Introduction to Provable Security

Foundation of Cryptography
Pei-yih Ting
NTOUCS

Familiar Public Key Schemes

- **RSA**: 1978
  - Key Generation: PK=(n, e) SK=d
    - Choose large prime numbers p, q, $n = p\cdot q$, $\Phi(n) = (p-1)\cdot(q-1)$
    - Choose integer e s.t. gcd(e, Φ) = 1, calculate d such that $e \cdot d \equiv 1 \pmod{\Phi}$
  - **Enc**(PK, m): $c \equiv m^e \pmod{n}$, **Dec**(SK, c): $m \equiv c^d \pmod{n}$
  - **Sign**(SK, m): $\sigma \equiv m^d \pmod{n}$, **Verify**(PK, m, σ): $\sigma^e \equiv m \pmod{n}$
- **ElGamal**: 1985
  - Key Generation: PK=(p, g, y), SK=x
    - prime p, $p = 2\,q + 1$, where q is also prime, a generator g' of Z_p , generator of G_q $g \equiv_p g'^2$, choose a secret integer x in Z_q, and calculate $y \equiv_p g^x$
  - **Enc**(PK, m): $r \in_R Z_q$, $u \equiv_p g^r$, $v \equiv_p y^r \cdot m$, **Dec**(SK, c): $m \equiv_p v \cdot u^{-x}$
  - **Sign**(SK, m): $k \in_R Z_q$, $r \equiv_p g^k$, $s \equiv_q k^{-1}(m-rx)$
    Verify(PK, m, σ): $g^m \equiv_p y^r \cdot r^s$
- Neat practical schemes, based on the difficulties of the *integer factoring problem* and the *discrete logarithm problem* respectively.

Familiar Schemes (cont'd)

- **Questions**:
  - Are they secure?
  - What do you mean by "secure"?
  - Are they secure unconditionally or under any condition?
  - Which one is better?
  - What is the primitive underneath?
- **Brief answers**:
  - RSA ciphertext hides the message s.t. reconstruction of m is hard
  - ElGamal encryption is IND-CPA s.t. "no info" about m is leaked
  - Forging valid RSA signature is easy, but not for specified message
  - Security of ElGamal signature? **quality / feature of scheme**
  - All the above depend on the ***definitions of security*** and are conditional on some ***computational assumptions***. **adversary**
  - Basic primitive for security protocols is OWF

Familiar Schemes (cont'd)

- RSA encryption: $c \equiv m^e \pmod{n}$
  - Secure if only a complete compromise of m, given c, n, e, is considered a security breach
  - Insecure if any partial information (e.g. Jacobi symbol) derived from m is considered a security breach
- RSA signature: $\sigma \equiv m^d \pmod{n}$
  - Secure if only forgery of the signature of an arbitrarily specified message is considered a security breach
  - Insecure if an existential forgery is considered a security breach
- ElGamal encryption: $r \in_R Z_q$, $u \equiv_p g^r$, $v \equiv_p y^r \cdot m$,
  - Secure if only distinguishing two adversary specified messages under chosen plaintext attack is considered a security breach
  - Insecure if only distinguishing two adversary specified messages under chosen ciphertext attack is considered a security breach

Encryption Security

- **Total break**
  - The adversary can determine the private key of a PKE or the secret key of a symmetric key encryption system.
- **Partial break**
  - The adversary can decrypt a previously unseen ciphertext (without knowing the private/secret key) or determine some interesting information about the plaintext given the ciphertext.
  - In some cryptosystems, partial information about the plaintext may be leaked by the ciphertext.
    e.g. The Jacobi symbol of the RSA plaintext.
    $c \equiv m^e \pmod n$, $\gcd(e, \phi(n))=1$, e must be odd
    $$\left(\frac{c}{n}\right) = \left(\frac{m}{n}\right)^e = \left(\frac{m}{n}\right)$$
- **Semantic Security** or **Polynomial Security**:
  - Whatever can be computed from the ciphertext can also be computed without it. Goldwasser & Micali 1984
  - A deterministic encryption scheme does not provide semantic security. e.g. plain RSA and a finite message space

Encryption Security (cont'd)

- **IND: Message Indistinguishability** (Ciphertext Indistinguishability): Given a ciphertext c from two possible messages m_0, m_1, it is computationally difficult to determine which one is actually hidden.
- **Non-malleability**: Given an encryption of a plaintext m, it is impossible to generate another ciphertext which decrypts to $f(m)$, for a known function f, without necessarily knowing or learning m
  e.g. RSA, ElGamal, Paillier, $m \oplus G(sk)$ are malleable
- **Plaintext awareness**: A cryptosystem is plaintext-aware if it is difficult for any efficient algorithm to come up with a valid ciphertext without being aware of the corresponding plaintext.
- **Adversary Resources**:
  Ciphertext Only Attack
  Known Plaintext Attack
  Chosen Plaintext Attack (CPA)
  Non-adaptive Chosen Ciphertext Attack (CCA1, Lunch-time Attack)
  Adaptive Chosen Ciphertext Attack (CCA2)

Relations among PKE Security Notions

$$\begin{tikzcd}
\text{NM-CPA} \arrow[d, "3.1"'] \arrow[dr, shift left, "3.6"', "/" marking] & \text{NM-CCA1} \arrow[l] \arrow[r, shift left, "3.7", "/" marking] \arrow[d, "3.1"] & \text{NM-CCA2} \arrow[l, shift left] \arrow[d, shift right, "3.1"'] \\
\text{IND-CPA} & \text{IND-CCA1} \arrow[l] \arrow[ul, shift left, "3.5"', "/" marking] & \text{IND-CCA2} \arrow[l] \arrow[u, shift right, "3.3"']
\end{tikzcd}$$

polynomial security, semantic security

Implication

$A \Rightarrow B$: A proof that if a public key encryption scheme meets notion of security A then this scheme also meets notion of security B

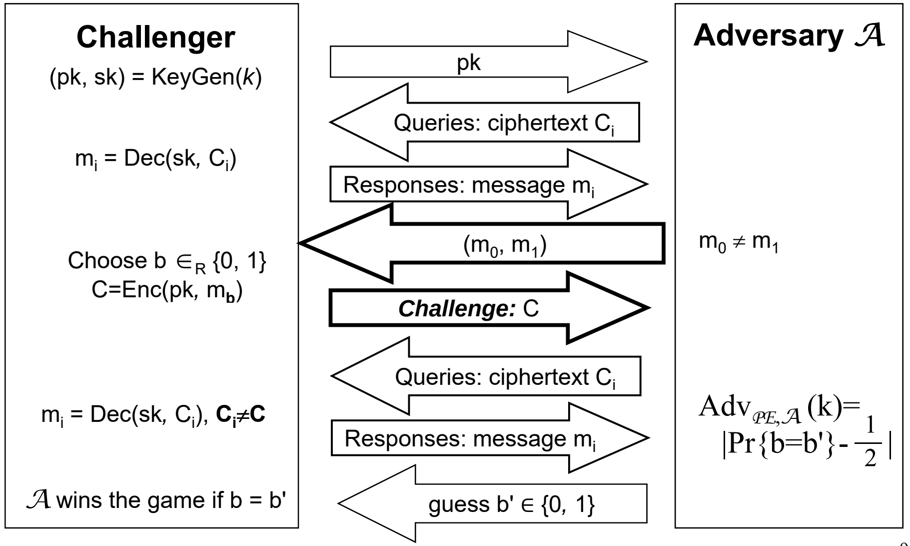
Separation

$A \not\Rightarrow B$: There exists a public key encryption scheme that provably meets notion of security A but provably does not meet notion of security B

Public Verifiable Signature Security

- **Total break**: key recovery
- **Universal forgery**: finding an efficient equivalent algorithm to produce signatures for arbitrary messages
- **Selective forgery**: forging the signature for a particular message chosen a priori by the attacker
- **Existential forgery**: forging at least one signature
- **Adversary Resources**:
  - **Key-only attack**: no-message attacks
  - **Known-message attack**
  - **Generic chosen-message attack**: non-adaptive, messages not depending on public key
  - **Directed chosen-message attack**: non-adaptive, messages depending on public key
  - **Adaptive chosen-message attack**: messages depending on the previously seen signatures

IND-CCA2-Game



Message Indistinguishability

Definition: IND-CPA, IND-CCA1, IND-CCA2

let $\mathcal{PE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme

$\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary

for atk $\in$ {CPA,CCA1,CCA2} and k $\in$ N, let the advantage

$$Adv^{ind\text{-}atk}_{\mathcal{PE},\mathcal{A}}(k) = |\Pr\{Exp^{ind\text{-}atk\text{-}1}_{\mathcal{PE},\mathcal{A}}(k) = 1\} - \Pr\{Exp^{ind\text{-}atk\text{-}0}_{\mathcal{PE},\mathcal{A}}(k) = 1\}| < 1/p(k)$$

where for b $\in$ {0,1},

Experiment $Exp^{ind\text{-}atk\text{-}b}_{\mathcal{PE},A}(k)$

$(pk, sk) \xleftarrow{R} \mathcal{K}(k)$; $(x_0, x_1, s) \leftarrow \mathcal{A}_1^{O_1(\cdot)}(pk)$; $y \leftarrow \mathcal{E}_{pk}(x_b)$;

return $d \leftarrow \mathcal{A}_2^{O_2(\cdot)}(x_0, x_1, s, y)$

If atk = CPA then $O_1(\cdot) = \varepsilon$ and $O_2(\cdot) = \varepsilon$

If atk = CCA1 then $O_1(\cdot) = \mathcal{D}_{sk}(\cdot)$ and $O_2(\cdot) = \varepsilon$

If atk = CCA2 then $O_1(\cdot) = \mathcal{D}_{sk}(\cdot)$ and $O_2(\cdot) = \mathcal{D}_{sk}(\cdot)$

CCA is stronger than CPA

- The encryption engine in CPA is free for a PKE. A CCA attack is given both encryption engine and decryption engine.
- Chosen ciphertext is more favorable to the adversary for an IND game
  - Choose c_0, c_1 far away and decrypt to m_0, m_1, use them as the first message, hopefully $c = E(m_b)$ would be easy to distinguish
- CCA2 attack on an IND-CPA homomorphic scheme is easy
  - Let the challenge ciphertext $c = E(m_b)$.
  - Choose a random r. Calculate $c' = E(r) \cdot c = E(r \cdot m_b)$, $c' \neq c$
  - Ask the decryption engine to decrypt c' and obtains $m_b = D(c')/r$

EUF-CMA

- GMR'86: S. Goldwasser, S. Micali, and R. Rivest, "A digital signature scheme secure against adaptive chosen-message attacks," SIAM J. Computing, pp.281-308, 1988
- A signature scheme $\mathcal{S}$ = (Gen, Sign, Ver) is existentially unforgeable under an adaptive chosen message attack (EUF-CMA) if it is infeasible for a forger who only knows the public key to produce a valid (message, signature) pair, even after obtaining polynomially many signatures on messages of his choice from the signer.
- Formally, $\forall$ PPT forger algorithm $\mathcal{F}$, $\forall$ positive polynomial $p(\cdot)$, $\forall$ sufficiently large n,

$$\Pr\left\{\begin{array}{l} (pk, sk) \leftarrow Gen(1^k); \\ \text{for } i=1,\ldots,n \\ \quad M_i \leftarrow \mathcal{F}(pk, M_1, \sigma_1, \ldots, M_{i-1}, \sigma_{i-1}); \ \sigma_i \leftarrow Sign(sk, M_i); \\ (M, \sigma) \leftarrow \mathcal{F}(pk, M_1, \sigma_1, \ldots, M_n, \sigma_n), \\ \qquad M \neq m_i \text{ for } i=1,\ldots,n, \text{ and } Ver(pk, M, \sigma)=1 \end{array}\right\} < 1/p(n)$$

Conditional Security

- Every **practical and provably secure** public/private key scheme is only secure under specific computational assumptions. e.g.
  - Rabin cryptosystem is secure if "integer factorization assumption (IFA)" holds
  - RSA cryptosystem is secure if "RSA assumption" holds **for target adv.'s**
  - ElGamal encryption is IND-CPA if "decisional Diffie-Hellman assumption (DDH)" holds **for target adversaries**
- The NP problem (OWF) behind every public key cryptosystem

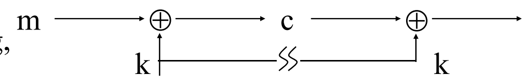
  Given the public key **PK**, there exists a unique matching secret key **SK**, but no polynomial time algorithm can uncover it.
- Provably secure $\mathcal{SE}$ (PRNG+OTP) is far less efficient than AES/DES
- Root computational assumption: **NP ≠ P** (weakest)
- RSA assumption ⇒ IFA ⇒ NP ≠ P

  DDHA ⇒ CDHA ⇒ DLA

  strongest adversary
- While addressing the security of a cryptosystem, we need to specify the **weakest assumption** possible (probably not the OWF hiding SK)

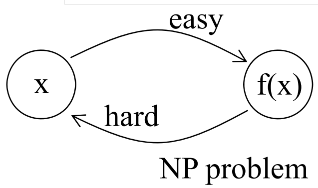
Unconditional Security

- **Information-theoretically secure**: **Perfect secure** or **Shannon secure**: the highest level of security for any scheme, no matter how large the computation power the adversary has, she cannot obtain any information from the ciphertext more than the a-priori information, no computational assumption
  - Transmitting one random bit: can you encrypt the message such that an adversary guess the message with success probability less than 1/2?
- Necessary condition: the key must be longer than the message, must be symmetric key encryption, not practical
- Example: one-time pad SMPC from secret sharing, inefficient



- **Perfect secrecy**: the distribution of ciphertext is independent of the encrypted message
- **Shannon secrecy**: the conditional entropy of the message given the ciphertext is the same as the entropy without the ciphertext

One-Way Function



- Easy: f is a polynomial time computable function
- Hard: for all poly-time probabilistic TM, probability to successfully invert the function is small
- For every probabilistic poly-time TM A',

  every positive polynomial $p(\cdot)$ and all sufficient large n

$$\Pr\{A'(f(U_n), 1^n) \in f^{-1}f(U_n)\} < 1 / p(n)$$

- Possible candidates:

  integer multiplication $n=p\cdot q$ is easy, factoring n is hard

  discrete logarithm $y = g^x \pmod p$ is easy, $x = \mathrm{dlog}_g\, y$ is hard
- Practical symmetric key schemes emulates OWFs (PRF or PRNG)
- OWF ….. Private key cryptography
- TDF …. Public key cryptography (designing PKE is to find trapdoors)

Common Computational Assumptions

on target adversaries

- NP ≠ P
- Existence of OWF, OWP, OWTP
- Integer Factoring: given n = p q, find p, q
- Discrete Logarithm: given $y \in Z_p$, find x s.t. $y \equiv_p g^x$
- Square Root Extraction: given n=pq, $y \in Z_n$, find x s.t. $y \equiv_n x^2$
- RSA (Root Extraction): given n=pq, e, $y \in Z_n$, find x s.t. $y \equiv_n x^e$
- Computational Diffie-Hellman: given g, g^x, g^y, find g^{xy}
- Decision Diffie-Hellman: given g, g^x, g^y, Z, determine if $Z \equiv_p g^{xy}$
- Quadratic Residue: given n=pq, x, determine if $x \in QR_n$
- Composite Residue: given n=pq, $y \in Z_{n^2}$, decide if $\exists\, x \in Z_{n^2}$ s.t. $y \equiv_{n^2} x^n$
- Bilinear Diffie-Hellman: given g, g^x, g^y, $g^z \in G$, find $e(g,g)^{xyz} \in G_T$
- Bilinear Decision Diffie-Hellman: given g, g^x, g^y, $g^z \in G$, and $W \in G_T$, decide if $W = e(g,g)^{xyz}$

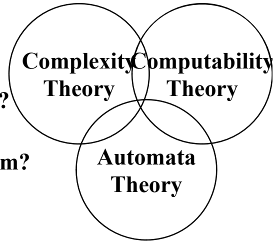
Computation Theory

Complexity Theory: building models for target adversaries

<u>**central problem**</u> **"What makes some problems computationally hard and others easy?"**

<u>**major achievements**</u>

1. **Schemes for classifying problems of different computational difficulties**
2. **Options in confronting a difficult problem**
   - **What is the most difficult part of a problem? Can we alter this part to avoid that problem?**
   - **Are there sub-optimal or heuristic solutions to a problem?**
   - **What kind of instance of a problem is hard?**
   - **Is there a randomized computable algorithm for a problem?**



Computability Theory:

<u>**central problem**</u> **"What is computable? What is not computable? in what model?"**

<u>**major achievements**</u>

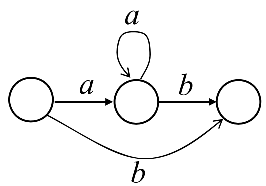
1. **Theoretical models of computers (ex. LBA, DTM, NTM, …)**
2. **Classify problems as solvable or non-solvable**

Automata Theory: definitions and properties of mathematical models of computation

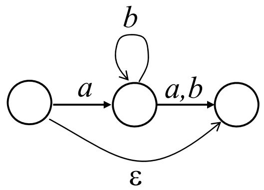
- **Finite automata: text processing, compilers, H/W design**
- **Push down automata: programming language, artificial intelligence**

Finite Automata

- Deterministic Finite Automata (DFA):
  - $M = (Q, \Sigma, \delta, s, F)$
    Q: $\{q_0, q_1, \ldots, q_{m-1}\}$ finite set of states
    Σ: alphabet
    s: start state
    F: set of final states
    $\delta: Q \times \Sigma \rightarrow Q$, transition function

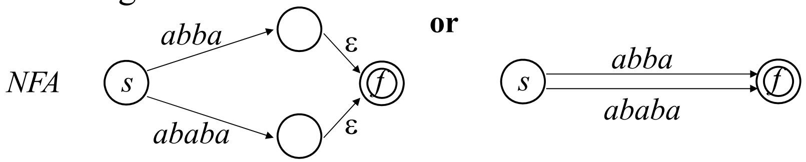


- Non-deterministic Finite Automata (NFA):
  - $M = (Q, \Sigma, \Delta, s, F)$
    Q: $\{q_0, q_1, \ldots, q_{m-1}\}$ finite set of states
    Σ: alphabet
    s: start state
    F: set of final states
    $\Delta: Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow 2^Q$, transition function

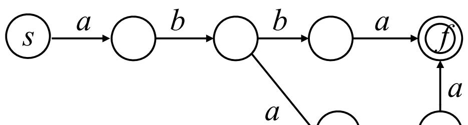


Example

- Design an NFA accepting strings with "abba" or "ababa" substrings.



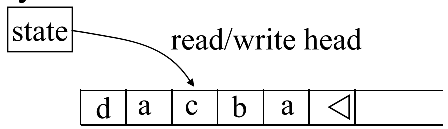
- An NFA can always be converted into a DFA.
- We can design an NFA first, then convert it into an equivalent DFA.



An equivalent DFA

Turing Machine

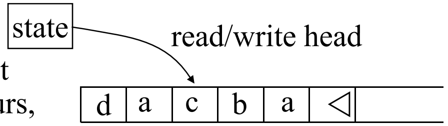
- Complexity / Computability is defined w.r.t. a certain model of computation



- Turing Machine
  - Alan Turing, 1936
  - Similar to finite automaton but with an unlimited and unrestricted memory
  - Formally, a 7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$
    1. Q is the set of states
    2. Σ is the input alphabet not containing the special blank symbol ◁
    3. Γ is the tape alphabet, where $\triangleleft \in \Gamma$ and $\Sigma \subseteq \Gamma$
    4. $\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ is the transition function
    5. $q_0 \in Q$ is the initial state
    6. $q_{accept} \in Q$ is the accept state
    7. $q_{reject} \in Q$ is the reject state, where $q_{reject} \neq q_{accept}$

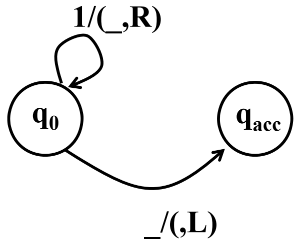
Turing Machine (cont'd)

- TM computes as follows:
  - M's input $w = w_1 w_2 \ldots w_n \in \Sigma^*$ on the leftmost n squares of the tape, the rest of the tape are blanks ◁ (the first ◁ marks the end)
  - Initial state is q_0
  - read/write head starts on the leftmost square
  - Computation proceeds according to the transition function δ
  - If M tries to move its head to the left off the left hand end of the tape, the read/write head stays at the same place for that move
  - The computation continues until it enters either the accept or reject state. If neither occurs, M goes on forever.



Example

- TM:
  - $Q=\{q_0, q_{accept}, q_{reject}\}$
  - $\Sigma=\{1\}$
  - $\Gamma=\{1,\_\}$
  - $\delta(q_0,1)=\{(q_0,\_,R)\}$
  - $\delta(q_0,\_)=\{(q_{accept},L)\}$



- Evaluation tableau (input 11)

#	q_0	1	1	_	#
#	_	q_0	1	_	#
#	_	_	q_0	_	#
#	_	q_{acc}	_	_	#

DTM vs. NTM

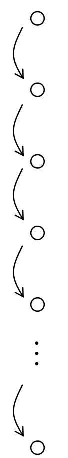
- Deterministic Turing Machine: at any time, a DTM knows its next configuration (the state, the tape head, the tape content) for sure; a single configuration specified by its transition function

  $\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$
- Non-deterministic Turing Machine: at each moment, an NTM has several choices to proceed as the next configurations. i.e. the range of the transition function is modified to be a set:

  $\delta: Q \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma \times \{L, R\}) \setminus \phi$
  - NTM has two equivalent evaluation ways if you only consider the capability:
    - Process in a massively parallel fashion
    - Process in a probabilistic fashion (seems much slower)

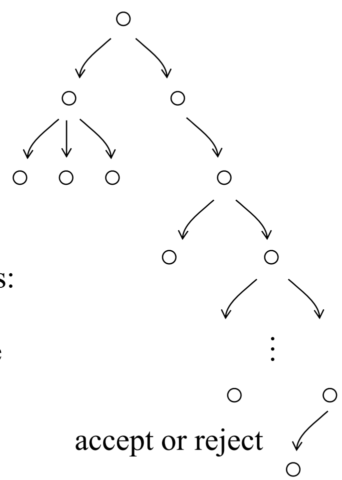
  The parallel one defines a language $L \in$ NP if it accepts $x \in L$ in polynomial time. The probabilistic one also defines NP if it accepts $x \in L$ in polynomial time with non-zero probability. The probabilistic one also defines a language $L \in$ BPP if it accepts $x \in L$ in polynomial time with correct probability bounded away from 0.5. Security professionals surely believe that BPP is a strict subset of NP.
- NTM (with time $O(p(n))$) can be proven to be equivalent to DTM (with time $O(k^{p(n)})$, where $k = \max |\mathcal{P}(Q \times \Gamma \times \{L, R\})|$)

Deterministic vs. Nondeterministic



accept or reject

Church-Turing hypothesis: Deterministic TM is equivalent to our intuitive notion of algorithms.



Note that an NTM decider halts on all branches.

Complexity Classes

- ✧ P: polynomial
  - ✶ problems that can be solved by an algorithm (TM) with computation complexity $O(p(n))$
    ex. Bubble sort $O(n^2)$ Quick sort $O(n \log n)$
  - ✶ there are many problems which are not P

ex. 2^n	knapsack (subset sum)
$n!$	traveling Salesman Problem (TSP)
unsolvable	halting problem

- ✧ NP: non-deterministic polynomial
  - ✶ decision problems that can be decided by an NTM
  - ✶ problems that have solutions (witnesses) which can be verified by a polynomial time algorithm.   ex. Decision versions of Fact, dLog, TSP, Satisfiability (SAT), knapsack...

Complexity Classes (cont'd)

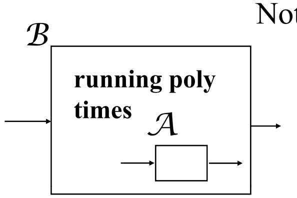
- ✧ NP-complete: the set of the hardest problems in NP
  - ✶ Def 1: NP problems, to which SAT can be reduced
  - ✶ Def 2: NP problems, all NP problems can be reduced to them
  - ✶ ex. SAT, TSP, G3C, Knapsack ...
- ✧ NP-hard: at least as hard as the hardest problems in NP
  - ✶ not limited to decision problem, not necessarily NP, all NP problems can be reduced to them, includes many search problems and optimization problems
  - ✶ ex. halting problem (undecidable), the solution cannot be verified in poly time, Shortest Vector Problem, Closest Vector Problem, Search version of TSP
- ✧ NP-complete = NP-hard $\cap$ NP

Standard (Plain) Security Model

- ✧ Reduce a **simple** problem (structurally simple, well analyzed but believed hard and unsolved problem) to a **complex** problem (the target protocol / cryptosystem).

Ex.   Fact $\leq_T$ Rabin Cryptosystem

"If there exists a PPT adversary $\mathcal{A}$ that breaks the target protocol, then ***using $\mathcal{A}$ as a blackbox***, we construct an algorithm $\mathcal{B}$ that breaks the simple but commonly believed hard problem"



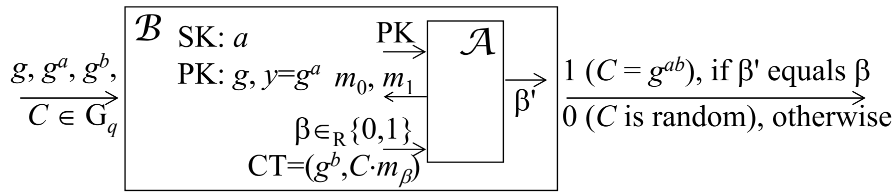
proof by contradiction

Note:
1. Fact $\leq_T$ breaking RSA   is probably false
   breaking RSA $\leq_T$ Fact   is trivial
2. $\mathcal{A}$ should be a blackbox because it is just an assumed entity, nobody knows its interior design. $\mathcal{B}$ only needs to supply $\mathcal{A}$ all necessary inputs including the oracles).

ElGamal is IND-CPA

- ✧ primes p, q, $p = 2q + 1$, a generator g' of Z_p, calculate a generator of G_q as $g \equiv_p g'^2$, i.e. G_q is QR_p, choose a secret key x in Z_q, and calculate the public key $y \equiv_p g^x$
- ✧ **Enc**(PK, m): $r \in Z_q$, $u \equiv_p g^r$, $v \equiv_p y^r \cdot m$, **Dec**(SK, c): $m \equiv_p v \cdot u^{-x}$

- ✧ IND-CPA under DDH assumption

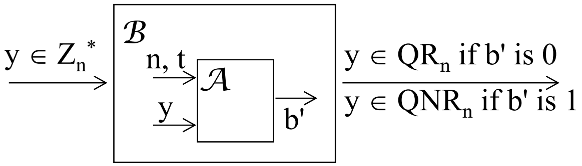


- ✧ DDH tuple: (g, g^a, g^b, g^{ab}) RAND tuple: (g, g^a, g^b, g^c)
- ✧ $\text{Adv}_{\mathcal{B}} = |\Pr[\mathcal{B}(\text{DDH})=1] - \Pr[\mathcal{B}(\text{RAND})=1]|$
  $= |\Pr[\mathcal{A}(\text{PK,CT})=\beta \mid \text{DDH}] - \Pr[\mathcal{A}(\text{PK,CT})=\beta \mid \text{RAND}]|$
  $= |\Pr[\mathcal{A}(\text{PK,CT})=\beta \mid \text{DDH}] - 1/2| \geq |(1/2 + 1/p(n)) - 1/2|$

Goldwasser-Micali is IND-CPA

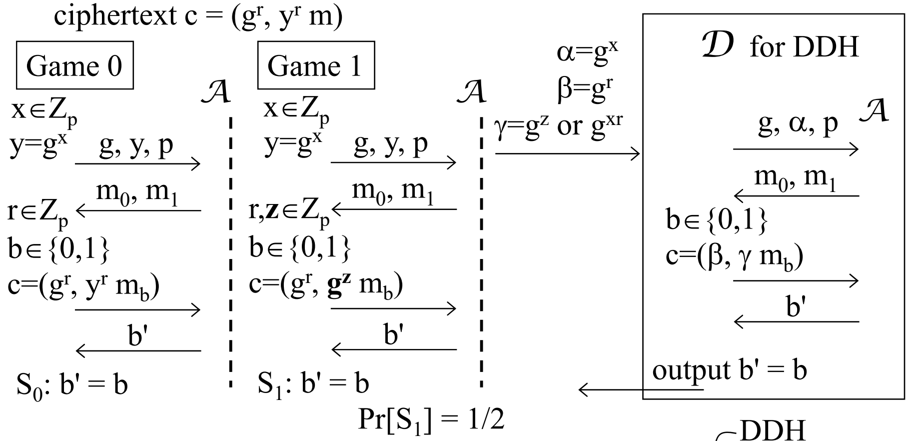
- S. Goldwasser and S. Micali, "Probabilistic encryption," JCSS'84, pp.270-299, 1984

- Choose two large prime numbers p, q, $n = p \cdot q$, choose $t \in_R QNR_n$
- $Enc(b) = r^2 \cdot t^b \pmod n$, where $r \in_R Z_n^*$
- IND-CPA under QRA



Seq. of Game Proof: ElGamal

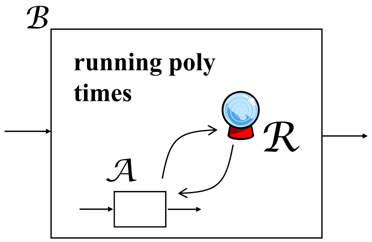
- IND-CPA, assumption: DDH
- ElGamal Encryption: PK: g, p, $y = g^x \pmod p$, SK: x
  ciphertext $c = (g^r, y^r\, m)$



$$| Pr[S_0] - Pr[S_1] | = | Pr[\mathcal{D}(g^{xr})=1] - Pr[\mathcal{D}(g^z)=1] | < 1/p(n)$$

Random Oracle Model

- In the random oracle model, all the settings for standard security model are kept the same except that all parties are modeled as **oracle machines** that operate with the help of a **random oracle**.



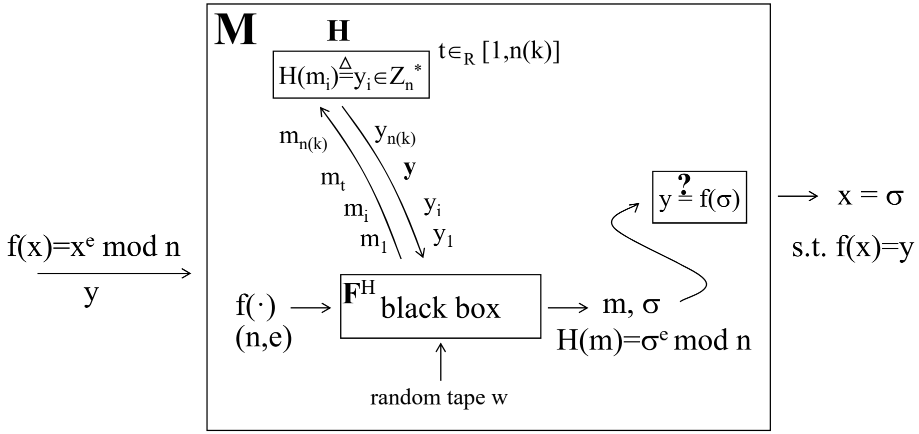
- **Random Function**:
  - The query-response mapping ismodeled as a random function $f(\cdot)$.
  - f(x) can only be obtained by asking the oracle.
- **Programmability**: The proof paradigm is essentially the same as in the standard model, except that $\mathcal{B}$ can program the random oracle such that $\mathcal{A}$ cannot tell the difference from a true random function. Thus, $\mathcal{A}$ behaves well and breaks the complex problem. $\mathcal{B}$ obtains both the (input, output)'s of $\mathcal{A}$ and (query, answer)'s to $\mathcal{R}$, and breaks the underlying problem with these extra information.

Random Oracle Model (cont'd)

- After the scheme with ideal random oracle is proven secure. The random oracle is instantiated with a practical primitive like DES or hash functions.
- The "random oracle model" is **ad hoc** and under severe criticisms. The substitution of a random oracle with a practical hash function is the major point to be condemned.

- Without the instantiation part, the security of the random oracle model is already weaker than that of the standard model.
  - In the random oracle model, the reduction would prove that there is no PPT machine with random oracle access can break the target system.
  - In the standard model, the reduction only proves that there is no PPT machine that can break the target system.

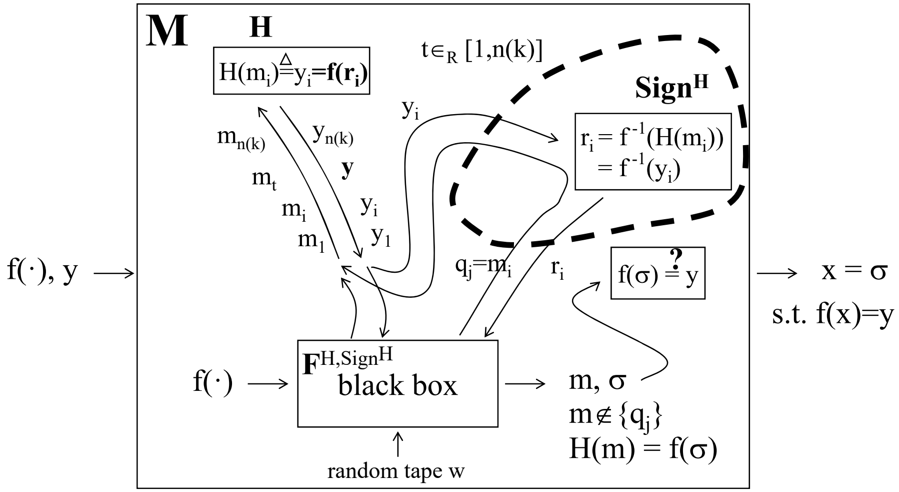
RSA Sig. is EUF-NMA in ROM



Full-domain hash (FDH) with any TDP, e.g. RSA function

Range(H(·)) = $\mathbf{Z_n}^*$ such that $y=f(x) \in$ Range(H(·))

RSA Sig. is EUF-CMA in ROM



Full-domain hash (FDH) with any TDP

$f(r_i)$ or y $f(r_i)$ are always in Range(H(·))

IND-CPA Encryption in RO

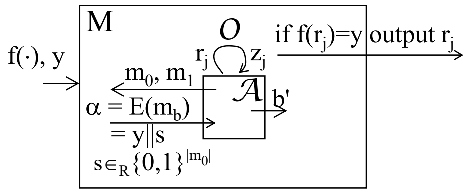
- Encryption: $E(x) = y \| s = f(r) \| (\mathcal{O}(r) \oplus x)$
  Decryption: $x = D(y \| s) = s \oplus \mathcal{O}(f^{-1}(y))$
- This scheme is called Efficient Probabilistic Encryption (EPE) scheme and is semantically secure (polynomially secure or message indistinguishable) if $f(\cdot)$ is a trapdoor 1-1 OWF and $\mathcal{O}(\cdot)$ is a PRNG
- This scheme is not CCA2: given a challenge ciphertext $y\|s$, the adversary can generate a random number s' and ask the decryption oracle $y\|s'$ to get $D(y\|s')=\mathcal{O}(f^{-1}(y))\oplus s'$ and the message is $s\oplus\mathcal{O}(f^{-1}(y))$
- We want to show that this scheme is semantically secure if $f(\cdot)$ is a trapdoor 1-1 OWF and $\mathcal{O}(\cdot)$, a hash function, is a random oracle

pf. ★ Assume that it is not IND, i.e. $\exists$ PPT adversary $\mathcal{A} = (\mathcal{A}_1^O, \mathcal{A}_2^O)$ that defeats the protocol with non-negligible probability

- ★ For an arbitrary $b \in_R \{0,1\}$, $\alpha = E(m_b)$, $\mathcal{A}_1^O(E)$ outputs (m_0, m_1) and $\mathcal{A}_2^O(E, m_0, m_1, \alpha)$ outputs b', s.t. $\Pr\{ b' = b \} \geq 1/2 + 1/p(k)$

IND-CPA Encryption (cont'd)

- We construct an algorithm M(f, y) that inverts f using $\mathcal{A}$
  - Simulate $\mathcal{O}$ oracle by flipping coins
  - Run $\mathcal{A}_1^O(E)$ to get (m_0, m_1). Output **r** if $\mathcal{O}$ is asked an r s.t. $f(r)=y$, and stop
  - Choose $s \in_R \{0,1\}^{|m_0|}$, let $\alpha = y \| s$
  - Run $\mathcal{A}_2^O(E, m_0, m_1, \alpha)$. Output **r** if $\mathcal{O}$ is asked an r s.t. $f(r)=y$, and stop



- $\mathcal{A}$ cannot guess correctly m_b with noticeable probability without asking the oracle $\mathcal{O}$ of r, where $\alpha = y \| (\mathcal{O}(r) \oplus m_b)$ and $y = f(r)$
- Success probability of M(f, y) is non-negligible
  - Define the event A_k: $\mathcal{A}$ asks the query $r = f^{-1}(y)$
  - $\Pr\{\mathcal{A} \text{ succeeds} \mid \neg A_k\} = 1/2 + 1/2^{|m_0|}$
  - $1/2 + 1/p(k) \leq \Pr\{\mathcal{A} \text{ succeeds}\} = \Pr\{\mathcal{A} \text{ succeeds} \mid A_k\} \cdot \Pr\{A_k\} + \Pr\{\mathcal{A} \text{ succeeds} \mid \neg A_k\} \cdot \Pr\{\neg A_k\} \leq \Pr\{A_k\} + 1/2 + 1/2^{|m_0|}$ contradiction ¶

Diffie-Hellman KX to ElGamal

Diffie-Hellman Key Exchange, 1976

Alice
1. choose x
2. g^x
6. **key** $\equiv (g^y)^x$

Bob
3. choose y
4. **key** $\equiv (g^x)^y$
5. g^y

Both x and y are permanent SK.

ElGamal PKE, 1985

Alice ***m***
3. choose k
4. key $\equiv y^k$
5. $u \equiv g^k$
6. $v \equiv y^k \cdot m$

Bob
1. choose x
2. $y \equiv g^x$
7. key $\equiv u^x$
8. $\boldsymbol{m} \equiv v \cdot u^{-x}$

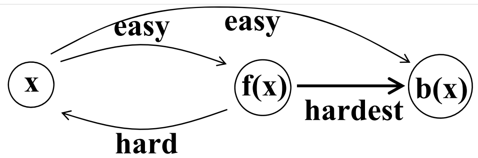
Only x is permanent SK, k is chosen temporarily by the sender.

Another View of ElGamal Design

- Decision Diffie-Hellman problem: prime p, q, q|p-1, order q subgroup $G \subset Z_p$, given g, g^x, g^y, $Z \in_R G$, determine if $Z \equiv g^{xy} \bmod p$
- **In other words, g^{xy} is indistinguishable from a random value Z**
- From the perfect secrecy of one time pad, we know that it is preferable to hide a message with a secret random value (key), e.g.,
  - $k \oplus m$
  - $k + m \bmod p$
  - $k\, m \bmod p$
- How about **g^{xy} m mod p**?
  Let $X = g^x \bmod p$ be the public key, x be the secret key, $g \in G$.
  The ElGamal ciphertext is
  $$c = (g^y, X^y\, m) = (g^y, g^{xy}\, m)$$

Hardcore Predicate

OWF f:



- idea: given f(x), predicting a certain bit (or some derived value, b(x)) of x might be easier than predicting x completely
- predicate: b: $\{0,1\}^* \rightarrow \{0,1\}$ b(x)
- Hardcore: poly-time computable predicate b is hardcore of a function f(·) if for all PPT A', for all p(·), for all sufficiently large n

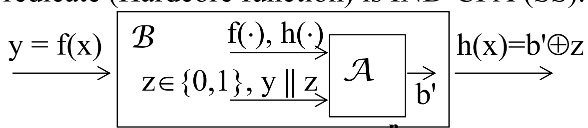
$$\Pr\{A'(f(U_n)) = b(U_n)\} < 1/2 + 1/p(n)$$

- A hardcore predicate b(·) must be unbiased $\Pr\{b(U_n)=1\} = ½$
- Hardcore bit of RSA or Rabin function: LSB(x)

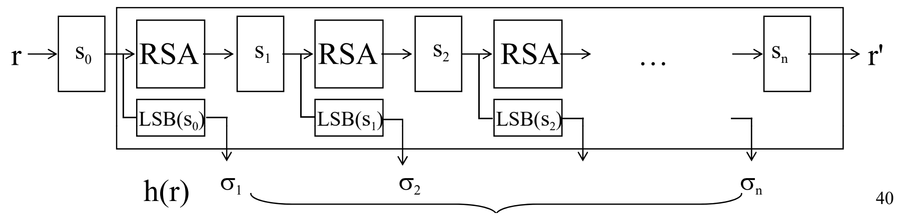
IND-CPA Scheme from TDF

- 1-1 or poly-1 TDF is a necessary ingredient of a PKC
  - Plain RSA encryption has two OWFs, $n = p \cdot q$, $c \equiv_n m^e$, the 2nd is a TDF,
  - ElGamal encryption has only one OWF, $y \equiv_p g^x$, $(g^r, y^r \cdot m)$, but has special commutative property
- TDP itself is not a SS PKE, even a strong OWTP is not.
- 1-1 TDF + Hardcore predicate (Hardcore function) is IND-CPA (SS).

E(b) = (f(r), h(r)⊕b)



- EPE is IND-CPA without random oracles **$E(m) = (f^n(r), h(r) \oplus m)$**



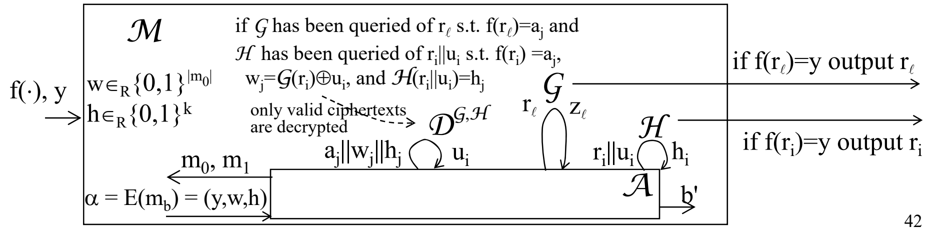
IND-CCA2 Conversion in RO

- Encryption: $E(x) = y \| s = f(r) \| (G(r) \oplus x) \| \mathbf{H(r \| x)}$
  Decryption: $x = D(y \| s) = s \oplus G(f^{-1}(y))$
- We want to show that this scheme is IND-CCA2 **if f(·) is a 1-1 trapdoor OWF**, G(·) and H(·) are instantiated by hash functions, which are assumed random oracles

pf.
- Assume that it is not IND under CCA2, i.e. ∃ PPT adversary $\mathcal{A} = (\mathcal{A}_1^{G, H, D^{G,H}}, \mathcal{A}_2^{G, H, D^{G,H}})$ that can win the game with non-negligible probability, let E = (f, G, H), i.e.
- $\mathcal{A}_1^{G, H, D^{G,H}}(E)$ outputs (m_0, m_1), $b \in_R \{0,1\}$, $\alpha = E(m_b)$, and $\mathcal{A}_2^{G, H, D^{G,H}}(E, m_0, m_1, \alpha)$ outputs b', s.t. $\Pr\{b=b'\} \geq 1/2 + 1/p(k)$
- Now, we are given a blackbox $(\mathcal{A}_1^{G, H, D^{G,H}}, \mathcal{A}_2^{G, H, D^{G,H}})$ and we want to break the fundamental assumption that f is a OWF.

IND-CCA2 Conversion (cont'd)

- $\mathcal{M}$ has control over the inputs to $\mathcal{A}$ and monitors its outputs/queries. If the distributions of all the inputs are the same as in a real attack, $\mathcal{A}$ would win the game with non-negligible advantage.
- $\mathcal{A}$'s inputs:
  - Inputs to $\mathcal{A}_1$: E, responses of $\mathcal{G}$, $\mathcal{H}$, and $\mathcal{D}^{\mathcal{G},\mathcal{H}}$
  - Inputs to $\mathcal{A}_2$: E, m_0, m_1, α, responses of $\mathcal{G}$, $\mathcal{H}$, and $\mathcal{D}^{\mathcal{G},\mathcal{H}}$
- Distributions of the inputs in a real attack:
  - $\mathcal{G}$, $\mathcal{H}$: must be uniformly random, must be a consistent function
  - $\mathcal{D}^{\mathcal{G},\mathcal{H}}$: must be able to decrypt a valid ciphertext
  - α: must be a valid ciphertext of either m_0 or m_1

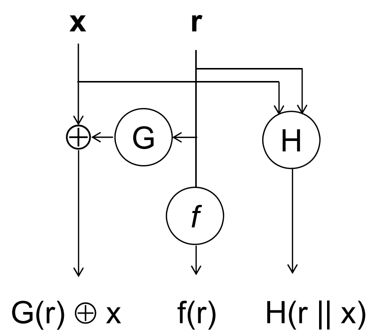


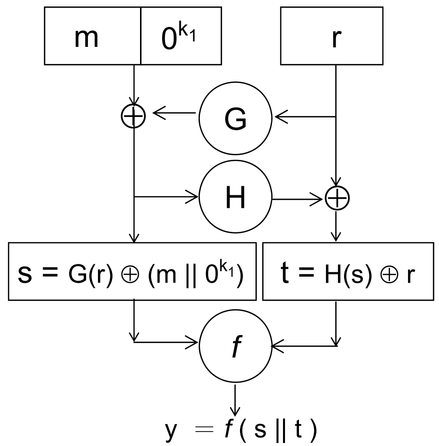
IND-CCA2 Conversion (cont'd)

- We construct an algorithm $\mathcal{M}$(f, y) that inverts f using $\mathcal{A}$
  - Simulate $\mathcal{G}$, $\mathcal{H}$, and $\mathcal{D}^{\mathcal{G},\mathcal{H}}$ by flipping coins and the following
    - If $\mathcal{G}$ is queried of **r** s.t. f(r)=y, returns r and stop, else returns $\mathbf{z} \in_R \{0,1\}^{|m_0|}$
    - If $\mathcal{H}$ is queried of **r** || x s.t. f(r)=y, returns r and stop, else returns $\mathbf{z} \in_R \{0,1\}^k$
    - If $\mathcal{D}^{\mathcal{G},\mathcal{H}}$ is queried of **a || w || h**, $\mathcal{G}$ is queried of **r**, and $\mathcal{H}$ is queried of **r || u** s.t. f(r)=a, $w = \mathcal{G}(r) \oplus u$, and $\mathcal{H}(r \| u) = \mathbf{h}$, returns **u**, otherwise return invalid
  - Run $\mathcal{A}_1^{\mathcal{G}, \mathcal{H}, \mathcal{D}^{\mathcal{G},\mathcal{H}}}(f)$ to get (state, m_0, m_1)
  - Choose $w \in_R \{0,1\}^{|m_0|}$ and $b \in_R \{0,1\}^k$, let $\alpha = y \| w \| h$
  - Run $\mathcal{A}_2^{\mathcal{G}, \mathcal{H}, \mathcal{D}^{\mathcal{G},\mathcal{H}}}(f, \text{state}, m_0, m_1, \alpha)$
- Why does this work?
  We believe that $\mathcal{A}$ cannot guess correctly with noticeable probability about b without querying the oracle $\mathcal{G}$ of r and $\mathcal{H}$ of r || m_b. If $\mathcal{A}$ does not query $\mathcal{H}$ of r || m_b, the decryption oracle is useless. If $\mathcal{A}$ does not query $\mathcal{G}$ of r, the message m_b is hidden perfectly. The challenge ciphertext satisfies y = f(r), $w = \mathcal{G}(r) \oplus m_b$, $h = \mathcal{H}(r \| m_b)$, $\alpha = y \| w \| h$

Comparison with OAEP

$E(x) = f(r) \| (G(r) \oplus x) \| \mathbf{H(r \| x)}$





FO99 Hybrid Encryption

- E. Fujisaki and T. Okamoto, "Secure Integration of Asymmetric and Symmetric Encryption Schemes," Crypto'99
- Enc: $\mathcal{E}^{hy}(PK, m) = <\mathcal{E}^{asym}_{PK}(\sigma; H(\sigma, m)),\ \mathcal{E}^{sym}_{G(\sigma)}(m)>$
  Dec: $\sigma' = \mathcal{D}^{asym}_{SK}(C_1)$, $m' = \mathcal{D}^{sym}_{G(\sigma)}(C_2)$, $h' = H(\sigma', m')$,
  check $C_1 \stackrel{?}{=} \mathcal{E}^{asym}(\sigma'; h')$
- If $\mathcal{E}^{asym}_{PK}(\cdot)$ is a OWE and $\mathcal{E}^{sym}_{G(\cdot)}(m)$ is SS,
  $\mathcal{E}^{hy}(\cdot)$ is IND-CCA2 in the random oracle model
- e.g. $\mathcal{E}^{asym}(\cdot)$ is ElGamal, $\mathcal{E}^{sym}(\cdot)$ is one-time pad

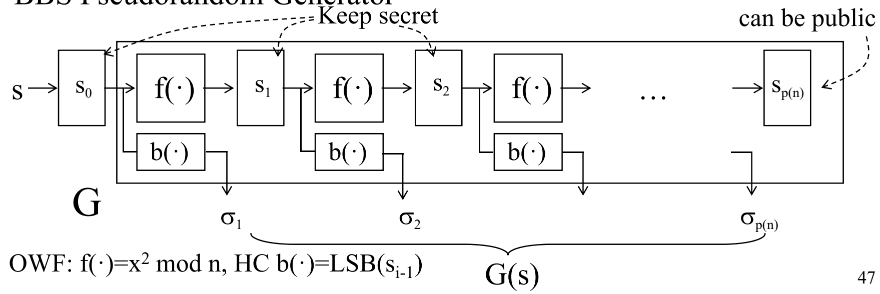
This is the second method to transform a weakly secure PKE (OWE) to an IND-CCA secure PKE

ElGamal or RSA?

- Efficiency
  - Computation time
  - Length of ciphertext / signature
- Security
  - A stricter security notion defines more secure scheme.
  - A weaker assumption is less prone to be invalid.
  - Standard (plain) model is far better than random oracle model.
  - RSA encryption is OWE itself; use $f(r) \,||\, (\mathcal{O}(r) \oplus x)$ to get an IND-CPA scheme in the RO model; use $f(r) \,||\, (G(r) \oplus x) \,||$ **$H(r \,||\, x)$** to get an IND-CCA scheme in the RO model
  - RSA signature is EUF-CMA in the RO model
  - ElGamal is IND-CPA in standard model; use FO99 transform to get an IND-CCA scheme in the RO model; Cramer-Shoup designed an IND-CCA secure scheme in the standard model based on a modified ElGamal scheme
  - ElGamal signature is ??-secure

Pseudorandomness and PRNG

- $\forall$ PPT algorithm D, every positive $p(\cdot)$, all sufficiently large n
  $$|\Pr\{D(X_n, 1^n)=1\} - \Pr\{D(U_n, 1^n)=1\}| < 1/p(n)$$
  n-bit uniform distribution
- f: $\{0,1\}^n \to \{0,1\}^{\ell(n)}$
  $f(\cdot)$ is a pseudorandom generator if $f(U_n) \approx^C U_{\ell(n)}$
  n-bit random seed
  computationally indistinguishable
  ℓ-bit random sequence
- BBS Pseudorandom Generator



OWF: $f(\cdot) = x^2 \bmod n$, HC $b(\cdot) = LSB(s_{i-1})$

BBS PRNG

- LSB(x) (even log n bits) is a hardcore predicate of $f(x) = x^2 \bmod n$
  - W. Alexi, B. Chor, O. Goldreich, and C. P. Schnorr, "RSA and Rabin functions: Certain parts are as hard as the whole," SIAM JC88
- Thus, the assumption underlying the pseudo-randomness of BBS is the one-wayness of Rabin function, which is equivalent to factoring.
- Original BBS paper
  - Lenore Blum, Manuel Blum, and Michael Shub, "Comparison of Two Pseudo-random number generator," Crypto'82

only proves that QRA implies the pseudo randomness of BBS

$$QR(n) \le_T LSB(n) \le_T Fact(n)$$

pf. If you have an adversary $\mathcal{A}$ that given $x^2 \bmod n$ for $x \in QR_n$ as input, can determine LSB(x). Construct an algorithm $\mathcal{B}$, given $y \in Z_n$, determine if $y \in QR_n$. ❶ calculate $z = y^2 \bmod n$, ❷ output $y \in QR_n$ if $\mathcal{A}(z) = LSB(y)$; otherwise output $y \notin QR_n$

Secure Applications from RF/PRF

- A general methodology for designing applications that share PRF
  - ① design your scheme (assuming all parties legitimate) sharing a random function $f:\{0,1\}^n \rightarrow \{0,1\}^n$ (the adversary can obtain, from legitimate users, the values of $f(\cdot)$ on arguments of their choices, but does not have direct access to $f(\cdot)$ itself)
  - ② prove the security of your system, assuming $f(\cdot)$ is a true random function
  - ③ replace the random function in your scheme with a pseudo random function
  - ④ if your new scheme become insecure (i.e. has different behavior from the true random scheme) then this system can be used to distinguish the pseudo random function from $f(\cdot)$

Secure PKE from PRF

Given a PRF $f_k(\cdot)$: $\{0,1\}^{\ell} \rightarrow \{0,1\}^{\ell}$, the cryptosystem is as follows:

❶ key generation: $k \in_R \{0,1\}^n$

❷ encryption: $m \in \{0,1\}^{\ell}$, $r \in_R \{0,1\}^{\ell}$, $c = E_k(m) = (r, f_k(r) \oplus m)$

❸ decryption: $m = D_k(r, s) = f_k(r) \oplus s$

Note: this is a symmetric block encryption scheme

$f_k^{-1}(\cdot)$ might not exist, might not be computable

Is the above scheme secure? (in what sense?)

❶ if a true random function is used in the above scheme, each block of message has perfect secrecy in which given a ciphertext c, the probability of correctly recovering m is only $2^{-\ell}$. The probability to correctly recover each bit is only 1/2 and is independent for each bit. In this sense, it does not matter how you choose ℓ in the above scheme.

❷ when a PRF f_k is used in place of the true random function, if there exists an adversarial algorithm which can guess the correct m given

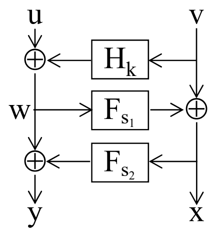
Secure $\mathcal{SE}$ from RP/PRP

- Invertible pseudo random permutation is a secure block encryption: c=f(m)
- Invertible PRP from PRF: Luby-Rackoff

  $H_k()$ is almost XOR universal

  $w = u \oplus H_k(v)$, $x = v \oplus F_{s_1}(w)$, $y = w \oplus F_{s_2}(x)$

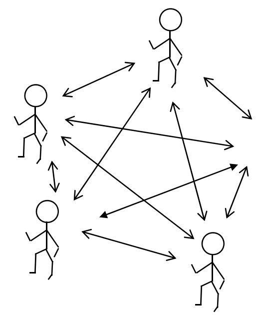
  $w' = y \oplus F_{s_2}(x)$, $v' = x \oplus F_{s_1}(w')$, $u' = w' \oplus H_k(v')$



- PRF in counter mode is a secure stream cipher
- DES is simulating a invertible random permutation
- Verifiable trapdoor pseudo-random permutation is a secure unique signature scheme

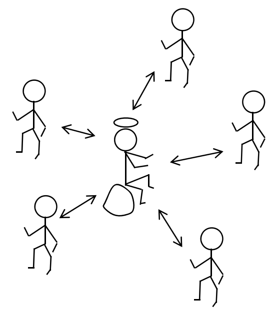
Multi-Party Computation

Real model



mutually distrustful parties

Ideal model



mutually trusted parties and a trusted party

- To what extent the trusted third party in the ideal model can be emulated by the mutually distrustful parties in the real model?
- To what extent the protocol in the real model can be simulated in the ideal model with the help of a trusted oracle?

Secure MPC

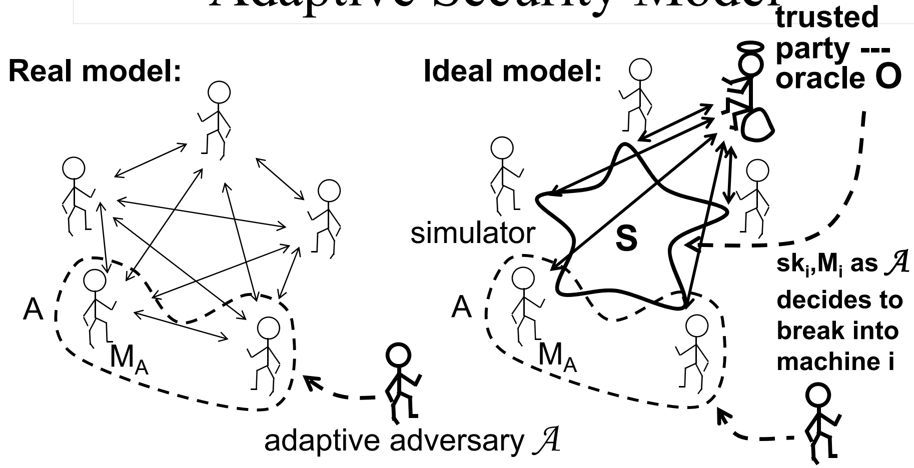
the Simulation Paradigm

- Used also in the definition of zero-knowledge and semantic security

--- A scheme is ***secure*** if whatever a *feasible* adversary can obtain after attacking it is also *feasibly* attainable in an "ideal setting" ---

- In this way, the protocol emulated the ideal setting – computation with the help of a trusted party – and achieves all the desired properties
  - Preservation of the ***privacy*** of each player's local inputs beyond what is revealed by the local outputs
  - ***Correctness*** of honest parties' local outputs???

Adaptive Security Model



- **Correctness: whatever can be obtained in the ideal model, can also be obtained in the real model.**
- **Privacy: whatever can be observed by the adversary in the real model, can also be observed by the adversary in the ideal model.**

Static vs. Adaptive Adversary

- **Static adversary**:

  Assume that $\mathcal{A}$ controls 2 out of n machines from the start

  Goal: $S(sk_1, sk_2, M_1, M_2) \approx^C$
  $View_{\Pi, \mathcal{A}}(sk_1, sk_2, \ldots, sk_n, M_1, M_2, \ldots, M_n)$

  i.e. simulator S in the ideal model must produce the view indistinguishable from that of an adversary in the real model

- **Adaptive adversary** (Mobile adversary in proactive model):

  As $\mathcal{A}$ decides to break into a machine, $\mathcal{A}$ obtains its secret key at that moment.

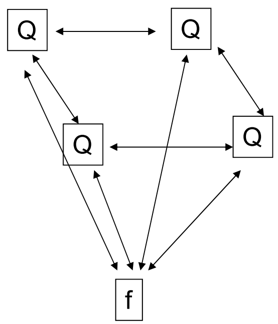
  Goal: $S^{O}() \approx^C View_{\Pi, \mathcal{A}}(sk_1, sk_2, \ldots, sk_n, M_1, M_2, \ldots, M_n)$

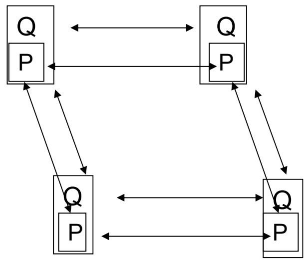
  i.e. simulator S in the ideal model, can ask an oracle **O** about the secret sk_i and the output M_i of the i-th machine during the simulation when the adversary chooses machines to attack and must produce the view indistinguishable from that of an adversary in the real model

Universal Composability

- What about security "in conjunction with other protocol executions"?
  - Other executions of the same protocol?
  - Other executions of arbitrary other protocols?
  - "Intended" (coordinated) executions?
  - "unintended" (uncoordinated) executions?
- Composition of instances of the same protocol:
  - With same inputs/different inputs
  - Same parties/different parties/different roles
  - Sequential, parallel, concurrent (either coordinated or uncoordinated).
- "Subroutine composition" (modular composition):
  - protocol Q calls protocol P as subroutine.
  - Non-concurrent, Concurrent
- General composition: Running in the same system with arbitrary other protocols (arbitrary network activity), without coordination.
- Is security maintained under these operations?

Modular Composition





Towards the composition theorem

The hybrid model with ideal access to func. f (the f-hybrid model):

- Start with the real-life model of protocol execution.
- In addition, the parties have access to a trusted party F for f:
  - At pre-defined rounds, the protocol instructs all parties to sends values to F.
  - F evaluates f on the given inputs and hands outputs to parties
  - Once the outputs are obtained the parties proceed as usual.
- Notation: $\mathrm{EXEC}^{f}_{P,H,Z}$ is the ensemble describing the output of Z after interacting with protocol P and adversary H in the f-hybrid model.

Note:

- During the “ideal call rounds” no other computation takes place.
- Can generalize to a model where in each “ideal call round” a different function is being evaluated. But doesn’t really add power (can use a single universal functionality).

Modular composition

(Originates with [Micali-Rogaway91])

Start with:

- Protocol Q in the f-hybrid model
- Protocol P that securely realizes f

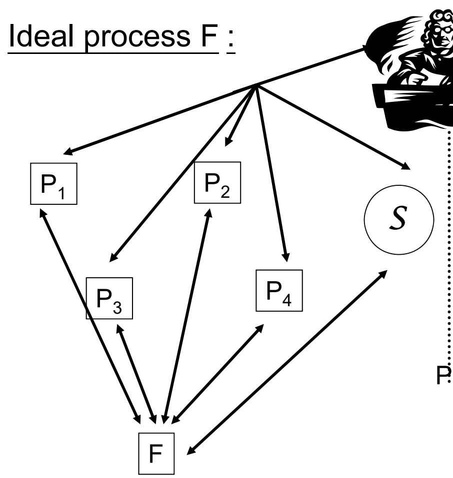
Construct the composed protocol Q^P:

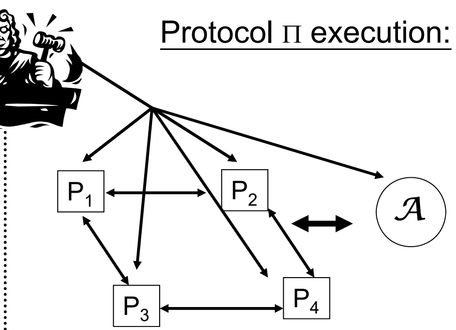
- Each call to f is replaced with an invocation of P.
- The output of P is treated as the value of f.

Notes:

- In Q^P, there is at most one protocol active (ie, sending messages) at any point in time: When P is running, Q is suspended.
- It is important that in P all parties terminate the protocol at the same round. Otherwise the composition theorem does not work…
- If P is a protocol in the real-life model then so is Q^P. If P is a protocol in the f'-hybrid model for some function f', then so is Q^P.

Universal Composability





Protocol Π securely realizes F if:

For any adversary $\mathcal{A}$

There exists an adversary $\mathcal{S}$

Such that no environment Z can tell whether it interacts with:

- A run of Π with $\mathcal{A}$
- An ideal run with F and $\mathcal{S}$