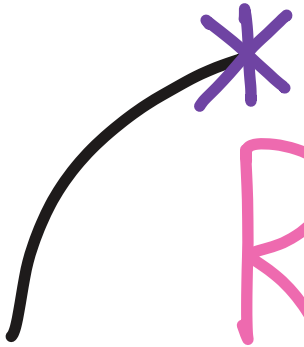


Relational Cryptis

A Separation Logic for Indistinguishability in the Symbolic Model

Chandradeep Dey
Arthur Azevedo de Amorim
Rochester Institute of Technology

A motivating example

Alice  generates a key k

A motivating example

Alice  generates a key k

 gives Eve  oracle access to $Enc_k(\cdot)$

A motivating example

Alice  generates a key k

 gives Eve  oracle access to $Enc_k(\cdot)$

 sends messages M_0, M_1 to 

A motivating example


Alice generates a key k


gives Eve  oracle access to $Enc_k(\cdot)$


sends messages M_0, M_1 to 


tosses a coin (1)

A motivating example

 generates a key k

 gives Eve  oracle access to $Enc_k(\cdot)$

 sends messages M_0, M_1 to 

 tosses a coin (1)

 sends $Enc_k(M_{(1)})$ to 

A motivating example (IND-CPA)

A cryposystem is indistinguishable under chosen-plaintext attack if  cannot determine whether $Enc_k(M_0)$ or $Enc_k(M_1)$ was sent.

The Dolev-Yao Model

- Terms, not bitstrings

```
Inductive term :=
  | TInt of Z
  | TPair of term & term
  | TKey of key-type & term
  | TSeal of term & term
      :
```

The Dolev-Yao Model

- Terms, not bitstrings
- Idealized crypto using rewrite systems

```
adec k (aenc pkey k) m = Some m
```

The Dolev-Yao Model

- Terms, not bitstrings
- Idealized crypto using rewrite systems
- **Attacker with unlimited computational power**
  - + receive, send, duplicate, manipulate messages sent over the network

The Dolev-Yao Model

- Terms, not bitstrings
- Idealized crypto using rewrite systems
- **Attacker with unlimited computational power**
  - + receive, send, duplicate, manipulate messages sent over the network
  - + bound by the rewrite rules

Cryptis

— Separation logic for symbolic cryptography

Arthur Azevedo de Amorim et al. 2025

Cryptis

— Separation logic for symbolic cryptography

— public t −∗ WP send c t {{ Ψ }}

```
Definition public t : iProp :=
  match t with
  | TInt t ⇒ True
  | TKey kt t ⇒ public_key_type kt
  | TSeal (TKey AEnc t) t' ⇒
      □ (public (TKey ADec t) ⇒ public t')
  ⋮
```

Arthur Azevedo de Amorim et al. 2025

Cryptis

— Separation logic for symbolic cryptography

— public t ⟶∗ WP send c t {{ Ψ }}

— The public predicate enforces rewrite rules

⌜adec k t = Some t'⌝ ∗ public k ∗ public t

⟶∗ public t'

Arthur Azevedo de Amorim et al. 2025

Cryptis

- Separation logic for symbolic cryptography
- public t $\twoheadrightarrow$ WP send c t {{ ψ }}
- The public predicate enforces rewrite rules
- Can prove term secrecy, agreement etc.

Arthur Azevedo de Amorim et al. 2025

Cryptis

- Separation logic for symbolic cryptography
- public t −∗ WP send c t {{ ψ }}
- The public predicate enforces rewrite rules
- Can prove term secrecy, agreement etc.
-  IND-CPA ?

Arthur Azevedo de Amorim et al. 2025

Cryptis

- Separation logic for symbolic cryptography
- public t $\twoheadrightarrow$ WP send c t {{ ψ }}
- The public predicate enforces rewrite rules
  - Can prove trace properties
  - Cannot prove hyperproperties

Arthur Azevedo de Amorim et al. 2025

IND-CPA?

— What does IND-CPA even mean in our setting?

IND-CPA?

— What does IND-CPA even mean in our setting?

— Observational equivalence?

ReLoC

— Separation logic on top of Iris for program equivalence.

Dan Frumin et al. 2021

ReLoC

— Separation logic on top of Iris for program equivalence.

— $e \lesssim_{ctx} e' : \tau$

Dan Frumin et al. 2021

ReLoC

— Separation logic on top of Iris for program equivalence.

— $e \lesssim_{ctx} e' : \tau$

— REL $e << e' : \tau \to e \lesssim_{ctx} e' : \tau$

Dan Frumin et al. 2021

ReLoC

- Separation logic on top of Iris for program equivalence.
- $e \lesssim_{ctx} e' : \tau$
- REL $e << e' : \tau \to e \lesssim_{ctx} e' : \tau$
- Attacker knowledge? τ?

Dan Frumin et al. 2021

Composition

Lemma foo : REL $e << e' : \tau$

Lemma foo' : $(\tau \to \psi) \to$
REL $e << e' : \psi$

Attacker knowledge

- Bijection between terms exposed to the network.

Attacker knowledge

- Bijection between terms exposed to the network.
- Must preserve Dolev-Yao properties

Attacker knowledge

- Bijection between terms exposed to the network.
- Must preserve Dolev-Yao properties
- public as a bijection ?

Attacker knowledge

- Bijection between terms exposed to the network.
- Must preserve Dolev-Yao properties
- public as a bijection ?

TSeal (TKey AEnc k) t, TSeal (TKey AEnc k') t'

TSeal (TKey AEnc k) t_1, TSeal (TKey AEnc k_1') t_1'

TKey ADec k, TKey ADec k'

publicly_related

```
PUB⟨t, t'⟩ ≜ match t, t' with
  | TInt n, TInt n' ⇒ ⌜n = n'⌝
  | TPair t1 t2, TPair t1' t2' ⇒ PUB⟨t1, t1'⟩ ∧
                                 PUB⟨t2, t2'⟩
  | TNonce a, TNonce a' ⇒
                  public_rel ↪BIJ □ ⟨t, t'⟩
  .
  .
  .
```

Ghost state

public_rel $\hookrightarrow_{BIJ} \square \langle t, t' \rangle$

— partial bijection

Ghost state

public_rel $\hookrightarrow_{BIJ} \square \langle t, t' \rangle$

- partial bijection
- elements track terms related by fiat

Ghost state 

public_rel $\hookrightarrow_{BIJ} \square \langle t, t' \rangle$

- partial bijection
- elements track terms related by fiat
- publicly_related does the rest by structural recursion

publicly_related

```
PUB⟨t, t'⟩ ≜ match t, t' with
      ⋮
| TSeal k t₁, TSeal k' t₁' ⇒
  (PUB⟨k, k'⟩ ∧ PUB⟨t₁, t₁'⟩) ∨
  (public_rel ↪BIJ □⟨t, t'⟩ ∧
   PRIV⟨k, k'⟩ ∧ PRIV⟨t₁, t₁'⟩ ∧
   □(match k, k' with ...
     | TKey AEnc k₁, TKey AEnc k₁' ⇒
         PUB⟨k₁, k₁'⟩ → PUB⟨t₁, t₁'⟩))
```

privately_related

PRIV ⟨t, t'⟩

"t and t' shall be united once more"

— ¬¬ Sybill Trelawney

privately_related

$$\mathrm{PRIV}\langle t, t' \rangle$$

"t and t' shall be united once more"

— ¬¬ Sybill Trelawney

$$\forall t'', \mathrm{PUB}\langle t, t'' \rangle \longrightarrow \ulcorner t' = t'' \urcorner$$

privately_related

PRIV ⟨t, t'⟩

"t and t' shall be united once more"

— ¬¬ Sybill Trelawney

— decreasing measure : min(tsize t, tsize t')

$\forall t'', \mathrm{PUB}\langle t, t'' \rangle \longrightarrow \ulcorner t' = t'' \urcorner$ 

privately_related

PRIV ⟨t, t'⟩

"t and t' shall be united once more"

— ¬¬ Sybill Trelawney

— decreasing measure : min(tsize t, tsize t')

$\forall t'', \mathrm{PUB}\langle t, t'' \rangle \to \ulcorner t' = t'' \urcorner$ ✗

$\forall t'', \triangleright \mathrm{PUB}\langle t, t'' \rangle \twoheadrightarrow \triangleright \ulcorner t' = t'' \urcorner$

privately_related

PRIV ⟨t, t'⟩

"t and t' shall be united once more"

— ¬¬ Sybill Trelawney

— decreasing measure : min(tsize t, tsize t')

$\forall t'', \mathrm{PUB}\langle t, t'' \rangle \to \ulcorner t' = t'' \urcorner$ ✗

$\forall t'', \triangleright \mathrm{PUB}\langle t, t'' \rangle \twoheadrightarrow \triangleright \ulcorner t' = t'' \urcorner$ ✓

IND-CPA

```
Lemma rel_alice c c' b :
  cryptis_rel_ctx -∗
                  channel_rel c c' -∗
  REL alice c << alice c' :
λ (choice, guess) (choice', guess').
b = choice → b ≠ choice' ∧ guess = guess'
```

IND-CPA

- create sk_A, sk_A'

IND-CPA

- create skA, skA'
- send PUB⟨pkey skA, pkey skA'⟩
  - + relate by fiat, not structure

IND-CPA

- create skA, skA'
- send PUB⟨pkey skA, pkey skA'⟩
- receive PUB⟨M_0, M_0'⟩, PUB⟨M_1, M_1'⟩

IND-CPA

- create skA, skA'
- send PUB⟨pkey skA, pkey skA'⟩
- receive PUB⟨M_0, M_0'⟩, PUB⟨M_1, M_1'⟩
- toss coin b

IND-CPA

- create skA, skA'
- send PUB⟨pkey skA, pkey skA'⟩
- receive PUB⟨M_0, M_0'⟩, PUB⟨M_1, M_1'⟩
- toss coin b
- send PUB⟨aenc (pkey skA) M_b, aenc (pkey skA') $M_{\bar{b}}$⟩

IND-CPA

PUB⟨M_0, M_0'⟩
PUB⟨M_1, M_1'⟩

— send PUB⟨pkey skA, pkey skA'⟩

— send PUB⟨aenc (pkey skA) M_b,
aenc (pkey skA') $M_{\bar{b}}$⟩

+must relate by fiat, not structure

IND-CPA

PUB⟨M_0, M_0'⟩
PUB⟨M_1, M_1'⟩

— send PUB⟨pkey skA, pkey skA'⟩

— send PUB⟨aenc (pkey skA) M_b,
aenc (pkey skA') $M_{\bar{b}}$⟩

+must relate by fiat, not structure

+ need to prove

PRIV⟨pkey skA, pkey skA'⟩

Problem

$$\mathrm{PUB}\langle \mathrm{pkey}\ \mathrm{skA}, \mathrm{pkey}\ \mathrm{skA}' \rangle \longrightarrow *$$

$$\mathrm{PRIV}\langle \mathrm{pkey}\ \mathrm{skA}, \mathrm{pkey}\ \mathrm{skA}' \rangle$$

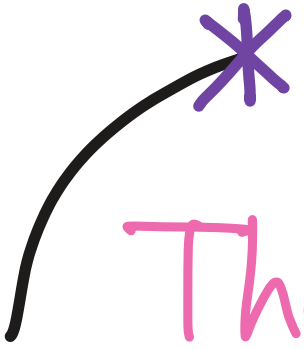
is unprovable!

(expands to a proof of $\triangleright P \twoheadrightarrow P$)

Potential Solutions ?

- a different definition of PRIV ?
- spookier ghost state ?

Reach out if you know 🥺



Thank You! Questions?