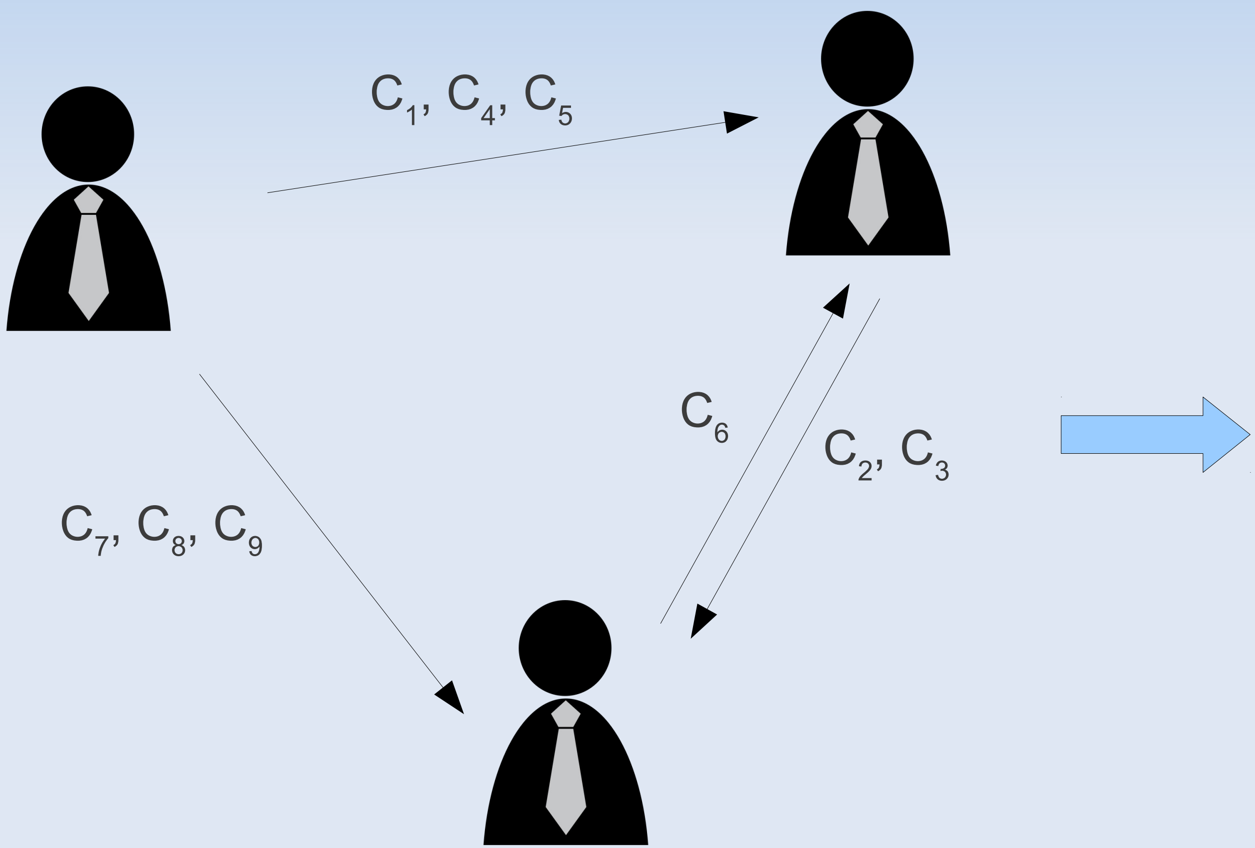
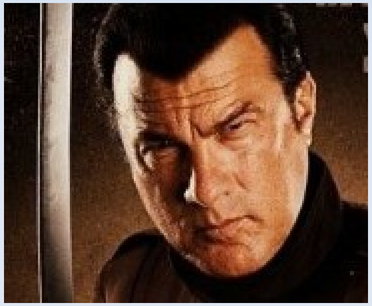
All-But-Many Lossy Trapdoor Functions and Their Applications

Dennis Hofheinz (Karlsruhe Institute of Technology)

Encryption: the "Real World"

- Many parties, many ciphertexts

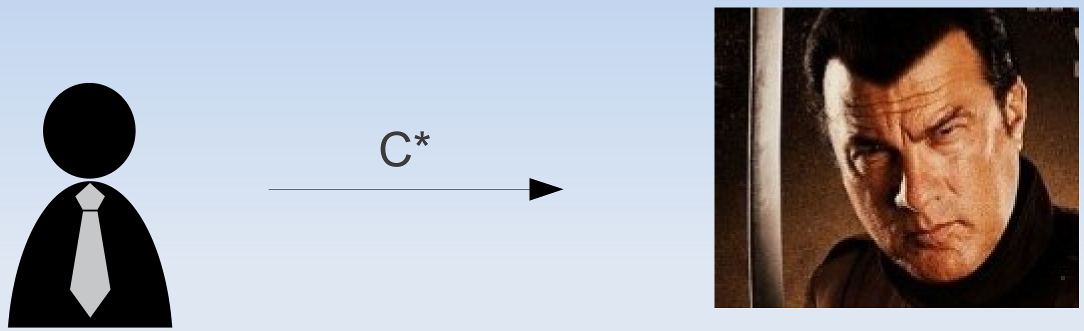




Adversary gets
C_1, …, C_9

A common simplification

- **Simpler:** one user/sender, one challenge



Adversary gets C*

- **Justification:** usually, hybrid argument works
  - E.g., IND-CCA implies multi-user-multi-challenge-IND-CCA
- **But:** connection to real world not tight
- **And:** problematic in some cases (KDM, SOA, leakage)

Overview over this talk

All-But-Many Lossy Trapdoor Functions (ABM-LTFs)
A technical tool specifically designed for the multi-user-multi-challenge case

Construction of ABM-LTFs
A new look on Waters signatures

Applications of ABM-LTFs
Selective opening security, tight IND-CCA security, more (?)

Next stop

All-But-Many Lossy Trapdoor Functions (ABM-LTFs)
A technical tool specifically designed for the multi-user-multi-challenge case

Recap: Lossy Trapdoor Functions

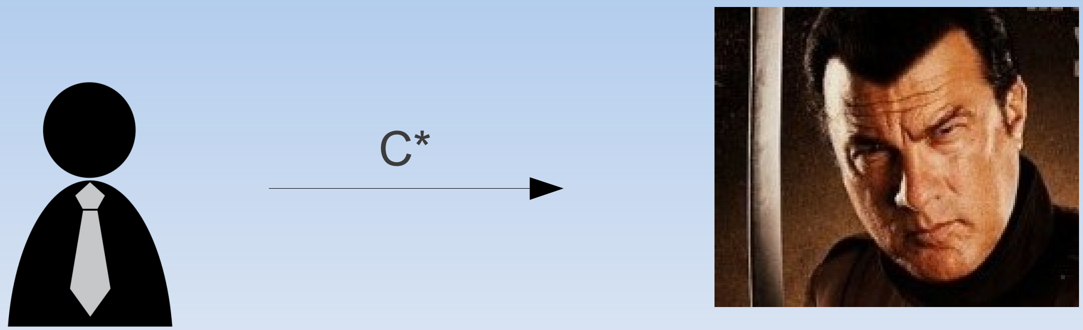
- Algorithms:
  - Gen(1^k) — outputs an evaluation/inversion keypair (ek,ik)
  - Eval(ek,X) — outputs $Y = F_{ek}(X)$ (for X from some preimage set **X**)
  - Invert(ik,Y) — outputs $F_{ek}^{-1}(Y)$
  - LGen(1^k) — outputs a (lossy) evaluation key ek'
- Properties:
  - Indistinguishability: Gen(1^k) ≈ LGen(1^k)
  - Lossiness: image set $F_{ek'}(\mathbf{X})$ "much smaller" than **X**
- Constructions from LWE, DDH, **DCR (efficient!):**

$$ek = (\, pk,\ C = E_{pk}(b)\,)$$

(Invertible mode: b=1, lossy mode: b=0)

$$F_{ek,T}(X) = C^X = E_{pk}(bX)$$

Recap: PKE security from LTFs

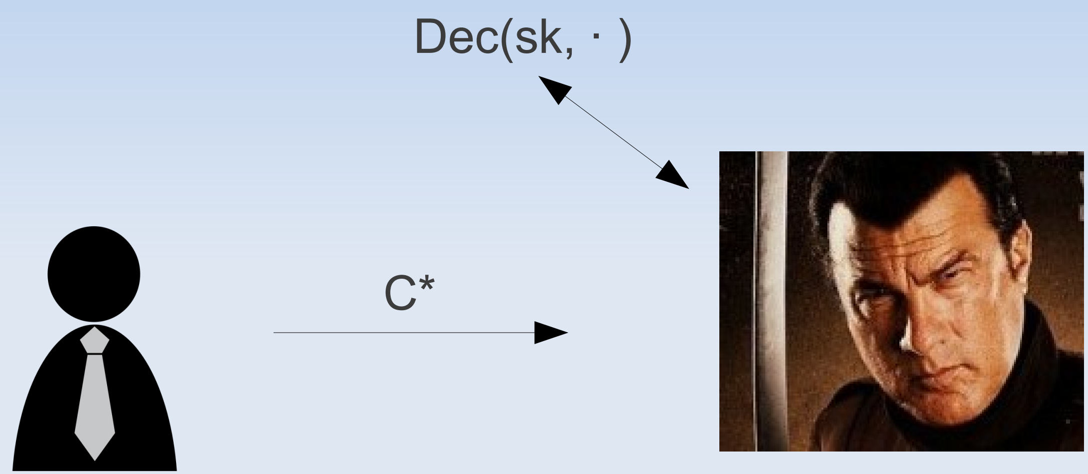


Adversary gets pk, C*

- Intuition:
  - Scheme uses LTF in invertible mode (Enc = LTF evaluation, Dec = LTF inversion)
- To show security:
  - Switch to lossy mode (use LTF indistinguishability)
  - Then, adversary gains no info about message (LTF lossiness)
  - Actually, yields **tight** proof for multi-challenge case

PKE security from LTFs: CCA?

- But wait... adversary could be **active**:



Adversary gets pk, C*
and access to Dec oracle

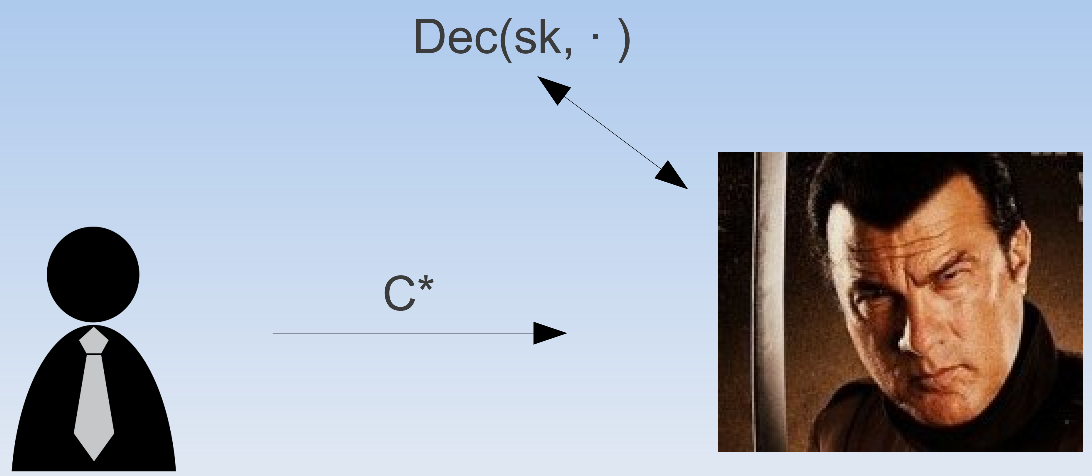
- Problem: if we switch to lossy mode, can't simulate Dec oracle

Recap: All-But-One LTFs

- Algorithms:
  - $Gen(1^k,T^*)$ outputs an evaluation/inversion keypair (ek,ik)
  - $Eval(ek,T,X)$ outputs $Y = F_{ek,T}(X)$ (for tag T)
  - $Invert(ik,T,Y)$ outputs $F_{ek,T}^{-1}(Y)$ (works only for $T \neq T^*$)
- Properties:
  - Indistinguishability: $Gen(1^k,T) \approx Gen(1^k,T')$
  - Lossiness: image set $F_{ek,T^*}(\mathbf{X})$ "much smaller" than $\mathbf{X}$
    (i.e., only F_{ek,T^*} lossy, all other $F_{ek,T}$ are invertible)
- Efficient construction based on Paillier/DJ encryption:

$$ek = (\ pk,\ C = E_{pk}(T^*)\ )$$
$$F_{ek,T}(X) = (C/E_{pk}(T))^X = E_{pk}((T^*-T)X)$$

Recap: PKE security from LTFs



Adversary gets pk, C*
and access to Dec oracle

- Intuition:
  - Scheme uses ABO-LTF, with **unique** tag for every ciphertext (Encryption is "double encryption/evaluation" with ABO-LTF and LTF [PW08])
- To show security (oversimplified):
  - Set lossy tag T* to C*-tag (use ABO-LTF and LTF indistinguishability)
  - Decrypt using ABO-LTF inversion key
  - **Does not work with many challenges (Leakage/KDM/SOA)**

All-But-N LTFs [HLOV09]

- Algorithms:
  - Gen(1^k,T_1^*,…,T_N^*) outputs an evaluation/inversion keypair (ek,ik)
  - Eval(ek,T,X) outputs $Y = F_{ek,T}(X)$ (for tag T)
  - Invert(ik,T,Y) outputs $F_{ek,T}^{-1}(Y)$ (works only for $T \neq T_i^*$)
- Properties:
  - Indistinguishability: $Gen(1^k,T_1,\ldots,T_N) \approx Gen(1^k,T_1',\ldots,T_N')$
  - Lossiness: image set $F_{ek,T^*}(\mathbf{X})$ "much smaller" than **X**
    (i.e., $F_{ek,T}$ lossy if and only if $T=T_i^*$ for some i)
- Construction based on Paillier/DJ encryption:

Prepare degree-N polynomial $f(T) = \sum f_i T^i$ with zeros $T_1^*, \ldots, T_N^*$

$$ek = (\ pk,\ C_0 = E_{pk}(f_0),\ \ldots,\ C_N = E_{pk}(f_N)\ )$$

$$F_{ek,T}(X) = (\ \prod C_i^{T^i}\ )^X = E_{pk}(\ f(T)\ X\ )$$

All-But-N LTFs [HLOV09]

- Algorithms:
  - $Gen(1^k, T_1^*, \ldots, T_N^*)$ outputs an evaluation/inversion keypair (ek,ik)
  - Eval(ek,T,X) outputs $Y = F_{ek,T}(X)$ (for tag T)
  - Invert(ik,T,Y) outputs $F_{ek,T}^{-1}(Y)$ (works only for $T \neq T_i^*$)
- Construction based on Paillier/DJ encryption:

Prepare degree-N polynomial $f(T) = \sum f_i T^i$ with zeros $T_1^*, \ldots, T_N^*$

$$ek = (\ pk,\ C_0 = E_{pk}(f_0),\ \ldots,\ C_N = E_{pk}(f_N)\ )$$

$$F_{ek,T}(X) = (\ \prod C_i^{T^i}\ )^X = E_{pk}(\ f(T)\ X\ )$$

- Problem: space complexity linear in the number of challenges
  - Actually, this is necessary to encode precisely N lossy tags
  - Yields SO-CCA secure PKE that depends on number of challenges

All-But-Many LTFs

- Intuition:
  - There are (superpoly) many lossy tags and (superpoly) many invertible tags
  - Lossy and invertible tags computationally indistinguishable
  - **Invertible** tags easy to sample, but **trapdoor** required to sample **lossy** tags

All-But-Many LTFs

- Algorithms (slightly simplified):
  - Gen(1^k) outputs evaluation/inversion/tag keys (ek,ik)
  - Eval(ek,T,X) outputs $Y = F_{ek,T}(X)$
  - Invert(ik,T,Y) outputs $F_{ek}^{-1}(Y)$ for all invertible tags T
  - LTag(ik) outputs a lossy tag
- Properties:
  - Indistinguishability: $A^{LTag(ik)}(ek) \approx A^{Random\text{-}Tag\text{-}Oracle(ek)}(ek)$ for all PPT A
  - Lossiness: $F_{ek,T}(\mathbf{X})$ "much smaller" than **X** for lossy tags T
  - Evasiveness: $\Pr[\ A^{LTag(ik)}(ek) \rightarrow$ fresh lossy tag $]$ negl. for all PPT A
- Syntactic similarity to **"blinded signatures"** (valid sig = lossy tag)

Next stop

Construction of ABM-LTFs
A new look on Waters signatures

First attempt

- Syntactic similarity to **”blinded signatures”** (valid sig = lossy tag)
- First attempt: so let's simply (Paillier/DJ-)encrypt signatures!

$$T = E(Sign(H))$$

Something unique and public (e.g., chameleon hash of T)

- Evaluation ”magically” verifies signature inside encryption

  ...should end up with C = E(0) **iff** sig is valid, then sets $Y{:=}C^X$
  - Sig valid $\Rightarrow$ C = E(0) $\Rightarrow$ $F_{ek,T}(X) = C^X = E(0)$ lossy
  - Sig invalid $\Rightarrow$ C = E(d) for d≠0 $\Rightarrow$ $F_{ek,T}(X) = C^X = E(dX)$ invertible
- Problem: (Paillier/DJ-)encryption only additively homomorphic
  - **How to evaluate signature using only addition in Z_N?**

Working with encrypted matrices

- **Idea 1:** use matrices instead of single elements (inspired by [PW08])

$$T \rightarrow E(M) = \begin{pmatrix} E(M_{1,1}) & E(M_{1,2}) & E(M_{1,3}) \\ E(M_{2,1}) & E(M_{2,2}) & E(M_{2,3}) \\ E(M_{3,1}) & E(M_{3,2}) & E(M_{3,3}) \end{pmatrix}$$

- Use ”encrypted” matrix-vector multiplication:

$$F_{ek,T}(X) = E(M) \circ \begin{pmatrix} X_1 \\ X_2 \\ X_3 \end{pmatrix} = \begin{pmatrix} \prod_j E(M_{1,j})^{Xj} \\ \prod_j E(M_{2,j})^{Xj} \\ \prod_j E(M_{3,j})^{Xj} \end{pmatrix} = E(M \cdot X)$$

- $F_{ek,T}$ lossy $\Leftrightarrow$ M non-invertible $\Leftrightarrow$ det(M)=0 (or non-invertible)
- **Payoff:** det(M) can be **cubic** in encrypted values
- **Use determinant to encode more complex computations**

Waters signatures

- Assume pairing $e: G \times G \to G_T$
- Verification key: $A = g^a$, $B = g^b$, $H_0,\ldots,H_n$ ($H(M) := H_0 \prod H_j^{M_j}$)
- Signature for M: $R = g^r$ $Z = g^{ab} H(M)^r$
- Verification: check $e(A,B)\, e(H(M),R) \stackrel{?}{=} e(g,Z)$
- Secure under CDH in G (Waters' hash H plays crucial role in proof)

- **Idea 2:** emulate Waters signatures in Z_N
  - Use encryption instead of exponentiation (A=E(a), B=E(b), etc.)
  - Pairing becomes Paillier/DJ multiplication **(encode verification into det(M)!)**
  - CDH in G becomes **"Paillier-No-Mult"**: E(a), E(b) → E(ab) hard

The construction (slightly simplified)

- Evaluation key: $ek = (\ A{=}E(a),\ \ B{=}E(b),\ \ H_i{=}E(h_i)\ \ (i{=}0,\dots,n)\ )$
- Inversion key: $ik = (ek,sk)$ (sk = secret key for P/DJ encryption)
- Tags: $(\ R{=}E(r),\ \ Z{=}E(z),\ \ rnd\ )$ (rnd is randomness for CHF)

$$T \rightarrow E(M) = \begin{pmatrix} E(z) & E(a) & E(r) \\ E(b) & E(1) & E(0) \\ E(h) & E(0) & E(1) \end{pmatrix} \quad \text{with } E(h) = H(t) = h_0 + \textstyle\sum t_i h_i \quad \text{for } t = CHF(R,Z;rnd)$$

Note: $\det(M) = z - (ab+rh)$, **so:** T lossy $\Leftrightarrow$ M singular $\overset{*}{\Leftrightarrow}$ $z = ab + rh$

- Lossy tags computable from CHF trapdoor, a,b, and the h_i
- Evaluation: $F_{ek,T}(X) = E(M) \circ X = E(M \cdot X)$
- Inversion: decrypt $E(M)$ and $E(M \cdot X)$, solve for X

Properties of our ABM-LTF

- Tags: (R=E(r), Z=E(z), rnd) (rnd is randomness for CHF)

$$T \rightarrow E(M) = \begin{pmatrix} E(z) & E(a) & E(r) \\ E(b) & E(1) & E(0) \\ E(h) & E(0) & E(1) \end{pmatrix} \quad \text{with } E(h) = H(t) = h_0 + \sum t_i h_i \quad \text{for } t = CHF(R,Z;rnd)$$

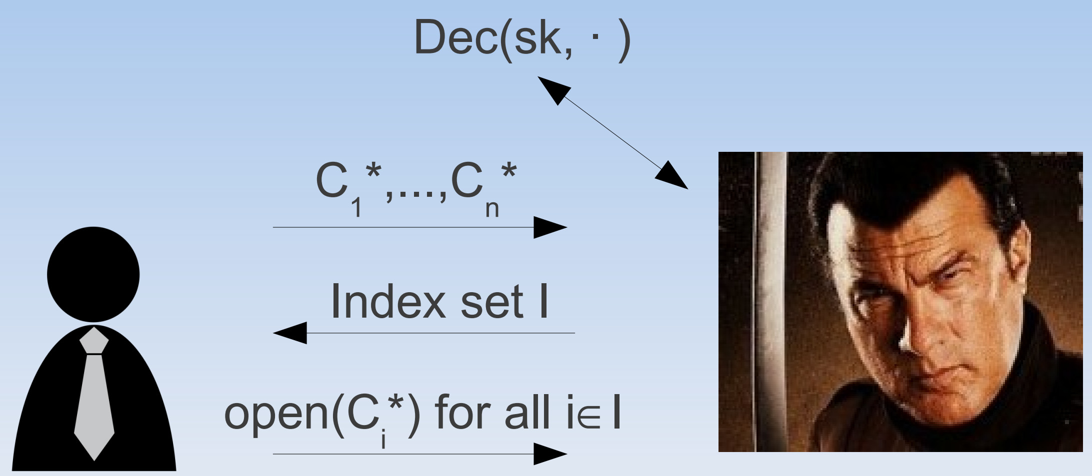
Note: det(M) = z – (ab + rh), **so:** T lossy $\Leftrightarrow$ M singular $\overset{*}{\Leftrightarrow}$ z = ab + rh

- Lossy tags computable from CHF trapdoor, a,b, and the h_i
- Indistinguishability (lossy tags look like random tags):
  - Lossy tags can be produced without sk $\Rightarrow$ reduction to DCR
- Evasiveness (cannot produce one more lossy tag):
  - Lossy tags are essentially Waters-in-Z_N sigs
  - Proof similar to Waters' proof, but reduction to **Paillier-No-Mult**

Next stop

Applications of ABM-LTFs
Selective opening security, tight IND-CCA security, more (?)

Selective Opening Security

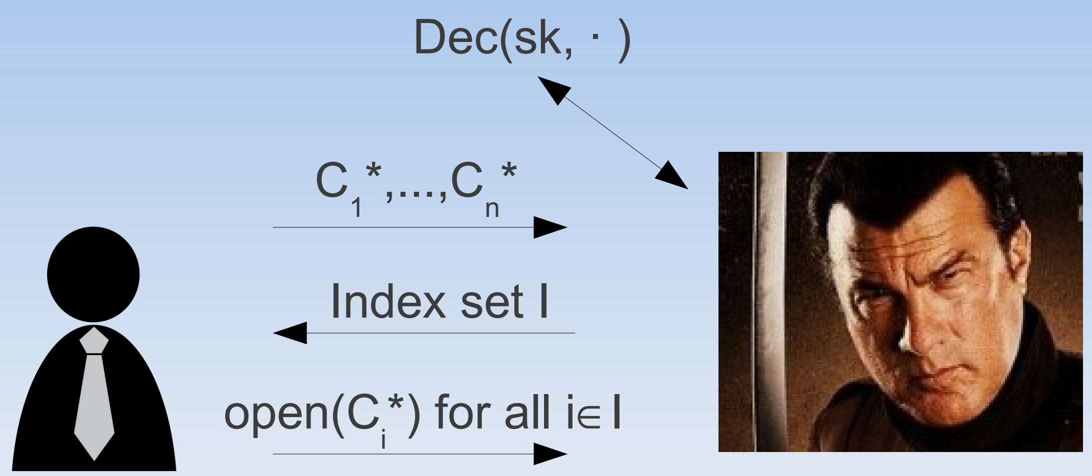


Adversary gets pk, $C_1^*,\ldots,C_n^*$,

openings of all C_i ($i \in I$)

- Intuition: adaptive corruption of multiple senders
- Security can be indistinguishability- or simulation-based
  - Intuition: adversary should not learn anything about unopened ciphertexts
  - **No hybrid argument, multiple challenges inherent**
- Without Dec oracle, lossy encryption works fine (make Enc lossy)
  - **Problem:** what if Enc is lossy and adversary makes Dec queries?

Selective Opening Security



Adversary gets pk, $C_1^*, \ldots, C_n^*$,
openings of all C_i ($i \in I$)

- **Idea** [HLOV09]: (double) encryption with tags, make **only** C_i^* lossy
  - [HLOV09] only have All-But-N-LTFs (inefficient, construction linear in n)
- Used with our ABM-LTF:
  - First SOA-CCA secure scheme with constant-sized ciphertexts and keys
  - Complexity of scheme does not grow with n

Tight CCA security

- **Open problem:** construct tightly CCA-secure PKE scheme
  - ”Tightly secure”: reduction is tight in number of challenges and users
  - Known: Cramer-Shoup can be proven tightly in number of users
- **Idea:** make all challenges lossy simultaneously (ABM-LTF)
- **Problem:** Paillier/DJ-based construction is itself not tight
- **Solution:** another ABM-LTF construction based on pairings
  - **Idea:** combine Boneh-Boyen sigs with ”blinding by subgroup element”
  - Yields tight CCA security in number of challenges, **but:**
  - Needs strong assumptions: strong DDH + subgroup indistinguishability
- **Better ABM-LTF constructions?**

More applications?

- **CCA-secure Key-Dependent Message Security (?)**
  - Many challenges, **but all may depend on secret key**
  - No hybrid argument, and ABM-LTF application not straightforward
  - But: use ABM-LTFs without inversion?
- **New signature schemes (?)**
  - Message = suitable ABM-LTF tag chosen by signer
  - Signature = ”proof” that tag for ABM-LTF is lossy
  - Does not work: ”proof” = different X_1, X_2 with $F_{ek,T}(X_1)=F_{ek,T}(X_2)$
- **Leakage resilience?**

Last slide

- **Open problems:**
  - Smaller, better, faster ABM-LTFs (from more reasonable assumptions)
  - More applications (KDM-CCA, sigs, …)