3039813559

EAV. deterministic
CPA. randomized

CCA + UE = Authenticate Encryption

private-key crypto : Enc + Mac

public-key crypto : PKE / IBE + Digi Sig

stronger but slower than Enc

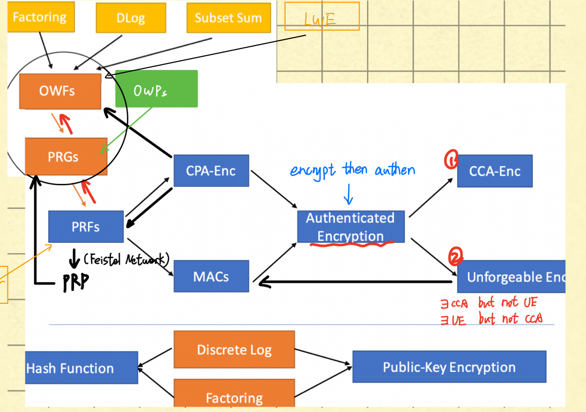
EAV = CPA. randomized

S_2 imply S_1

Assume for contradiction ∃ PPT adversary A break S_1 with non-neg probability

We will use A to construct adversary B break S_2 with non-neg probability

Contradiction to S_2 is ... thus initial assumption is false. S_1 is also ...



- View H as a random function

For $x_1, \dots x_q$ $\Pr[\exists\, i,j\ H(x_i) = H(x_j)] \approx \frac{q^2}{2 \cdot 2^\ell}$

- Thus, probability is ½ for $q = \Theta(2^{\ell/2})$

- Birthday problem: What is the probability that q people have birthday on the same day of the year?
  - Only need $\sqrt{365} \approx 23$ people to get a collision with probability ½
- Attempt 1: The probability two hashes collide is $1/2^\ell$. Thus, probability of collision is $\binom{q}{2} \cdot 1/2^\ell$.

num of func

$\{0,1\}^n \to \{0,1\}^n$. $2^n \times n$ bit $\Rightarrow 2^{2^n \cdot n}$ 个函数

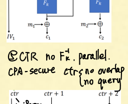
$\{0,1\}^n \to \{0,1\}^{2n}$. 个函数. $2^{2^n \cdot 2n}$

from $\log_2(n)$ bits to n^2 bits? $2^{\log_2 n} \cdot n^2 = n^3 \Rightarrow 2^{n^3}$ 个函数.

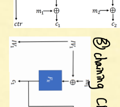
num of permutation

from $\{0,1\}^n$ to $\{0,1\}^n$ is $(2^n)!$

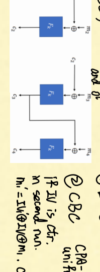
④ OFB no F_k^{-1}. F_k 提前 sequencial



⑤ CTR no F_k^{-1}. parallel. CPA-secure ctr { no overlap, no query }



③ chaining CBC, choose $m_3 = IV_1 \oplus c_2 \oplus m_1$ and c_3



① ECB $m_1 \to \boxed{F_k} \to c_1$. not random

② CBC CPA-secure uniform IV_1

if IV is ctr. in second run. $m_1' = IV_1 \oplus IV_1' \oplus m_1$. $c_1' = F_k(m_1' \oplus IV_1') = c_1$

hardness relationship, weak ⇒ strong

DDH ⟹ CDH ⟹ DLog ⟹ CRHF ⟹ OWF

CDH ⟹ DBDH

cyclic group

DLOG (Discrete-log) > CDH (Computational Diffie-Hellman) > DDH (Decisional Diffie-Hellman)

$h = g^x \to x$ $g^a, g^b \to g^{ab}$ $g^a, g^b \to$ distinguish g^{ab}. g^r

construct CRHF OWF   construct PRG

$\mathrm{DLog}_{\mathcal{A},\mathcal{G}}(n)$
1. Run $\mathcal{G}(1^n)$ to obtain (G, g, q).
2. Sample uniform $h \in G$.
3. $\mathcal{A}$ is given (G, g, q, h) and it outputs x.
4. Output 1 if $g^x = h$ and 0 otherwise.

We say that the Discrete-Log Problem is hard relative to $\mathcal{G}$ if ∀ PPT adversaries $\mathcal{A}$, ∃ function negl(·) such that

$$|\Pr[\mathrm{DLog}_{\mathcal{A},\mathcal{G}}(n) = 1]| \le \mathrm{negl}(n).$$

$\mathrm{CDH}_{\mathcal{A},\mathcal{G}}(n)$
1. Run $\mathcal{G}(1^n)$ to obtain (G, g, q).
2. Sample uniform $a, b \in \mathbb{Z}_q^*$.
3. $\mathcal{A}$ is given (G, g, q, g^a, g^b) and it outputs h.
4. Output 1 if $g^{ab} = h$ and 0 otherwise.

We say that the CDH Problem is hard relative to $\mathcal{G}$ if ∀ PPT adversaries $\mathcal{A}$, ∃ function negl(·) such that

$$|\Pr[\mathrm{CDH}_{\mathcal{A},\mathcal{G}}(n) = 1]| \le \mathrm{negl}(n).$$

$\mathrm{DDH}_{\mathcal{A},\mathcal{G}}(n)$
1. Run $\mathcal{G}(1^n)$ to obtain (G, g, q).
2. Sample uniform $a, b, r \in \mathbb{Z}_q^*$. Sample a uniform bit $c \in \{0,1\}$.
3. $\mathcal{A}$ is given $(G, g, q, g^a, g^b, g^{ab+cr})$ and it outputs c'.
4. Output 1 if $c = c'$ and 0 otherwise.

We say that the DDH Problem is hard relative to $\mathcal{G}$ if ∀ PPT adversaries $\mathcal{A}$, ∃ function negl(·) such that

$$|\Pr[\mathrm{DDH}_{\mathcal{A},\mathcal{G}}(n) = 1]| \le \frac{1}{2} + \mathrm{negl}(n).$$

bilinear group

1. "Groups where CDH is hard, but DDH is easy"
2. Consider a group G of prime order q and generator g:
3. We get a pairing operation e such that:
   - $e : G \times G \to G_T$ another group
   - If g is a generator of G then $e(g, g)$ is a generator of G_T
   - $\forall a, b \in \mathbb{Z}_q^*$, $e(g^a, g^b) = e(g, g)^{ab}$
4. Intuition:
   - DDH is easy because if A, B, C is a DDH tuple, we can check $e(A, B) = e(g, C)$

DBDH (Decisional Bilinear Diffie-Hellman)

given g^a, g^b, g^c, can compute $e(g,g)^{ab}$, $e(g,g)^{bc}$, $e(g,g)^{ac}$

cannot compute $e(g,g)^{abc}$

$\mathrm{DBDH}_{\mathcal{A},\mathcal{G}}(n)$
1. Run $\mathcal{G}(1^n)$ to obtain $(G, G_T, g, q, e(\cdot,\cdot))$.
2. Sample uniform $a, b, c, r \in \mathbb{Z}_q^*$. Sample a uniform bit $\beta \in \{0,1\}$.
3. $\mathcal{A}$ is given $(G, G_T, g, q, g^a, g^b, g^c, e(g,g)^{abc+\beta r})$ and it outputs β'.
4. Output 1 if $\beta = \beta'$ and 0 otherwise.

We say that the DBDH Problem is hard relative to $\mathcal{G}$ if ∀ PPT adversaries $\mathcal{A}$, ∃ function negl(·) such that

$$|\Pr[\mathrm{DBDH}_{\mathcal{A},\mathcal{G}}(n) = 1] - \frac{1}{2}| \le \mathrm{negl}(n).$$

CDH ⟹ DLog:
1. Want to show that if computing x from g^x in G was easy, then so is computing g^{ab} from g^a and g^b in G.
2. Given (G, g, q, g^a, g^b), run $\mathcal{A}_{\mathrm{Dlog}}$ on g^a to get a. Compute $(g^b)^a = g^{ab}$.
3. This approach wins with the same probability that $\mathcal{A}_{\mathrm{Dlog}}$ solves the Dlog instance (non-negl).

DDH ⟹ CDH:
1. Want to show that if computing g^{ab} from g^a and g^b in G was easy, then so is distinguishing DDH triples.
2. Given $(G, g, q, g^a, g^b, g^{ab+cr})$, run $\mathcal{A}_{\mathrm{CDH}}$ on g^a and g^b to get g^{ab} and check if it equals g^{ab+cr}.
3. This approach wins the DDH game with non-negl $- \frac{1}{q} =$ non-negl probability.

neg fn

12   $\log_2 n$ $n^{\frac{1}{3}}$ $2^{\log n}$ = n n^3 $2^{\sqrt{n}}, 2^{\log^2 n}$ 2^n

Const < log < sublinear < linear < poly < subexp < exp < n! < n^n

neg def. ① ∀ polynomial p. for large enough n. $f(n) < \frac{1}{p(n)}$. $\lim_{n\to\infty} \frac{p(n)}{f(n)^{-1}} = 0$

② ∀ constant c. for large enough n. $f(n) < n^{-c}$.

∃ $N \in \mathbb{Z}^+$. ∀ n > N

neg 2^{-n} 1.01^{-n} 2^{-n^2} $2^{-(\log n)^2}$ $2^{-(\log n)^3}$ $2^{-\sqrt{n}}$ $n^{-\log n}$

$$f(n) = \begin{cases} 2^{-\log^2 n} & n \text{ is odd} \\ 2^{-n} & n \text{ is even} \end{cases}$$

non-neg n^{-100} $2^{-\log n} = \frac{1}{n}$ $2^{\log^2 n} = n^{\log n}$ $e^{\log n} = n^{\log e} = n$

- $f(n) = 2^{-n}$ for n mod 2 = 0
- $\quad = n^{-c}$ for n mod 2 = 1

poly · neg = neg

prove neg ⇒ find big enough N. prove non-neg ⇒ find right p

∀p. ∃N. ∀n>N. $f(n) < \frac{1}{p(n)}$ or n^{-c}

eg: prove 2^{-n} is neg

$p(n) = c_d n^d + c_{d-1} n^{d-1} + \dots + c_0 \le d \cdot c^* n^d$, $c^* = \max\{c_0 \cdots c_d\}$

for $n > dc^*$ ⇒ $\le n^{d+1} = 2^{(d+1)\log_2 n}$

for $n > (d+1)\log n$ ⇒ $< 2^n$

$2^{-n} < \frac{1}{p(n)}$

$N = \max\{dc^*, (d+1)\log n\}$

PRG pseudorandom generator

$G: \{0,1\}^n \to \{0,1\}^{\ell(n)}$, where $\ell(n) > n$

2^n in $2^{\ell(n)}$

G is pseudorandom generator if ∀ PPT A we have ∃ negl(·) such that,

$$\left| \Pr_{x \leftarrow U_{\ell(n)}}[A(x) = 1] - \Pr_{s \leftarrow U_n}[A(G(s)) = 1] \right| \le negl(n)$$

(uniform)

$\mathrm{PRG}_{A,G}(1^n)$
1. $b \leftarrow \{0,1\}$,
2. If $b = 0$ set $x \leftarrow G(U_n)$ else set $x \leftarrow U_{\ell(n)}$.
3. Give x to A
4. A output b'
5. Output 1 if $b = b'$ and 0 otherwise

G is a PRG if ∀ PPT A it holds that:

$$\Pr[\mathrm{PRG}_{A,G}(1^n) = 1] \le \frac{1}{2} + negl(n)$$

Seed must be kept secret. Analogous to the secret key in an encryption scheme.

PRF pseudorandom fn

Let F: $\{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ be an efficient, length-preserving, keyed function. F is a PRF if for all PPT distinguishers D, there is a negligible function $negl(\cdot)$ such that:

$$\left|\Pr[D^{F_k(\cdot)}(1^n) = 1] - \Pr[D^{f(\cdot)}(1^n) = 1]\right| \le negl(n)$$

where $k \leftarrow U_n$ and $f \leftarrow Func_n$.

PRP pseudorandom permutation

Let F: $\{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ be an efficient, length-preserving, keyed permutation. F is a (strong) PRP if for all PPT distinguishers D, there is a negligible function $negl(\cdot)$ such that:

$$\left|\Pr\left[D^{F_k(\cdot), F_k^{-1}(\cdot)}(1^n) = 1\right] - \Pr\left[D^{f(\cdot), f^{-1}(\cdot)}(1^n) = 1\right]\right| \le negl(n)$$

where $k \leftarrow U_n$ and $f \leftarrow Perm_n$.

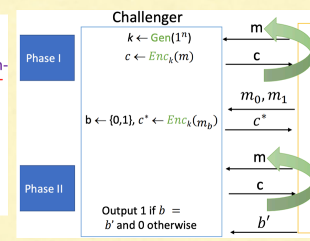
CPA chosen plaintext attack

$\mathrm{PrivK}^{\mathrm{CPA}}_{A,\Pi}(n)$
1. Sample $k \leftarrow \mathrm{Gen}(1^n)$, $A^{Enc_k(\cdot)}$ outputs $m_0, m_1 \in \{0,1\}^*, |m_0| = |m_1|$.
2. $b \leftarrow \{0,1\}$, $c \leftarrow Enc_k(m_b)$
3. c is given to $A^{Enc_k(\cdot)}$
4. $A^{Enc_k(\cdot)}$ output b'
5. Output 1 if $b = b'$ and 0 otherwise

Encryption scheme $\Pi = (Gen, Enc, Dec)$ has indistinguishable encryptions under chosen-plaintext attack, or is CPA-secure if

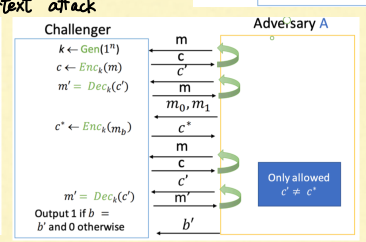
∀ PPT A it holds that:

$$\Pr[\mathrm{PrivK}^{\mathrm{CPA}}_{A,\Pi} = 1] \le \frac{1}{2} + \mathrm{negl(n)}$$



CCA chosen ciphertext attack

$\mathrm{PrivK}^{\mathrm{CPA}}_{A,\Pi}(n)$
1. Sample $k \leftarrow \mathrm{Gen}(1^n)$, $A^{Enc_k(\cdot)}$ outputs $m_0, m_1 \in \{0,1\}^*, |m_0| = |m_1|$.
2. $b \leftarrow \{0,1\}$, $c \leftarrow Enc_k(m_b)$
3. c is given to $A^{Enc_k(\cdot)}$
4. $A^{Enc_k(\cdot)}$ output b'
5. Output 1 if $b = b'$ and 0 otherwise

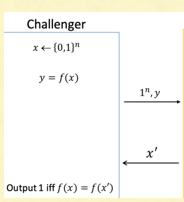


OWF one-way fn

- Syntax:

$$f(x) = y$$

- A function $f : \{0,1\}^* \to \{0,1\}^*$ is *one-way* if
- It's easy to compute, i.e., computing $f(x)$ runs in "probabilistic polynomial time.", but
- It's hard to invert, i.e., there is no "probabilistic polynomial time" algorithm that can compute $f^{-1}(y)$.
- Note: $\{0,1\}^* \to \{0,1\}^*$ means the input and output can be arbitrarily long bit strings.



- $\text{OWF-Sec}_{\mathcal{A},f}(n)$:
  1. The challenger randomly samples an input $x \leftarrow \{0,1\}^n$ and gives $f(x)$ to the adversary $\mathcal{A}$ along with 1^n.
  2. $\mathcal{A}$ produces a value $x' \in \{0,1\}^n$.
  3. $\mathcal{A}$ wins (and the game outputs 1) if $f(x') = f(x)$
  4. Otherwise, $\mathcal{A}$ loses (the game outputs 0).
- The probability $\mathcal{A}$ wins the above game should be at most negl(n) for f to be secure.
- This can be expressed equivalently as:

$$\Pr_{x \leftarrow \{0,1\}^n}[\mathcal{A}(1^n, f(x)) \in f^{-1}(f(x))] \le \mathrm{negl}(n).$$

OWP one way permutation

hc Hardness-Concentrate bit

- A function hc: $\{0,1\}^n \to \{0,1\}$ is a hard-concentrate predicate of a permutation f if:

1. hc can be computed in polynomial time.
2. ∀ PPT A, ∃ a negligible function neg(·) such that OWP $f(x) \to x$ unique $\to hc(x)$ unique

$$\Pr_{x \leftarrow \{0,1\}^n}[A(1^n, f(x)) = hc(x)] \le \frac{1}{2} + neg(n)$$

security game

challenger sample $x \leftarrow \{0,1\}^n$. give $y = f(x)$ to A.

A output bit b.

challenger check $hc(x) = b$

valid OWF & hc

$g(x, r) = f(x) || r$, $|x| = |r|$

OWF   OWP

$$hc(x, r) = \sum_{i=1}^{n} x_i \cdot r_i \bmod 2$$

Mac message authentication code

A Message Authentication Codes (MAC) is a keyed checksum, which is sent along with the message. It takes in a fixed-length secret key and an arbitrary-length message, and outputs a fixed-length checksum. A secure MAC has the property that any change to the message will render the checksum invalid.

unforgeability (new m. quieried t)

strong unforgeability (new m. quieried t). (quieried m, new t)

adversary want to forge a valid (m^*, t^*) pair. $m^* \notin M$

A MAC scheme consists of 3 PPT algorithms (Gen, MAC, Verify):
- $Gen(1^n)$: Outputs a key k.
- $MAC_k(m)$: Outputs a tag t.
- $Verify_k(m, t)$: Outputs 0/1.

These satisfy 2 properties:
1. **Correctness:** $\forall n, k \leftarrow Gen(1^n), \forall m \in \{0,1\}^*$, we have that $Verify_k(m, MAC_k(m)) = 1$.
2. **Security:** $Verify_k(m, t)$ outputs 1 if and only if $MAC_k(m) = t$.

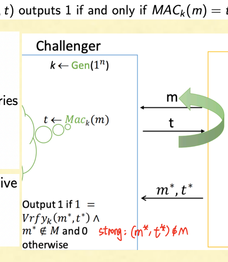
The game is between a challenger C and the adversary $\mathcal{A}$.

$\mathrm{MACForge}_{\mathcal{A},\Pi}(1^n)$:
1. C samples $k \leftarrow Gen(1^n)$.
2. $\mathcal{A}$ makes MAC queries to the challenger. Let M be the list of queries $\mathcal{A}$ makes.
3. Finally, $\mathcal{A}$ outputs (m^*, t^*).   strong: $(m^*, t^*) \notin M$
4. C outputs 1 if $Verify(m^*, t^*) = 1 \wedge m* \notin M$ and 0 otherwise.

$\Pi = (Gen, MAC, Verify)$ is existentially unforgeable under the adaptive chosen attack if ∀ PPT $\mathcal{A}$ it holds that:

ea-cma-secure

$$\Pr[\mathrm{MACForge}_{\mathcal{A},\Pi} = 1] \le \mathrm{negl}(n)$$



CRHF collision resistant hash fn

- Syntax:

$$H^s(x) = y$$

- A **collision** in H^s is a pair (x, x') such that $x \neq x'$ but $H^s(x) = H^s(x')$.
- H is guaranteed to have collisions. We require that $|y| < |x|$ (H is **compressing**).
- If it's hard to find those collisions, then the hash function is **collision-resistant**.
- The hash function $\mathcal{H}$ is a pair of algorithms: $\mathcal{H} = (\mathrm{Gen}, H)$.
- Gen: outputs a random key/seed s:

$$s \leftarrow \mathrm{Gen}(1^n)$$

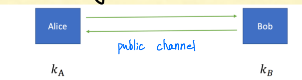
The key is allowed to be public.
- H^s: This is also sometimes referred to as the hash function. The output length – and sometimes the input length – are fixed. H^s is deterministic.
- **Summary:** The adversary is given s and a description of H, and they try to find a collision in H^s with non-negligible probability.
- $\text{Hash-coll}_{\mathcal{A},\mathcal{H}}(n)$:
  1. The challenger samples a key $s \leftarrow \mathrm{Gen}(1^n)$ and gives s to the adversary $\mathcal{A}$.
  2. $\mathcal{A}$ produces two inputs (x, x') to H^s.
  3. $\mathcal{A}$ wins (and the game outputs 1) if (x, x') are a collision:

$$x \neq x' \text{ and } H^s(x) = H^s(x')$$

  Otherwise, $\mathcal{A}$ loses (the game outputs 0).
- Note that the adversary can compute H^s by themselves.
- $\mathcal{H}$ is **collision-resistant** if for any PPT adversary $\mathcal{A}$, there is a negligible function negl such that:

$$\Pr[\text{Hash-coll}_{\mathcal{A},\mathcal{H}}(n) = 1] \le \mathrm{negl}(n)$$

key exchange



- Correctness: $k = k_A = k_B$
- Security (Informally): Eve listening on the channel should not be able to guess k.

Consists of three randomized algorithms (P_1, P_2, P_3):
- Alice computes $(m_1, st) \leftarrow P_1(1^n)$ and sends m_1 to Bob.
- Bob computes $(m_2, k) \leftarrow P_2(m_1)$. Then he sends m_2 to Alice and outputs k.
- Alice computes $k \leftarrow P_3(st, m_2)$ and outputs k.

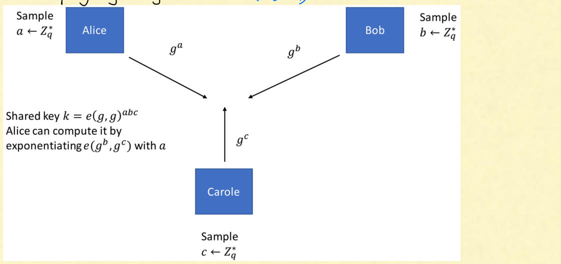
- **Correctness**: Both parties get the same key k.
- **Security**: No eavesdropper can distinguish between (m_1, m_2, k) and (m_1, m_2, r) where r is a random element.

EAV

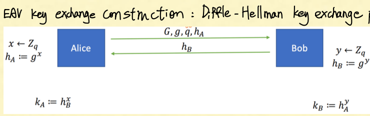
$KE^{eav}_{A,\Pi}(n)$
1. Two parties holding 1^n execute Π. This results in a transcript Ω of the communication and a key k output for each party.
2. Sample a uniform bit b. If $b = 0$, then set $\hat{k} = k$, else set $\hat{k}$ uniformly. ($b=0$ key, $b=1$ random)
3. A is given $(\Omega, \hat{k})$ and it outputs b'. (cannot distinguish)
4. Output 1 if $b' = b$ and 0 otherwise

A key-exchange protocol Π is secure if
$\forall$ PPT A $\exists$ negl such that: $|\Pr[KE^{eav}_{A,\Pi}(n) = 1]| \le ½ + negl(n)$.

EAV 3-party key exchange construction (DBDH)



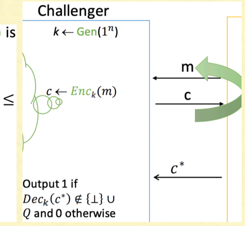
EAV key exchange construction: Diffie-Hellman key exchange protocol (DDH)



UE adversary want to output a valid c^* for $m^* \notin N$

$EncForge_{A,\Pi}(1^n)$
1. Sample $k \leftarrow Gen(1^n)$.
2. Let c^* be the output of $A^{Enc_k(\cdot)}(1^n)$. Let Q be the list of messages A gets ciphertexts for from the oracle.
3. Output 1 if $Dec_k(c^*) \notin \{\perp\} \cup Q$ and 0 otherwise.

$\Pi = (Gen, Enc, Dec)$ is unforgeable if
$\forall$ PPT A it holds that:
$\Pr[EncForge_{A,\Pi} = 1] \le negl(n)$



OWF

CRHF $H^s : \{0,1\}^n \to \{0,1\}^{\ell} \to$ OWF

Prove: $f(s \parallel x) = s \parallel H^s(x)$ is a OWF.

Step 2: Construction of $\mathcal{B}$:
- $\mathcal{B}$ is given the truly random seed s from the CRHF challenger.
- $\mathcal{B}$ samples a random $x \leftarrow \{0,1\}^n$ (calculate $H^s(x)$) and runs $\mathcal{A}$ on $H^s(x)$ to obtain x'. ($s \parallel H^s(x) \to s \parallel x'$)
- If $x = x'$, abort.
- Otherwise, output (x, x') as a collision.

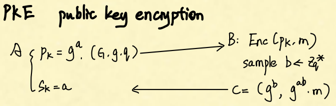
PRG → OWF PRG is $\frac{n}{2} \to n$

Construction of $\mathcal{B}$:
(a) The PRG challenger gives $\mathcal{B}$ a string $y \in \{0,1\}^n$. Either $y = G(x)$, where $x \leftarrow \{0,1\}^{n/2}$, or $y \leftarrow \{0,1\}^n$.
(b) $\mathcal{B}$ runs $\mathcal{A}$ on inputs $(1^n, y)$. $\mathcal{A}$ outputs $x' \in \{0,1\}^{n/2}$.
(c) $\mathcal{B}$ checks whether $G(x') = y$. If so, $\mathcal{B}$ outputs 1. If not, $\mathcal{B}$ outputs 0.

hc → OWF. f is one-to-one

$\mathcal{B}$:
(a) In the hard-concentrate predicate security game, the challenger samples $x \leftarrow \{0,1\}^n$ and gives $(1^n, f(x))$ to $\mathcal{B}$.
(b) $\mathcal{B}$ runs $\mathcal{A}(1^n, f(x))$, to obtain x'.
(c) $\mathcal{B}$ checks whether $f(x') = f(x)$.
- If so, $\mathcal{B}$ computes and outputs $hc(x')$.[3]
- If not, $\mathcal{B}$ samples $b \leftarrow \{0,1\}$ and outputs it.

PKE public key encryption



- $Gen(1^n) \to (\mathbf{sk}, \mathbf{pk})$
- $Enc(\mathbf{pk}, m) \to c$
- $Dec(\mathbf{sk}, c) \to m / \perp$

and these satisfy two properties
- **Correctness**: $Dec(sk, Enc(pk, m)) = m$.
- **Security**: EAV = CPA security / CCA security (public key → no Enc oracle; give Dec oracle)

EAV / CPA
- Challenger samples $(sk, pk) \leftarrow Gen(1^n)$ and gives pk to $\mathcal{A}$.
- $\mathcal{A}$ outputs two message m_0, m_1.
- Challenger samples a bit $b \in \{0,1\}$ and outputs $Enc(pk, m_b)$.
- $\mathcal{A}$ outputs b' as a guess for b.

CPA-secure if for all PPT $\mathcal{A}$

$$Pr[b' = b] \le \frac{1}{2} + negl(n)$$

CCA

$PubK^{CCA}_{A,\Pi}(n)$
1. $(pk, sk) \leftarrow G(1^n)$ and give pk to A.
2. $A^{Dec(sk,\cdot)}$ outputs $m_0, m_1 \in \{0,1\}^*$, $|m_0| = |m_1|$.
3. $b \leftarrow \{0,1\}$, $c^* \leftarrow Enc(pk, m_b)$
4. c is given to $A^{Dec(sk,\cdot)}$ and it outputs b' (query c^* not allowed)
5. Output 1 if $b = b'$ and 0 otherwise

Encryption scheme $\Pi = (Gen, Enc, Dec)$ is indistinguishable in the presence of a CCA attacker, or is CCA-secure if
$\forall$ PPT A it holds that:
$\Pr[PubK^{cca}_{A,\Pi} = 1] \le \frac{1}{2} + negl(n)$

EAV/CPA PKE Construction: El Gamal (CDH)
- $Gen(1^n)$: Generate cyclic group $\mathbb{G}$ of order q and a generator g. (fixed) Sample $x \in \mathbb{Z}_q$ and $h = g^x$.
  Output $pk = (\mathbb{G}, q, g, h)$, $sk = x$
- $Enc(pk, m) \to (c_1, c_2)$: Sample $r \in \mathbb{Z}_q$. ($m \in \mathbb{G}$)
  Output $(c_1, c_2) = (g^r, m \cdot h^r)$ 长度是m的2倍 → hybrid encryption
- $Dec(sk, (c_1, c_2)) \to m$: **Output** $m = \frac{c_2}{c_1^x}$

IBE → CPA PKE (public key → no Enc oracle; give Dec oracle)

Show that IBE implies PKE, i.e., given a CPA-secure IBE scheme (S, K, E, D), construct a CPA-secure PKE scheme (Gen, Enc, Dec).

- $Gen(1^\lambda)$: Run $S(1^\lambda) \to (msk, mpk)$ and return $sk = msk$, $pk = mpk$
- $Enc(pk, m)$: Sample a random ID and run $E(mpk, ID, m) \to ct$. Output (ID, ct) as the ciphertext.
- $Dec(sk, (ID, ct))$: First, derive sk_{ID} for the ID and then run $Dec(sk_{ID}, ct) \to m$.

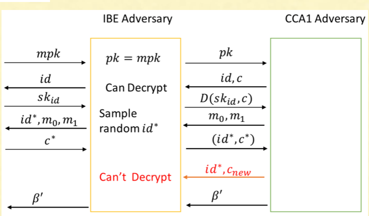
Correctness: follows from correctness of IBE.

CPA security: Suppose PKE was not CPA-secure. Let $\mathcal{A}$ be an adversary that wins in the CPA game for PKE. We'll build an adversary B to break CPA security of IBE.
- IBE challenger runs $S(1^\lambda) \to (msk, mpk)$ and gives mpk to B. B sends it to A as pk.
- A outputs two challenge messages m_0, m_1.
- B samples a random ID and sends (ID, m_0, m_1) to the IBE challenger.
- The IBE challenger chooses random $b = 0/1$ and returns $c = E(mpk, ID, m_b)$.
- B sends (ID, c) to A and outputs whatever A outputs.

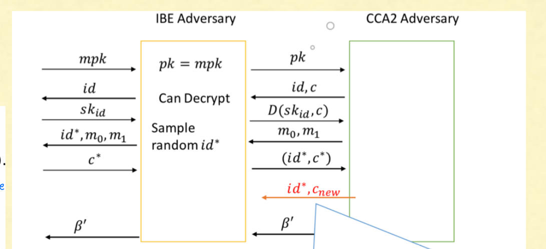
Hash-Mac
- Let $(Gen, Mac, Vrfy)$ be a MAC on messages of length $\ell(n)$ and (Gen_H, H) be a hash function with output length $\ell(n)$. Then MAC $(Gen', Mac', Vrfy')$ for arbitrary-length messages is:
- $Gen'(1^n)$: Output $k' = (k, s)$ where $k \leftarrow Gen(1^n)$ and $s \leftarrow Gen_H(1^n)$.
- $Mac'_{k'}(m \in \{0,1\}^*)$: Output $Mac_k(H^s(m))$
- $Vrfy'_{k'}(m, t)$: Output 1 iff $Vrfy_k(H^s(m), t) = 1$.

CCA PKE construction (IBE)

CCA phase-1 secure (cannot answer phase-2 query) — no Dec oracle in phase-2
- $Gen(1^n)$: Sample $(mpk, msk) \leftarrow S(1^n)$ and output $pk = mpk, sk = msk$
- $Enc(pk, m)$: Sample a random identity id. Output ciphertext as $(id, E(pk, id, m))$
- $Dec(sk, (id, c))$: Output $D(K(sk, id), c)$



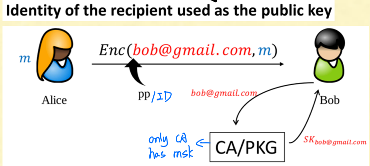
use digital signature to prevent asking unsolveable phase-2 query
- $Gen(1^n)$: Sample $(mpk, msk) \leftarrow S(1^n)$ and output $pk = mpk, sk = msk$
- $Enc(pk, m)$: Sample $(pk_{sig}, sk_{sig}) \leftarrow Gen_{sig}(1^n)$. (strongly unforgeable)
  1. Set $id = pk_{sig}$
  2. $c \leftarrow E(pk, id, m)$
  3. $\sigma \leftarrow Sign(sk_{sig}, c)$
  4. Output ciphertext as (id, c, σ)
- $Dec(sk, (id, c))$: Output $D(K(sk, id), c)$ if $Ver(pk_{sig} = id, c, \sigma) = 1$ and error $\perp$ otherwise.



The ciphertext $c_{new} = (id^*, c', \sigma')$ is such that $(c', \sigma') \ne (c^*, \sigma^*)$ and it needs to be decrypted only if $Vrfy(id^*, c', \sigma') = 1$. Specifically, IBE adversary can safely return $\perp$ if this test is the signature verification fails. However, if the signature verification success then (c', σ') is actually a forgery for the underlying signature scheme.

IBE identity-based encryption

Identity of the recipient used as the public key



- $Setup(1^\lambda) \to (msk, mpk)$.
- $KeyGen(msk, \mathbf{ID}) \to \mathbf{sk_{ID}}$
- $Enc(mpk, \mathbf{ID}, m) \to ct$
- $Dec(\mathbf{sk_{ID}}, ct) \to m$

Properties:
- Correctness: $Dec(\mathbf{sk_{ID}}, Enc(mpk, \mathbf{ID}, m)) \to m$

CCA-Mac. Mac with Verify oracle

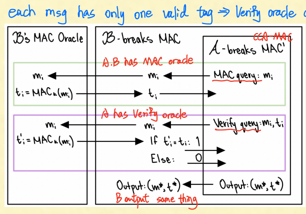
(a) Provide a formal definition of CCA-secure MACs.
- The challenger samples $k \leftarrow Gen(1^n)$.
- The adversary $\mathcal{A}$ is given input 1^n and oracle access to $Mac_k(\cdot)$ and $Verify_k(\cdot,\cdot)$. The adversary eventually outputs a pair (m, t). Let Q denote the set of all queries that $\mathcal{A}$ asked to its $Mac_k(\cdot)$ oracle.
- The output of the experiment is defined to be 1 if and only if (1) $Verify_k(m, t) = 1$ and (2) $m \notin Q$.

Π is a CCA-secure MAC if for all adversaries $\mathcal{A}$,

$$Pr[CCA - Mac - Forge_{\mathcal{A},\Pi}(n) = 1] = negl(n).$$

(b) Assume that Π is a standard MAC with canonical verification:
i) the Mac algorithm is deterministic
ii) the Verify algorithm, on input (m, t), recomputes t' := Mack(m) accepts if t' = t.
Prove that Π also satisfies your definition from part (a)

each msg has only one valid tag ⇒ Verify oracle doesn't help

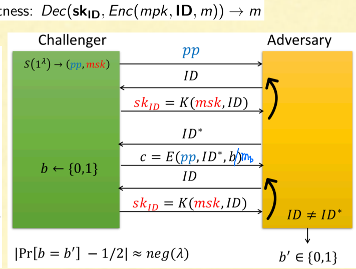


(c) Assume that $\Pi = (Gen, Mac, Vrfy)$ is a standard secure MAC. Construct another MAC $\Pi' = (Gen', Mac', Vrfy')$ that is standard secure, but is *not* secure under your definition from part (a).

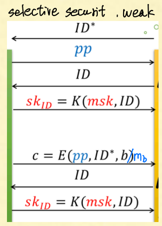
idea: attacker can use Vrfy oracle to learn key k one bit at a time
$Gen' = Gen$
$Mac'_k(m) = (Mac_k(m), 0, 0, 0, 0) = (t, 0, 0, 0, 0)$
$Vrfy'_k(m, (t, t_1, i, b))$: if $t_1 = 0$: { $Vrfy_k(m, t) = 1$ output 1; otherwise output 0 }; if $t_1 = 1$: { $Vrfy_k(m, t) = 1$ output i^{th} bit of key ⇒ b; otherwise output 0 }

CPA

- Challenger runs $Setup(1^\lambda) \to (msk, mpk)$ and sends mpk to the adversary.
- **Keygen Queries: Phase 1** Adversary sends ID to the challenger and gets back $sk_{ID} \leftarrow KeyGen(msk, ID)$ corresponding to the ID.
- **Challenge phase:** Adversary sends a $ID*$ that was not queried as well as messages $m_0 \ne m_1$.
- Challenger picks $b \leftarrow \{0,1\}$ and returns $c_b \leftarrow Enc(mpk, ID^*, m_b)$.
- **Keygen Queries: Phase 2** Adversary sends ID to the challenger and gets back $sk_{ID} \leftarrow KeyGen(msk, ID)$ corresponding to the ID (ID^* *not allowed*).
- Adversary outputs a guess b' for b.



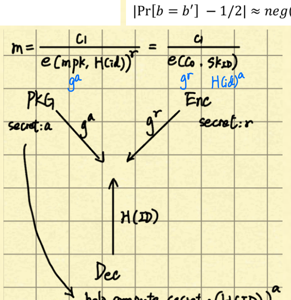
selective securit. weak



CPA IBE construction (DBDH)

Let $H : \{0,1\}^* \to G$ be a hash function
- $S(1^n)$: Output $mpk = g^a$ and $msk = a$
- $K(msk, id)$: Output $sk_{id} = H(id)^a$
- $E(mpk, id, m \in G)$: Sample $r \leftarrow Z_q^*$ and output $c_0 = g^r, c_1 = m \cdot e(mpk, H(id))^r$
- $D(sk_{id}, (c_0, c_1))$: Output $\frac{c_1}{e(c_0, sk_{id})}$

- Correctness: Follows by a simple check
- Security: Given g^a, g^r and $H(id)$, $e(g, H(id))^{ar}$ is indistinguishable from uniform.



Digital signature

- $\mathsf{Gen}(1^n)$: Outputs public key and secret key pair (pk, sk).
- $\mathsf{Sign}_{sk}(m)$: Outputs a signature σ on the message m.
- $\mathsf{Vrfy}_{pk}(m, \sigma)$: Outputs 0/1.

Correctness: For all n, except for negligible choices of (pk, sk), it holds that for all m, $\mathsf{Vrfy}_{pk}(m, \mathsf{Sign}_{sk}(m)) = 1$.

adversary want to forge a valid sign without sk

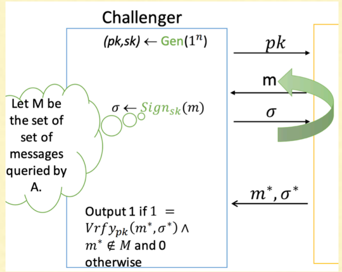
unforgeability

$\mathsf{Forge}_{A,\Pi}(1^n)$

1. Sample $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$.
2. Let (m^*, σ^*) be the output of $\mathsf{Sign}_{sk}(\cdot)$ by adversary $A(pk)$. Let M be the list of queries A makes.
3. Output 1 if $\mathsf{Vrfy}_{pk}(m^*, \sigma^*) = 1 \wedge m^* \notin M$ and 0 otherwise.

$\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ is existentially unforgeable under adaptive chosen message attack if $\forall$ probabilistic polynomial time (PPT) adversary A, it holds that:

$$\Pr[\mathsf{Forge}_{A,\Pi} = 1] \leq \mathsf{negl}(n)$$



CRHF

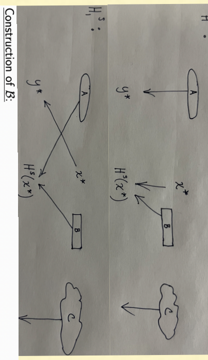
Drop last bit → not CRHF

- Let $\mathcal{H} = (\mathsf{Gen}, H)$ be a CRHF. Let x^* belong to the domain of H^s, and let y^* belong to the range of H^s.
- Next, for any $s \leftarrow \mathsf{Gen}(1^n)$:

$$\text{let } H_1^s(x) = \begin{cases} y^* & \text{if } x = x^* \\ H^s(x^*) & \text{if } x \neq x^* \text{ and } H^s(x) = y^* \\ H^s(x) & \text{otherwise} \end{cases}$$

H_1: drop last bit → no, x^* is collision

- Prove that (Gen, H_1) is a CRHF.

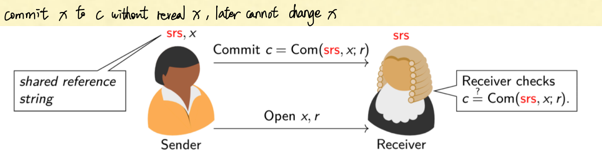


Construction of $\mathcal{B}$:

- In the CRHF game for H_1, the challenger samples $s \leftarrow \mathsf{Gen}(1^n)$ and gives s to the adversary $\mathcal{B}$.
- $\mathcal{B}$ will run $\mathcal{A}$ on input s until $\mathcal{A}$ produces two inputs (x, x'). (H_1 collision)
- $\mathcal{B}$ makes a list of collision candidates: $C := \{(x, x^*), (x, x'), (x', x^*)\}$ and checks whether each candidate $(x_1, x_2) \in C$ satisfies the conditions: $x_1 \neq x_2$ and $H^s(x_1) = H^s(x_2)$.
- $\mathcal{B}$ outputs the first candidate $(x_1, x_2) \in C$ that satisfies the conditions.

Commitment scheme

commit x to c without reveal x, later cannot change x



- **Correctness:** A sender should be able to convince an honest receiver of the correct opening with *overwhelming* probability. (Easy to see)
- **Binding:** No PPT cheating sender can find two openings for the same commitment. That is, $\forall$ PPT $\mathcal{A}$ we have that

$$\Pr[(x, r, x', r') \leftarrow \mathcal{A}(1^\lambda, \mathsf{srs}) \text{ such that } x \neq x' \text{ and } \mathsf{Com}(\mathsf{srs}, x, r) = \mathsf{Com}(\mathsf{srs}, x', r')] = \mathsf{neg}(\lambda)$$

- **Hiding:** The commitment doesn't leak any information about the committed value x. That is, $\forall$ PPT $\mathcal{A}, x, x'$ we have that

$$|\Pr[\mathcal{A}(1^\lambda, \mathsf{srs}, \mathsf{Com}(\mathsf{srs}, x; r)) = 1] - \Pr[\mathcal{A}(1^\lambda, \mathsf{srs}, \mathsf{Com}(\mathsf{srs}, x'; r')) = 1]| \leq \frac{1}{2} + \mathsf{neg}(\lambda)$$

- $\mathsf{Gen}(1^n) \rightarrow \mathsf{params}$
- $\mathsf{Commit}(\mathsf{params}, m; r) = \mathsf{com}$
  - $\mathcal{M}$ is the message space, and $m \in \mathcal{M}$.
  - Other notation: $\mathsf{Commit}(\mathsf{params}, m) \rightarrow \mathsf{com}$
- Open: Committer publishes m and proves that com is a commitment to m. The verifier decides whether to accept or reject the proof.

Canonical Opening Procedure:

- Committer publishes (m, r).
- Verifier checks whether $\mathsf{com} = \mathsf{Commit}(\mathsf{params}, m; r)$. If so, they accept; if not, they reject.

The definition of hiding resembles CPA security.

Hiding-Game$(n, \mathcal{A})$:

1. The challenger samples params $\leftarrow \mathbf{Gen}(1^n)$ and sends params to the adversary $\mathcal{A}$.
2. $\mathcal{A}$ outputs two messages $m_0, m_1 \in \mathcal{M}$.
3. The challenger samples $b \leftarrow \{0, 1\}$ and computes:

$$\mathsf{com}^* \leftarrow \mathsf{Commit}(\mathsf{params}, m_b)$$

They send com^* to $\mathcal{A}$.

4. $\mathcal{A}$ outputs a guess b' for b. The output of the game is 1 if $b' = b$ and 0 otherwise.

The commitment scheme is **computationally hiding** (a.k.a. **hiding**) if for any PPT adversary $\mathcal{A}$,

$$\Pr[\text{Hiding-Game}(n, \mathcal{A}) \rightarrow 1] \leq \frac{1}{2} + \mathsf{negl}(n)$$

The commitment scheme is **statistically hiding** if for any adversary $\mathcal{A}$ *running in unbounded time*,

$$\Pr[\text{Hiding-Game}(n, \mathcal{A}) \rightarrow 1] \leq \frac{1}{2} + \mathsf{negl}(n)$$

The definition of binding resembles collision-resistance.

Binding-Game$(n, \mathcal{A})$:

1. The challenger samples params $\leftarrow \mathbf{Gen}(1^n)$ and sends params to the adversary $\mathcal{A}$.
2. $\mathcal{A}$ outputs two pairs (m_0, r_0) and (m_1, r_1), where $m_0, m_1 \in \mathcal{M}$.
3. The output of the game is 1 if $m_0 \neq m_1$, and

$$\mathsf{Commit}(\mathsf{params}, m_0; r_0) = \mathsf{Commit}(\mathsf{params}, m_1; r_1)$$

Otherwise, the output of the game is 0.

The commitment scheme satisfies **computational binding** (a.k.a. **binding**) if for any PPT adversary $\mathcal{A}$,

$$\Pr[\text{Binding-Game}(n, \mathcal{A}) \rightarrow 1] \leq \mathsf{negl}(n)$$

The commitment scheme satisfies **statistical binding** if for any adversary $\mathcal{A}$ *running in unbounded time*,

$$\Pr[\text{Binding-Game}(n, \mathcal{A}) \rightarrow 1] \leq \mathsf{negl}(n)$$

No commitment scheme can be both statistically hiding and statistically binding.

PRG → Commit

The following construction uses a PRG to construct a commitment scheme.

Let $G : \{0,1\}^n \rightarrow \{0,1\}^{3n}$ be a PRG. Let $m \in \{0,1\} = \mathcal{M}$.

1. $\mathsf{Gen}(1^n)$: Sample $s \leftarrow \{0,1\}^{3n}$ and output params $= s$.
2. $\mathsf{Commit}(\mathsf{params}, m; r)$: Let $r \leftarrow \{0,1\}^n$. Compute

$$\mathsf{com} = G(r) \oplus (m \cdot s)$$

Prove that this construction satisfies computational hiding and statistical binding.

① break hiding → break PRG

Construction of $\mathcal{B}$:

1. - *Pseudorandom Case:* The PRG challenger samples $r \leftarrow \{0,1\}^n$ and sends $g = G(r)$ to $\mathcal{B}$.
   - *Truly Random Case:* The PRG challenger samples $g \leftarrow \{0,1\}^{3n}$ and sends g to $\mathcal{B}$.
2. $\mathcal{B}$ chooses $m_0 = 0$ and $m_1 = 1$ and then samples $b \leftarrow \{0,1\}$.
3. $\mathcal{B}$ computes

$$\mathsf{com}^* = g \oplus (m_b \cdot s)$$

and sends com^* to $\mathcal{A}$.

4. $\mathcal{A}$ outputs a guess b' for b. $\mathcal{B}$ checks whether $b = b'$. If so, $\mathcal{B}$ outputs 0. If not, $\mathcal{B}$ outputs 1.

- *Pseudorandom Case:* If $g = G(r)$ for some random $r \leftarrow \{0,1\}^n$, then $\mathcal{B}$ simulates the hiding security game for the commitment scheme. In this case,

$$\Pr[b = b'] = \Pr[\text{Hiding-Game}(n, \mathcal{A}) \rightarrow 1] \geq \frac{1}{2} + \text{non-negl}(n)$$

- *Truly Random Case:* If $g \leftarrow \{0,1\}^{3n}$, then com^* is independent of b. com^* is basically a one-time pad ciphertext. In this case:

$$\Pr[b = b'] = \frac{1}{2}$$

In summary, $\mathcal{B}$ breaks the PRG security of G because:

$$\Pr[\mathcal{B} \rightarrow 0 | \text{Pseudorandom Case}] - \Pr[\mathcal{B} \rightarrow 0 | \text{Truly Random Case}] \geq \frac{1}{2} + \text{non-negl}(n) - \frac{1}{2} \geq \text{non-negl}(n)$$

② binding

- If the adversary can break binding, then they can find two openings $(0, r_0)$ and $(1, r_1)$ such that

$$G(r_0) = G(r_1) \oplus s$$

- This is only possible if there exist values $(r_0, r_1) \in \{0,1\}^n \times \{0,1\}^n$ such that $G(r_0) \oplus G(r_1) = s$.
- Let T be the set of all the values that $G(r_0) \oplus G(r_1)$ can take:

$$T = \{t \in \{0,1\}^{3n} : \exists (r_0, r_1) \in \{0,1\}^n \times \{0,1\}^n \text{ s.t. } t = G(r_0) \oplus G(r_1)\}$$

- $|T| \leq 2^{2n}$ because there are at most 2^{2n} values of (r_0, r_1).
- Finally, s is sampled uniformly at random from $\{0,1\}^{3n}$. Therefore,

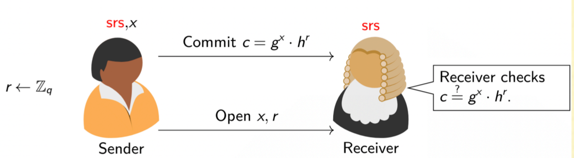
$$\Pr[s \in T] = \frac{|T|}{2^{3n}} \leq \frac{2^{2n}}{2^{3n}} = 2^{-n} = \mathsf{negl}(n)$$

- If $s \notin T$, then no adversary, even a computationally unbounded one, can break binding.

Over the randomness of s, the probability that a computationally unbounded adversary can break binding is $\leq 2^{-n} = \mathsf{negl}(n)$. Therefore, the commitment scheme satisfies statistical binding.

Pederson Commitment Schemes

$$\mathsf{srs} = (G, g, q, h)$$



- **Binding:** Given x, x', r, r' such that $g^x \cdot h^r = c = g^{x'} \cdot h^{r'}$ we can compute $dlog_g(h)$
- **Hiding:** For every $c = g^x h^r$ and x' there exists $r' = r + \frac{x' - x}{dlog_g(h)}$.

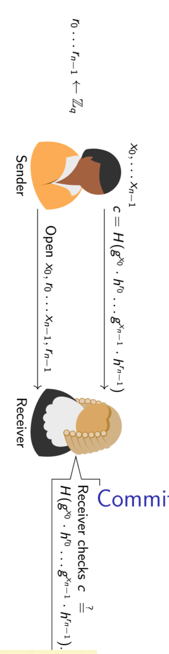
KZG Polynomial Commitment/Pairing Curve BLS12-381

- Gives groups $G_1 = \langle g_1 \rangle, G_2 = \langle g_2 \rangle$ and G_T (of the same prime order p) along with a bilinear pairing operation e.
- For every $\alpha, \beta \in \mathbb{Z}_p^*$, we have that $e(g_1^\alpha, g_2^\beta) = e(g_1, g_2)^{\alpha\beta}$.
- **Setup:** srs generation that supports committing to degree $d - 1$ polynomials:
  - Sample $\tau \leftarrow \mathbb{Z}_p^*$.
  - $\mathsf{srs} = (h_0 = g_1, h_1 = g_1^\tau, g_1^{\tau^2}, \ldots, h_d = g_1^{\tau^{d-1}}, g_2, h' = g_2^\tau)$
- **Commitment:** Given srs and a polynomial $f(x) = c_0 + c_1 x + \ldots c_{d-1} x^{d-1}$ of degree $d - 1$, we can compute $\mathsf{Com}(f)$ as:

$$F = \mathsf{Com}(f) = g_1^{f(\tau)} = \prod_{i=0}^{d-1} h_i^{c_i}$$

- **Opening:** Show that $f(z) = s$. In this case, $g(x) = f(x) - s$ is such that $g(z) = 0$. Or, $x - z$ divides $f(x) - s$.
- Sender computes $T(x) = \frac{f(x) - f(z)}{x - z}$ and sends $W = \mathsf{Com}(T)$.
- **Receiver Accepts if:** $e\left(\frac{F}{g_1^s}, g_2\right) = e\left(W, \frac{h'}{g_2^z}\right)$.

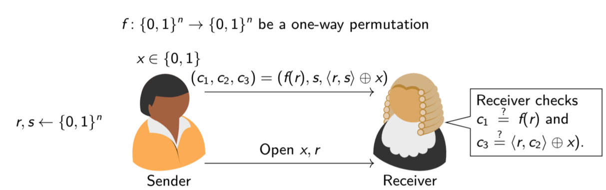
Merkle Commitment Schemes



- **Hashing in More Detail ($n = 2^\ell$):** For every $i \in \{0, n-1\}$, $c_i^\ell = g^{x_i} h^{r_i}$. For all $j \in \{0 \ldots \ell - 1\}, i \in \{0 \ldots 2^j - 1\}$ set $c_i^j = H(c_{2i}^{j+1} || c_{2i+1}^{j+1})$. Finally, $c = c_0^0$.
- **Binding:** An attacker that outputs distinct $x_0, r_0, \ldots x_{n-1}, r_{n-1}$ and $x'_0, r'_0, \ldots x'_n, r'_n$ such that $\exists i$ with $x_i \neq x'_i$ and the receiver checks pass on both can be used to break either (i) CRHF, or (ii) compute $dlog_g(h)$.
- **Hiding:** For every $c_i^\ell = g^{x_i} h^{r_i}$ that is hashed and x'_i there exists $r'_i = r_i + \frac{x'_i - x_i}{dlog_g(h)}$.
- **Partial Opening (Location k):** Opening c_k^ℓ, x_k, r_k and $\forall j \in \{0, \ell\}$ send $c^j_{\lfloor k/2^{\ell-j} \rfloor}$ and $c^j_{\lfloor k/2^{\ell-j} \rfloor + 1}$.

Commitment Scheme From Hardness Concentration

$f \colon \{0,1\}^n \rightarrow \{0,1\}^n$ be a one-way permutation



- **Binding:** Because f is a permutation, given c there is a unique value of r, x such that $c_1 = f(r)$ and $c_3 = \langle r, c_2 \rangle \oplus x$).
- **Hiding:** Follows from the hardness concentration property.

secret sharing

(t,n). split secret s into n piece, one need ≥t piece to reconstruct s

- A dealer takes s as input and uses a sharing algorithm to split the secret s into parts $s_1, \dots, s_n$ to be given to parties $P_1, \dots, P_n$.
- **Correctness:** Any t parties can reconstruct s.
- **Security:** No collusion of $< t$ parties can reconstruct s.

A (t, n)-secret sharing scheme (**Share**, **Reconstruct**) is defined as follows.

- **Share**(s): On input a secret s it outputs shares $s_1, \dots, s_n$.
- **Reconstruct**$(\{s_i\}_{i \in T})$: Outputs s or $\perp$.
- **Correctness**: For any T such that $|T| \geq t$ and secret s we have that **Reconstruct**$(\{s_i\}_{i\in T}) = s$.
- **Security**: For any T such that $|T| < t$, secrets s, s' and adversary A (unbounded) we have that $p = p'$ where (cannot distinguish)

$$p = \Pr[A(\{s_i\}_{i\in T}) = 1 \mid (s_1, \dots, s_n) \leftarrow \textbf{Share}(s)],$$

$$p' = \Pr[A(\{s'_i\}_{i\in T}) = 1 \mid (s'_1, \dots, s'_n) \leftarrow \textbf{Share}(s')].$$

construction. Shamir's

To share $s \in \mathbb{Z}_q$: choose a random degree $t-1$ polynomial $p(x)$ such that $p(0) = s$. Give out the shares $(p(1), \dots, p(n))$.

- Given t shares, we can reconstruct $p(x)$, and can then recover $p(0)$.

Sharing:

- Given a secret $s \in \mathbb{Z}_q$, choose $p(x) = s + a_1x + \dots + a_{t-1}x^{t-1}$, where a_i's are chosen randomly in $\mathbb{Z}_q$. Give out the shares $(p(1), \dots, p(n))$.

Reconstruct:

- Given t values $(i_1, p(i_1)), \dots, (i_t, p(i_t))$, reconstruct p and output $p(0)$.

① ZKP for 3-coloring

- Graph $G = (V, E)$.
- Task: Show a coloring function $c : V \to \{R, B, G\}$ such that such that $\forall (u, v) \in E$, we have that $c(u) \neq c(v)$.

intuition: (R)—(G)—(B) ⇒ permuted graph: (B)—(R)—(G)

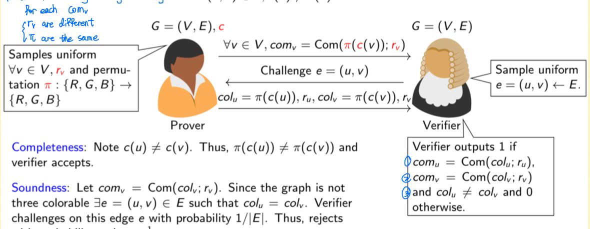
Com₁ = com(B, r₁)
Com₂ = Com(R, r₂)
com3 = com(G, r₃)

verifier only challenge on one edge (2 vertice). permutation prevent him from learning anything

after prover do commit, he cannot make an change

in one round (only challenge one edge)
for each com_v
{ r_v are different
{ π are the same

$\forall (u,v) \in E, c(u) \neq c(v)$



Completeness: Note $c(u) \neq c(v)$. Thus, $\pi(c(u)) \neq \pi(c(v))$ and verifier accepts.

Soundness: Let $com_v = \text{Com}(col_v; r_v)$. Since the graph is not three colorable $\exists e = (u, v) \in E$ such that $col_u = col_v$. Verifier challenges on this edge e with probability $1/|E|$. Thus, rejects with probability at least $\frac{1}{|E|}$

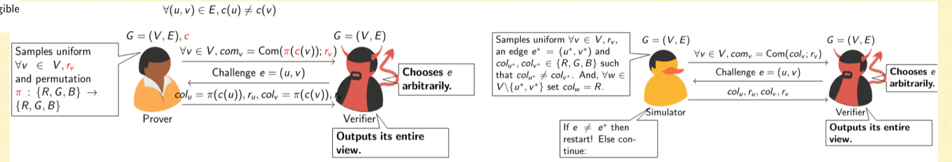
- Repeat the protocol $n|E|$ times.
- A malicious prover succeeds in the i^{th} execution with probability $\leq (1 - \frac{1}{|E|})$.
- A malicious prover succeds in all $n|E|$ execution with probability $\leq \left(1 - \frac{1}{|E|}\right)^{n|E|} \approx e^{-n}$ which is negligible in n.

zero knowledge. only need to prove one round

Simulator: 选一条 edge e*的2个 vertice 给不同 color. 其他全给红色. commit

如果 challenge 的不是 e* 就 restart

$\forall (u,v) \in E, c(u) \neq c(v)$



- The verifier is now malicious and can have arbitrary behavior and output.
- Simulator attempts to generate an indistinguishable output — without the witness's knowledge.
- $\Pr[e = e^*] = 1/|E|$. Furthermore, when this happens, the output of the adversary is indistinguishable from the case with an honest prover. (Note that commitment is hiding.)
- Simulator runs the malicious verifier roughly $|E|$ times to get an output.

② ZKP for hamiltonian cycle

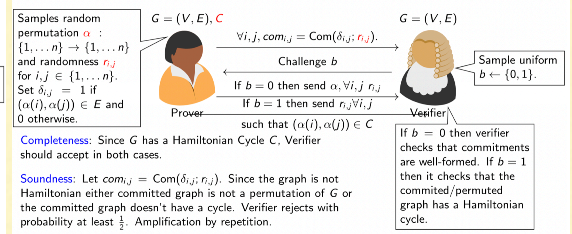
- Graph $G = (V, E)$ with $V = \{1, \dots n\}$.
- Represent as a $n \times n$ matrix M such that $M_{i,j} = 1$ if $(i, j) \in E$ and $M_{i,j} = 0$ otherwise.
- Task: Does these exist a cycle $C \subseteq E$ in G that visits each vertex exactly once?

rename vertice and keep edge consistent, commit permuted graph

if b=0. show that permuted graph is correctly generated.

if b=1. open cycle edges in permuted graph (cycle C knowledge is only used here)

$\exists C \subseteq E$ — a Hamiltonian Cycle in G.



Completeness: Since G has a Hamiltonian Cycle C, Verifier should accept in both cases.

Soundness: Let $com_{i,j} = \text{Com}(\delta_{i,j}; r_{i,j})$. Since the graph is not Hamiltonian either committed graph is not a permutation of G or the committed graph doesn't have a cycle. Verifier rejects with probability at least $\frac{1}{2}$. Amplification by repetition.

Zero-Knoweldge: Cropping Argument. Like Graph 3 Coloring.

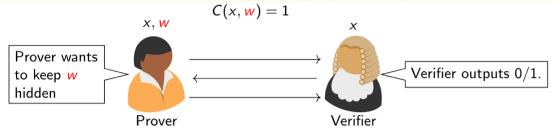
5 / 13

ZKP Zero knowledge proof

C. circuit   w. witness   x. instance

Prover prove he has solution w to problem x. and keep w hidden

$C(x, w) = 1$



- Syntax: Two algorithms, $P(1^n, x, w)$ and $V(1^n, x)$.
- Completeness: Honest prover convinces an honest verifier with *overwhelming* probability.

$$\Pr[V \text{ outputs 1 in the interaction } P(1^n, x, w) \leftrightarrow V(1^n, x)] = 1 - \text{neg}(n)$$

cheat P*. honest V. false w

- Soundness: A PPT cheating prover P^* cannot make a Verifier accept a false statement. For all PPT P^*, x such that $\forall w, C(x, w) = 0$then we have that

$$\Pr[V \text{ outputs 1 in the interaction } P^*(1^n, x) \leftrightarrow V(1^n, x)] = \text{neg}(n)$$

honest P. cheat V*. true w

- Zero-Knowledge: The proof doesn't leak any information about the witness w. $\exists$ a PPT simulator $\mathcal{S}$ that for all PPT V^*, x, w such that $C(x, w) = 1$, we have that $\forall$ PPT D (distinguisher):

V* can only learn as many thing from honest P as simulator S

$$\left|\Pr[D(V^*\text{'s view in } P(1^n, x, w) \leftrightarrow V^*(1^n, x)) = 1] - \Pr[D(\mathcal{S}^{V^*}(1^n, x)) = 1]\right| \leq \text{neg}(n)$$

V* learn from honest P

③ ZKP for Quadratic Residuosity

Q: Come up with a ZKP for Quadratic Residuosity: Consider a modulus m and a w such that $x = w^2 \mod m$. The instance is (x, m) and the witness is the square root of $x \mod m$. x 可能有/没有 square root w

Hint: This is also a three round protocol similar to other protocols you have seen. We only want soundness 1/2 – we can use soundness amplification to make it negligible.

Construction:

1. The prover samples a random $r \in \mathbb{Z}$ and sends $a = r^2 \mod m$ to the verifier.
2. The verifier samples a random bit $b \leftarrow \{0, 1\}$ and sends it.
3. The prover sends $z = w^b \cdot r \mod m$ to the verifer. ($z = w \cdot r$ / r)
4. Verifier accepts if $z^2 = x^b a \mod m$. ($z^2 = x \cdot a$ / a; $x = w^2$, $a = r^2$)

Correctness:

$$z^2 = w^{2b}r^2 = x^b a \mod m$$

Soundness: Suppose there does not exist a square root of x. For the prover to succeed with probability $> 1/2$, the prover should be able to pass the check for both $b = 0$ and $b = 1$ for some choice of first message a. If both checks pass, notice that

$$z_1^2 = a \mod m$$

$$z_2^2 = xa \mod m$$

$$\implies \left(\frac{z_2}{z_1}\right)^2 = x \mod m$$

which is a contradiction.

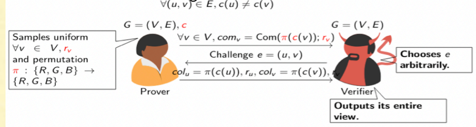
Zero-Knowledge: Idea = Prover can always answer correctly if they know what bit the verifier would pick before they send the first message.
The simulator works like that of Graph Isomorphism (Disc 11).

1. *Sim* samples a random bit b', samples a random $z \mod m$ and computes $a = \frac{z^2}{x^{b'}}$ as the first message.
2. *Sim* runs V^* with a as the first message. If the second message from V^* is the same as b', send z in the third step. Else go to step 1 and start over.

In expectation, *Sim* will need two tries to succeed as V^*'s view is independent of b' after the first message.

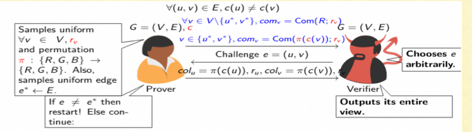
Simulator output is indistinguishable

$\forall (u,v) \in E, c(u) \neq c(v)$



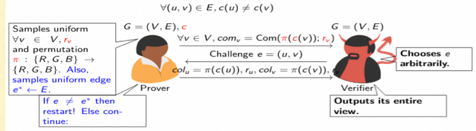
Hybrid H_0.

$\forall (u,v) \in E, c(u) \neq c(v)$

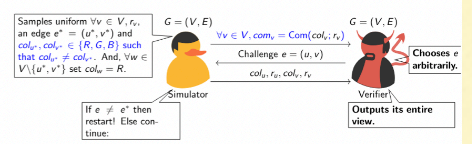


Hybrid H_2. (Indistinguishable from H_1 using the hiding property of the commitment scheme.)

$\forall (u,v) \in E, c(u) \neq c(v)$



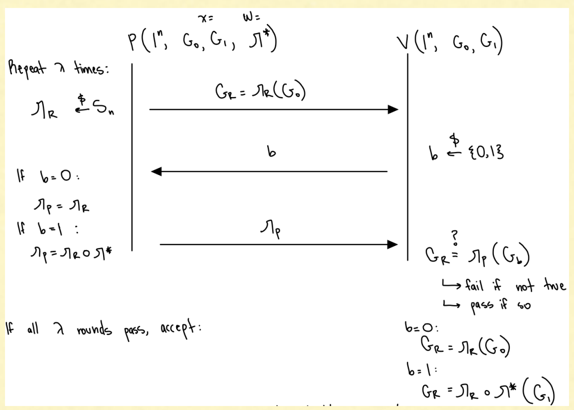
Hybrid H_1.(Information theoretically indistinguishable from H_0. Cropping Argument.)



Hybrid H_3. (Only renaming things from H_3. Not using c anymore.)

③ ZKP for isomorphic graph

Two graphs are **isomorphic** if it is possible to permute the vertices of one graph to obtain the other graph.



Zero-knowledge: The protocol satisfies ZK.

$Sim^{V^*}(1^\lambda, G_0, G_1)$:

Each round:

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

$\pi_P \xleftarrow{\$}$

$G_R = \pi_P(G_b)$

Run V^* on G_R

V^* outputs b

If $b' = b$, Sim sends π_P to V^*

Otherwise, rewinds V^* & restarts

Soundness:

Claim: If $G_0 \not\cong G_1$, P^* will convince V to accept w.p. $\leq 2^{-\lambda}$. Negligible!

On each round $G_R \not\cong$ to both G_0 & G_1 (otherwise $G_0 \cong G_R \cong G_1$, $G_0 \cong G_1$)

W.p. ≥ V picks b s.t. $G_R \not\cong G_b$.

If $G_R \not\cong G_b$, $\nexists$ π_p s.t. $G_R = \pi_p(G_b)$

On each round prob. $\leq \frac{1}{2}$ of accept

λ rounds: $2^{-\lambda}$

Commit , OWF

Show that the existence of a *non-interactive perfectly binding bit commitment scheme* implies a one-way function (10 points).

Recall that a non-interactive perfectly binding bit commitment scheme Com is an algorithm that takes in a bit $b \in \{0,1\}$ and random coins $r \in \{0,1\}^n$, and produces a commitment c. It satisfies the following two properties.

- **Hiding**, which states that no PPT adversary can distinguish between $\mathsf{Com}(0;r)$ and $\mathsf{Com}(1;r)$ for uniformly sampled $r \leftarrow \{0,1\}^n$.
- **Perfect binding**, which states that there do not exist any r_0, r_1 such that $\mathsf{Com}(0;r_0) = \mathsf{Com}(1;r_1)$.

assume we have ability to invert OWF

let $f(x) = \mathsf{Com}(x_1; r = x_2 \cdots x_n)$. A can invert f.

assume perfect binding hold, hiding must break

$\forall \quad r_1. \quad \mathsf{Com}(0;r_0) \neq \mathsf{Com}(1;r_1)$

So inver ↑ and get $x_1 x_2 \cdots x_n$. $x_1 = 0$. must be $\mathsf{Com}(0;r)$

$x_1 = 1$. must be $\mathsf{Com}(1;r)$

thus break hiding

Digi Sig , OWF

Let $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ be a perfectly correct secure digital signature scheme. Perfect correctness states that for any message m,

$$\Pr_{r_{\mathsf{Gen}}, r_{\mathsf{Sign}} \leftarrow \{0,1\}^n, (vk,sk) := \mathsf{Gen}(1^n; r_{\mathsf{Gen}})} [\mathsf{Vrfy}(vk, m, \mathsf{Sign}(sk, m; r_{\mathsf{Sign}})) = 1] = 1,$$

where r_{Gen} are the random coins used by Gen and r_{Sign} are the random coins used by Sign. **Define** $f(x)$ to output the verification key vk output by $\mathsf{Gen}(1^n; x)$. **Show that** f is a one-way function.

Challenger

$(vk, sk) \leftarrow \mathsf{Gen}(1^n; r_{\mathsf{Gen}})$ —pk→

B break digi sig

—pk→ A invert f(x)

invert $vk = \mathsf{Gen}(1^n; x)$

$(vk, sk) \leftarrow \mathsf{Gen}(1^n; r_{\mathsf{Gen}})$ ←x— get $x = r_{\mathsf{Gen}}$

pick m^*

$\sigma^* \leftarrow \mathsf{Sign}(sk, m)$

—m^*, σ^*→

Show that there exists a secure digital signature scheme $(\mathsf{Gen}', \mathsf{Sign}', \mathsf{Vrfy}')$ that is *not* perfectly correct (i.e. it is only correct with probability $1 - \mathsf{negl}(n)$), for which f as defined above is *not* a one-way function (5 points). **Solution:** $\mathsf{Gen}'(1^n; x)$ parses $x = (x_1, x_2) \in \{0,1\}^{2n}$ and outputs x_2 if $x_1 = 0^n$, and otherwise runs $\mathsf{Gen}(1^n; x_2)$. Note that Gen' will only fail to run Gen with negligible probability, but that any image y of $f(\cdot)$ has pre-image $(0, y)$.

ZKP. witness indistinguishable

Let L be a language in NP (e.g. graph three coloring) and let R_L be the NP-relation defined by the language L, meaning that $x \in L$ iff there exists a witness w (e.g. the graph three coloring function) such that $(x, w) \in R_L$. Let (P, V) be an interactive proof system for L. That is, both P and V are initialized with an instance x, the prover P is additionally given a witness w such that $(x, w) \in R_L$, and P attempts to convince V that $x \in L$. They do so by interacting, and we let $\langle P(w), V\rangle(x)$ denote the verifier's *view* of this interaction, consisting of the messages sent back and forth, as well as the verifier's private state.

We say that (P, V) is *witness indistinguishable (WI)* if for all adversarial PPT V^*, all $x \in L$, and all distinct witnesses w_0, w_1 such that $(x, w_0) \in R_L$ and $(x, w_1) \in R_L$, the following two distributions are computationally indistinguishable.

$$\langle P(w_0), V^*\rangle(x) \approx \langle P(w_1), V^*\rangle(x).$$

1. Show that if (P, V) is a computational *zero-knowledge* proof system, then it is also witness indistinguishable (5 points).

   Solution: The zero-knowledge property implies that there exists a simulator Sim such that for any witnesses w, it holds that $\langle P(w), V^*\rangle(x) \approx \mathsf{Sim}(x)$. This in particular means that $\langle P(w_0), V^*\rangle(x) \approx \mathsf{Sim}(x)$ and that $\langle P(w_1), V^*\rangle(x) \approx \mathsf{Sim}(x)$, and the claim follows.

Let (P, V) be a witness-indistinguishable proof system. Define $(\tilde{P}, \tilde{V})$ to repeat (P, V) independently k times *in parallel* (where k is some polynomial), and $\tilde{V}$ accepts if and only if V accepts in all the parallel executions. Show that $(\tilde{P}, \tilde{V})$ is still witness indistinguishable (5 points). **Solution:** We will define a sequence of hybrids where Hyb_i uses the witness w_1 in the first i parallel repetitions and uses the witness w_0 in the rest. Notice that Hyb_0 corresponds to the distribution where w_0 is used in every repetition and Hyb_k corresponds to the distribution where w_1 is used in every repetition. It follows from witness indistinguishability that Hyb_i is computationally indistinguishable to Hyb_{i+1} for all i and the claim follows.

4. Give the formal security definition of a one-time strongly unforgeable signature scheme $(\mathsf{Sig.Gen}, \mathsf{Sig.Sign}, \mathsf{Sig.Vrfy})$ (recall that "one-time" security states that security holds as long as the adversary only sees a signature on a single message, and "strong unforgeablity" states that the adversary cannot even produce a different signature on a message that it queried to its signing oracle) (5 points). **Solution:** The adversary $\mathcal{A}$ is given the public key pk and is able to query the signing oracle on exactly one message m, receiving $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$. Then they output (m', σ') and win if $\mathsf{Vrfy}(\mathsf{pk}, m', \sigma') = 1$ and $(m', \sigma') \neq (m, \sigma)$. Security states that no PPT $\mathcal{A}$ can win except with $\mathsf{negl}(n)$ probability.
5. Show how to use a one-time strongly unforgeable signature scheme to modify the public-key encryption scheme given above so that it achieves CCA2 security (where the adversary is additionally given decryption oracle access after the challenge phase) (5 points). **Solution:** Rather than sampling $v \leftarrow \{0,1\}^n$, sample $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Sig.Gen}(1^n)$ and set $v = \mathsf{vk}$. Then the ciphertext is $(v, y_1, \ldots, y_n, c, \sigma)$, where $\sigma \leftarrow \mathsf{Sig.Sign}(\mathsf{sk}, (y_1, \ldots, y_n, c))$. During decryption, first verify that σ is a valid signature on $(y_1, \ldots, y_n, c)$ under key v, and if not output $\perp$.
6. Prove that your modification of the scheme satisfies CCA2 security (10 points). **Solution:** Let the challenge ciphertext returned to the adversary be $(v^*, y_1^*, \ldots, y_n^*, c^*, \sigma^*)$. Note that as long as $v \neq v^*$ in each of the adversary's phase 2 queries, the argument from part 2 suffices to show CCA2 security. Thus, it remains to handle queries that begin with v^*. For such queries, we know that the remainder of the ciphertext $(y_1, \ldots, y_n, c, \sigma) \neq (y_1^*, \ldots, y_n^*, c^*, \sigma^*)$. Thus, if σ^* is a valid signature on the remainder of the ciphertext under key v^*, the adversary can be used to break the security of the one-time strongly unforgeable signature scheme. Thus, each of these queries can be answered with $\perp$.

CCA-1 . PKE. OWF. hc

An *injective trapdoor function* f is a keyed injective one-way function that can be inverted given a "trapdoor" td associated with the public key k. It consists of the following three algorithms.

- $\mathsf{Gen}(1^n) \to (k, \mathsf{td})$ outputs a key and trapdoor pair.
- $f_k(x) = y$ evaluates the function on input x.
- $f_k^{-1}(\mathsf{td}, y) = x$ inverts the function on output y, given the trapdoor td.

Correctness states that for any $(k, \mathsf{td}) \leftarrow \mathsf{Gen}(1^n)$ and $x \in \{0,1\}^n$, it holds that $f_k^{-1}(\mathsf{td}, f_k(x)) = x$. One-wayness states that for any PPT $\mathcal{A}$,

$$\Pr_{(k,\mathsf{td}) \leftarrow \mathsf{Gen}(1^n), x \leftarrow \{0,1\}^n} [\mathcal{A}(k, f_k(x)) \to x] = \mathsf{negl}(n).$$

We now define a stronger security property for injective trapdoor functions, which we call *correlated input security*. Intuitively, this states that the adversary, given $f_{k_i}(x)$ for each $i \in \{1...n\}$ (where each k_i is sampled independently), cannot recover x. Formally, for any PPT $\mathcal{A}$,

$$\Pr_{\{(k_i,\mathsf{td}_i) \leftarrow \mathsf{Gen}(1^n)\}_{i\in[n]}, x \leftarrow \{0,1\}^n} [\mathcal{A}(k_1, \ldots, k_n, f_{k_1}(x), \ldots, f_{k_n}(x)) \to x] = \mathsf{negl}(n).$$

output 1 bit

Let $h(\cdot)$ be a hard-core predicate for a correlated input secure injective trapdoor function f. Now consider the following public-key encryption scheme.

- $\mathsf{PKE.Gen}(1^n)$ samples $2n$ keys $\{(k_i^0, \mathsf{td}_i^0) \leftarrow \mathsf{Gen}(1^n), (k_i^1, \mathsf{td}_i^1) \leftarrow \mathsf{Gen}(1^n)\}_{i\in[n]}$ and defines the secret key and public key as

$$\mathsf{pk} := ((k_1^0, k_1^1), \ldots, (k_n^0, k_n^1))$$
$$\mathsf{sk} := ((\mathsf{td}_1^0, \mathsf{td}_1^1), \ldots, (\mathsf{td}_n^0, \mathsf{td}_n^1))$$

- $\mathsf{PKE.Enc}(\mathsf{pk}, m)$ for $m \in \{0,1\}$ samples $v \leftarrow \{0,1\}^n$ and $x \leftarrow \{0,1\}^n$ and outputs $\mathsf{ct} = (v, y_1, \ldots, y_n, c)$, where each $y_i = f_{(k_i^{v_i})}(x)$, and $c = h(x) \oplus m$.

k_i^0 k_i^1

1. Prove that correlated input security implies one-wayness (5 points). **Solution:** An adversary that can break one-wayness can be used to break correlated input security as follows. The reduction will forward one of its images (e.g. $f_{k_1}(x)$) to the one-wayness adversary, who returns x with inverse polynomial probability. Since f is injective, this x would be the pre-image for the reduction's challenge.
2. Give a description of $\mathsf{PKE.Dec}(\mathsf{sk}, \mathsf{ct})$ to go along with $\mathsf{PKE.Gen}, \mathsf{PKE.Enc}$ defined above (5 points). **Solution:** $\mathsf{PKE.Dec}(\mathsf{sk}, \mathsf{ct})$ inverts each y_i with $\mathsf{td}_i^{v_i}$ to obtain x_i. If $x_1 = \cdots = x_n = x$, output $m = c \oplus h(x)$.
3. Show that, assuming f is a correlated input secure injective trapdoor function with hard-core predicate $h(\cdot)$, this scheme satisfies CCA1 security (recall that CCA1 security does not give the adversary access to the decryption oracle after the challenge phase) (10 points). **Solution:**

$$\left| \Pr_{x \leftarrow \{0,1\}^n} [\mathcal{A}(1^n, f(x), h(x)) = 1] - \Pr_{x \leftarrow \{0,1\}^n, b \leftarrow \{0,1\}} [\mathcal{A}(1^n, f(x), b) = 1] \right| \leq \mathsf{negl}(n)$$

Challenger — B break hc — A break CCA1 PKE (no need Enc Oracle)

Sample $(k_i, td_i) \leftarrow \mathsf{Gen}$ — k_i →

sample $x \leftarrow \{0,1\}^n$

Calculate $y_i = f_{k_i}(x)$ — y_i →

sample $v^* \leftarrow \{0,1\}^n$

set $k_i^{v_i^*} = k_i$

sample $k_i^{1-v_i^*}, td_i^{1-v_i^*}$

set $pk = ((k_1^0, k_1^1) \cdots (k_n^0, k_n^1))$ — pk →

phase-1 Dec query

let $v_j \neq v_j^*$ (one bit n bit) ← query $ct = (v, y_1 \cdots y_n, c)$

Calculate $x_j = f^{-1}(td_j^{1-v_j^*}, y_j)$

output $m = c \oplus h(x_j)$

$v \neq v^*$ with 1-neg prob

at least one bit different

⇒ we have td for it

50%. $b = h_c(x)$

50%. $b = r$ → b

m^* ← sample $m^* \in \{0,1\}$.

$c = b \oplus m^*$

$ct^* = (v^*, y_1 \cdots y_n, c)$ → ct^*

can distinguish

if $b = h_c(x)$, output 1

if $b = r$, output 0 ← {if $ct^* = \mathsf{Enc}(pk, m^*)$, output 1; ≠ 0}

m phase-2. A knows v^*

query $ct = (v^*, y_1 \cdots y_n, c)$

cannot answer

ZKP

Let (P_0, V_0) and (P_1, V_1) be *public-coin*[1] honest verifier zero-knowledge proof systems for languages L_0 and L_1 respectively. For simplicity, let us assume that there are three messages in both protocols. The first message is from the prover to the verifier, denoted by α_0 (resp. α_1). The second message is just the verifier's random coins and this will be β_0 (resp. β_1) (we will assume that the length of the verifier's random coins is the same for both proof systems and this length will be n). The final message is from the prover to the verifier, denoted by γ_0 (resp. γ_1). Let S_0 and S_1 be the honest verifier zero-knowledge simulators for the two proof systems. Specifically, for both $c = 0$ and $c = 1$, $S_c(\beta_c)$ (for a randomly chosen β_c) outputs (α_c, γ_c) such that $V_c(\alpha_c, \beta_c, \gamma_c) = 1$ and the distribution of $(\alpha_c, \beta_c, \gamma_c)$ is identical to the real protocol.

Now consider the language $L = \{(x_0, x_1) \mid x_0 \in L_0 \text{ or } x_1 \in L_1\}$. Consider the following honest-verifier zero-knowledge protocol for L.[2] The protocol has a few missing parts and your task is to fill in the details. Also, fill in the details of the simulator S of the constructed honest-verifier, public-coin zero-knowledge protocol (P, V).

- **Input.** The prover and verifier gets two statements x_0 and x_1 such that $(x_0, x_1) \in L$. The prover additionally obtains a witness (c, w) such that w is a valid witness for $x_c \in L_c$. Note that x_{1-c} may not be in L_{1-c}.
- $P \to V$: In the first round, prover chooses $\beta_{1-c} \leftarrow \{0,1\}^n$ uniformly at random. It generates the first round message α_c using the prover P_c on input x_c and the witness w. It generates α_{1-c} as **Solution:** $[S_{1-c}(\beta_{1-c})]_1$. It sends (α_0, α_1) to the verifier.
- $V \to P$: In the second round, the verifier sends its random coins $\beta \leftarrow \{0,1\}^n$ to the prover.
- $P \to V$: the last round, the prover generates β_c as **Solution:** $\beta_{1-c} \oplus \beta$

  It generates the third round message γ_c as **Solution:** $P_c(x_c, w, \alpha_c, \beta_c)$.

  It generates γ_{1-c} as **Solution:** $[S_{1-c}(\beta_{1-c})]_2$. It sends $(\beta_0, \beta_1, \gamma_0, \gamma_1)$ to the verifier.
- The verifier accepts if $\beta =$ **Solution:** $\beta_0 \oplus \beta_1$ and $V_0(\alpha_0, \beta_0, \gamma_0) = 1$ and $V_1(\alpha_1, \beta_1, \gamma_1) = 1$.

Give the description of the honest-verifier zero-knowledge simulator for the above protocol.

$S(\beta)$ outputs $(\alpha = (\alpha_0, \alpha_1), \gamma = (\gamma_0, \gamma_1))$ where $\beta_0 \leftarrow \{0,1\}^n$ is chosen uniformly at random,

$\beta_1 =$ **Solution:** $\beta \oplus \beta_0$,

$\alpha_0 =$ **Solution:** $[S_0(\beta_0)]_1$

$\alpha_1 =$ **Solution:** $[S_1(\beta_1)]_1$

$\gamma_0 =$ **Solution:** $[S_0(\beta_0)]_2$

$\gamma_1 =$ **Solution:** $[S_1(\beta_1)]_2$

---

stream cipher : LFSR . trivium . RC4

block cipher : ECB . CBC . OFB . CTR . SPN . Feistel Network . DES . AES

confusion - diffusion

efficient . OFB . CTR

An encryption scheme that allows for the following: given a ciphertext c encrypting a message m and a function f, compute a ciphertext encrypting $f(m)$.

Solution: homomorphic encryption scheme/fully homomorphic encryption scheme

perfectly secure . $|m| \le |k|$

- Let $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a pseudorandom function.
- Let $\mathcal{H} = (\mathsf{Gen}, H)$ be a collision-resistant hash function with key space $\{0,1\}^n$ and input space $\mathcal{X}$, which may be very large. For every key $s \leftarrow \mathsf{Gen}(1^n)$, $s \in \{0,1\}^n$ and $H^s : \mathcal{X} \to \{0,1\}^n$.
- Let $G : \{0,1\}^{2n} \times \mathcal{X} \to \{0,1\}^n$ be defined as follows:

$$G((k, s), x) = F(k, H^s(x))$$

- **Question:** Prove that G is a pseudorandom function.

Let $\underline{\mathsf{Hyb}_0(\mathcal{A}, n)}$ be the PRF security game in which the adversary $\mathcal{A}$ gets query access to G. In particular:

1. The PRF challenger samples $k \leftarrow \{0,1\}^n$ and $s \leftarrow \mathsf{Gen}(1^n)$.
2. The adversary $\mathcal{A}$ gets query access to the following function:

$$G(\cdot) = F(k, H^s(\cdot))$$

3. The adversary outputs a bit b, which is the output of the hybrid.

Let $\underline{\mathsf{Hyb}_1(\mathcal{A}, n)}$ be the same as $\mathsf{Hyb}_0(\mathcal{A}, n)$, except $F(k, \cdot)$ is replaced with a uniformly random function $R_1 : \{0,1\}^n \to \{0,1\}^n$:

1. The PRF challenger samples a function R_1 uniformly at random from the set of all functions mapping $\{0,1\}^n \to \{0,1\}^n$. They also sample $s \leftarrow \mathsf{Gen}(1^n)$.
2. The adversary $\mathcal{A}$ gets query access to the following function:

$$R_1(H^s(\cdot))$$

3. The adversary outputs a bit b, which is the output of the hybrid.

Let $\underline{\mathsf{Hyb}_2(\mathcal{A}, n)}$ be the same as $\mathsf{Hyb}_0(\mathcal{A}, n)$ except $F(k, H^s(\cdot))$ is replaced with a uniformly random function $R_2 : \mathcal{X} \to \{0,1\}^n$:

1. The PRF challenger samples a function R_2 uniformly at random from the set of all functions mapping $\mathcal{X} \to \{0,1\}^n$.
2. The adversary $\mathcal{A}$ gets query access to:

$$R_2(\cdot)$$

3. The adversary outputs a bit b, which is the output of the hybrid.

Prove that for any PPT adversary $\mathcal{A}$,

$$\left|\Pr[\mathsf{Hyb}_0(\mathcal{A}, n) \to 1] - \Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1]\right| \le \mathsf{negl}(n)$$

Prove that for any PPT adversary $\mathcal{A}$,

$$\left|\Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1] - \Pr[\mathsf{Hyb}_2(\mathcal{A}, n) \to 1]\right| \le \mathsf{negl}(n)$$

To prove lemma 3.1:

- $F(k, \cdot)$ is indistinguishable from $R_1(\cdot)$ because F is a pseudorandom function.
- We can treat $H^s(x)$ as just an input to $F(k, \cdot)$ or $R_1(\cdot)$ in hybrids 0 and 1.
- In conclusion, $F(k, H^s(\cdot))$ is indistinguishable from $R_1(H^s(\cdot))$ because $F(k, \cdot)$ is indistinguishable from $R_1(\cdot)$.

To prove lemma 3.2:

- R_1 and R_2 are truly random functions, so $R_1(H^s(\cdot))$ and $R_2(\cdot)$ are also uniformly random in some sense.
- Given query access to $R_1(H^s(\cdot))$ or $R_2(\cdot)$, the adversary cannot tell which of the two functions they are querying, because in either case, every query receives a uniformly random string in response.
- Therefore, Hyb_1 and Hyb_2 are indistinguishable.

To prove lemma 3.2:

- Since H^s is collision-resistant, then the adversary in Hyb_1 will (with overwhelming probability) query the function on inputs that do not collide.
- In response to each distinct query, the adversary will receive a uniformly random string that is independent of the other responses. This is the same distribution of responses that the adversary receives in Hyb_2. Therefore, Hyb_1 and Hyb_2 are indistinguishable.

MAC ① CPA . not CCA , $Mac'(k,m) = C(Mac(\cdot,\cdot,\cdot))$; $Vrfy'(C(k,m,(t,\beta,i,b)))$: if $\beta=0$ return $Vrfy(k,m,t)$; if $\beta=1$ output ith bit of key $\Rightarrow b$ if $Vrfy(k,m,t)=1$

MAC secure . not strongly secure . $Mac(k,m) = (Mac(m), \text{random bit})$

② Insecure . attack : $m_1 \| m_2 \Rightarrow$ query $m_1 \| m_2 \Rightarrow$ test $m_2 \| m_1$; Q_1 leak t_1 ; Q_2 leak t_2 ; $m_1 \oplus m_2 \Rightarrow m_1 \oplus m_1 = 0$

③ non-PRF : $F'(k,m) = F(k,m) \| m$.

CRHF ① → not CRHF; drop last bit . when $x = x^*$. $y = y^* = H(x_0)$ with last bit flipped

CRHF ⊕ CRHF . $H(x) = (x_1, H^s(x_2 \cdots x_{2n}))$. $(0, x')$, $(1, x')$

OWF ① non-OWF : $f(x) = 0$; non-hc : $f(x) = f(x_{-i}), i, x_i$

② Insecure : PRP F_k^{-1} attack

③ not OWF : OWF ⊕ OWF . $(g_1, 0^{\frac{n}{2}}) \oplus (0^{\frac{n}{2}}, g_2) = (g_1, g_2) = g$; if $(\cdot, 0^{\frac{n}{2}})$; if $(0^{\frac{n}{2}}, \cdot)$

drop last bit $\begin{cases} h(x) \| 0^{\frac{n}{2}} \\ x \| 0^{\frac{n}{2}-1} \| 1 \end{cases}$

$f(x) \oplus x$. $0^{\frac{n}{2}} \| h(x_0)$ ⊕ (x_0, x_1) ; $f(x)$; $0^{\frac{n}{2}} \| h(0^{\frac{n}{2}})$