T$), and our scheme remains **anonymous** (since the DLIN problem remains difficult in a bilinear setting).

Protection against the Opener

- Our scheme only provides "weak" semantic security since the issuing key and the opening key are the same

- separation of the authorities ⇝ embed our scheme into a *bilinear setting*.

- Let $(\mathbb{G} = \langle g \rangle, \mathbb{G}_T, e)$ be a pairing friendly structure, i.e.

$$e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$$

  is a (non-degenerate) bilinear map:

  - this structure permits to give group elements $g^{x_i^{(b)}}$ to the opening authority instead of the scalar exponents $x_i^{(b)}$.
  - encryption of the message is done in the group $\mathbb{G}_T$ instead of the group $\mathbb{G}$ (*i.e.*

$$A_i = e(g, g)^{t_i} \times K_i, B_i = e(g, (\Omega_i)^{(h_i)})^{t_i} \ldots$$

  - but the tracing elements $(C_i, D_i) \in \mathbb{G}$ remains unchanged

- we can then achieve the strong notion of **semantic security** (assuming the hardness of DDH in $\mathbb{G}_T$), and our scheme remains **anonymous** (since the DLIN problem remains difficult in a bilinear setting).

Protection against the Opener

- Our scheme only provides "weak" semantic security since the issuing key and the opening key are the same
- separation of the authorities $\rightsquigarrow$ embed our scheme into a *bilinear setting*.
- Let $(\mathbb{G} = \langle g \rangle, \mathbb{G}_T, e)$ be a pairing friendly structure, i.e.

$$e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$$

  is a (non-degenerate) bilinear map:

  - this structure permits to give group elements $g^{x_i^{(b)}}$ to the opening authority instead of the scalar exponents $x_i^{(b)}$.
  - encryption of the message is done in the group $\mathbb{G}_T$ instead of the group $\mathbb{G}$ (*i.e.*

$$A_i = e(g,g)^{t_i} \times K_i, B_i = e(g,(\Omega_i)^{(h_i)})^{t_i} \ldots$$

  - but the tracing elements $(C_i, D_i) \in \mathbb{G}$ remains unchanged

- we can then achieve the strong notion of **semantic security** (assuming the hardness of DDH in $\mathbb{G}_T$), and our scheme remains **anonymous** (since the DLIN problem remains difficult in a bilinear setting).

Protection against the Opener

- Our scheme only provides "weak" semantic security since the issuing key and the opening key are the same
- separation of the authorities $\rightsquigarrow$ embed our scheme into a *bilinear setting*.
- Let $(\mathbb{G} = \langle g \rangle, \mathbb{G}_T, e)$ be a pairing friendly structure, i.e.

  $$e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$$

  is a (non-degenerate) bilinear map:
  - this structure permits to give group elements $g^{x_i^{(b)}}$ to the opening authority instead of the scalar exponents $x_i^{(b)}$.
  - encryption of the message is done in the group $\mathbb{G}_T$ instead of the group $\mathbb{G}$ (*i.e.*

    $$A_i = e(g, g)^{t_i} \times K_i, B_i = e(g, (\Omega_i)^{(h_i)})^{t_i} \ldots$$
  - but the tracing elements $(C_i, D_i) \in \mathbb{G}$ remains unchanged
- we can then achieve the strong notion of **semantic security** (assuming the hardness of DDH in $\mathbb{G}_T$), and our scheme remains **anonymous** (since the DLIN problem remains difficult in a bilinear setting).

Protection against the Opener

- Unfortunately, in the subliminal freeness game, the pairing structure combined with the knowledge of one secret key are enough to give the adversary all the control over the simulation.
- But if:
  - we replace the code that encodes identity with a code with minimal distance 3
  - and the tracing procedure now enumerates all the words at distance at most one from the public keys $\mathsf{pk}_{\mathsf{id}} \in \mathcal{L}$

  we can only prove security against generic adversaries.
- we leave as an open problem the design of a two-level trust secure scheme in the standard model.

Protection against the Opener

- Unfortunately, in the subliminal freeness game, the pairing structure combined with the knowledge of one secret key are enough to give the adversary all the control over the simulation.

- But if:
  - we replace the code that encodes identity with a code with minimal distance 3
  - and the tracing procedure now enumerates all the words at distance at most one from the public keys $\mathsf{pk}_{\mathsf{id}} \in \mathcal{L}$

  we can only prove security against generic adversaries.

- we leave as an open problem the design of a two-level trust secure scheme in the standard model.

Protection against the Opener

- Unfortunately, in the subliminal freeness game, the pairing structure combined with the knowledge of one secret key are enough to give the adversary all the control over the simulation.
- But if:
  - we replace the code that encodes identity with a code with minimal distance 3
  - and the tracing procedure now enumerates all the words at distance at most one from the public keys $\mathsf{pk}_{\mathsf{id}} \in \mathcal{L}$

  we can only prove security against generic adversaries.
- we leave as an open problem the design of a two-level trust secure scheme in the standard model.

Protection against the Issuer

- The issuer can be involved in malicious activities (*key escrow problem*).
- to prevent the issuer from breaking the semantic security:
  - first encrypt m with an appropriate encryption scheme in $\mathbb{G}^r$ for $r \in \mathbb{N}$. (e.g. ElGamal: $r = 2$)
  - then to re-encrypt (component by component) the ciphertext with our MATE scheme
- If the global ciphertext is not traceable, then the underlying ciphertext will be totally random, and thus, no information will be transmitted.

Protection against the Issuer

- The issuer can be involved in malicious activities (*key escrow problem*).
- to prevent the issuer from breaking the semantic security:
  - first encrypt m with an appropriate encryption scheme in $\mathbb{G}^r$ for $r \in \mathbb{N}$. (e.g. ElGamal: $r = 2$)
  - then to re-encrypt (component by component) the ciphertext with our MATE scheme
- If the global ciphertext is not traceable, then the underlying ciphertext will be totally random, and thus, no information will be transmitted.

Protection against the Issuer

- The issuer can be involved in malicious activities (*key escrow problem*).
- to prevent the issuer from breaking the semantic security:
  - first encrypt m with an appropriate encryption scheme in $\mathbb{G}^r$ for $r \in \mathbb{N}$. (e.g. ElGamal: $r = 2$)
  - then to re-encrypt (component by component) the ciphertext with our MATE scheme
- If the global ciphertext is not traceable, then the underlying ciphertext will be totally random, and thus, no information will be transmitted.

Protection against Active Adversaries

- The security model could be enhanced to provide an access to a decryption oracle.
  (but under specific conditions, because of the natural malleability of the schemes).
- Unfortunately, our scheme does not achieve security against active adversaries
- It seems highly non-trivial to achieve security against active adversaries and covert freeness.

  We leave as an open problem the design of such a scheme.

Protection against Active Adversaries

- The security model could be enhanced to provide an access to a decryption oracle.
  (but under specific conditions, because of the natural malleability of the schemes).

- Unfortunately, our scheme does not achieve security against active adversaries

- It seems highly non-trivial to achieve security against active adversaries and covert freeness.

  We leave as an open problem the design of such a scheme.

Contents
Traceable Anonymous Encryption

1. Introduction
   - Motivation
   - Model
2. Basic scheme
   - Description
   - Security
3. Extensions and open problems
   - Protection against the Opener
   - Protection Against the Issuer
   - Protection Against Active Adversaries
4. Conclusion

Conclusion

- we examined covert channels in the context of anonymous traceable encryption
- we introduced a new primitive: mediated anonymous traceable encryption
- we gave security definitions for this new primitive and a construction meeting the formalized requirements.
- our construction is fairly efficient, with ciphertexts that have **logarithmic size** in the number of group members
- Its security analysis requires classical complexity assumptions in the **standard model**.

Open problems

- design of a **two-level trust** secure scheme in the standard model.
- design of a scheme secure against **active adversaries** and covert-channel freeness.
- design of a scheme with **constant-size** ciphertexts.

  Unfortunately, the Boneh-Franklin universal re-encryption IBE scheme is not subliminal free
- design a scheme secure against **collusions of traitors** (i.e. in which the adversary is given access to several private keys).

  Conjecture: a variant of our scheme with frameproof codes reaches this strong security notions.

Open problems

- design of a **two-level trust** secure scheme in the standard model.
- design of a scheme secure against **active adversaries** and covert-channel freeness.
- design of a scheme with **constant-size** ciphertexts.

  Unfortunately, the Boneh-Franklin universal re-encryption IBE scheme is not subliminal free
- design a scheme secure against **collusions of traitors** (i.e. in which the adversary is given access to several private keys).

  Conjecture: a variant of our scheme with frameproof codes reaches this strong security notions.

Open problems

- design of a **two-level trust** secure scheme in the standard model.
- design of a scheme secure against **active adversaries** and covert-channel freeness.
- design of a scheme with **constant-size** ciphertexts.

  Unfortunately, the Boneh-Franklin universal re-encryption IBE scheme is not subliminal free
- design a scheme secure against **collusions of traitors** (i.e. in which the adversary is given access to several private keys).

  Conjecture: a variant of our scheme with frameproof codes reaches this strong security notions.

Traceable Anonymous Encryption

Julien Cathalo[1], Malika Izabachène[2], David Pointcheval[2] and Damien Vergnaud[2]

[1] Université Catholique de Louvain, Crypto Group (Belgium)
[2] École Normale Supérieure, C.N.R.S. – I.N.R.I.A. (France)

June 28, 2009
Grenoble