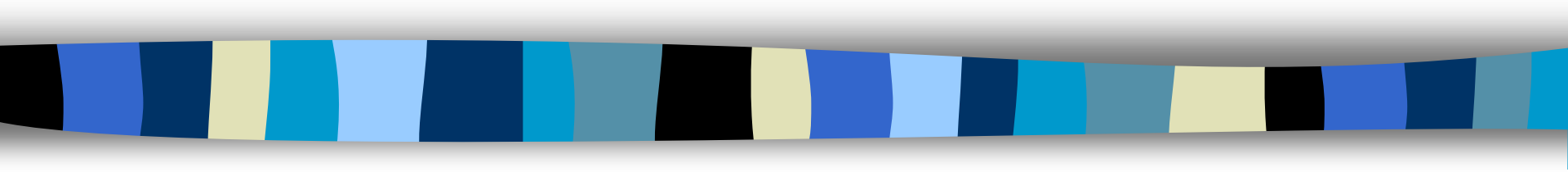
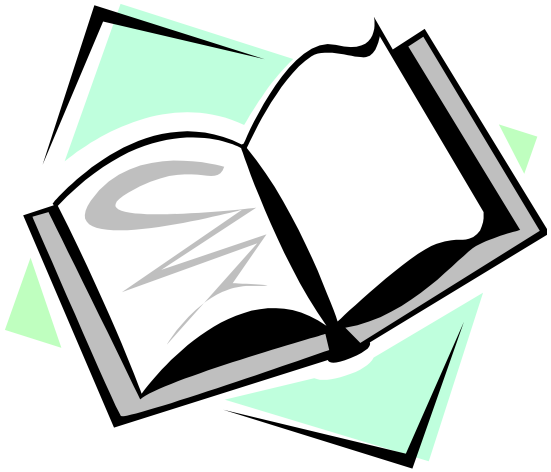
Cryptography
CS 555



Topic 4: Computational Approach to Cryptography

Outline and Readings

- Outline
  - Principles of Modern Cryptography
  - Computational Security
  - Ciphertext indistinguishability security



- Readings:
  - Katz and Lindell: 1.4, 3.1, 3.2

Kerckhoffs's Principle

- **Kerckhoffs's Principle:**
  - The cipher method must not be required to be secret, and it must be able to fall into the hands of the enemy without inconvenience
- **Shannon's maxim**: "The enemy knows the system."
- Open design; Security by obscurity doesn't work
- Should assume that the adversary knows the algorithm; the only secret the adversary is assumed to not know is the key
  - Reverse engineering, careful review of algorithm, etc.
- What is the difference between the algorithm and the key?

Formulation of Exact Definitions

- Formal definitions of security are essential prerequisites of cryptography
  - Design: without a definition, doesn't know whether a design achieves it
  - Usage: without a definition, doesn't know whether using a crypto primitive in a setting is suitable
  - Study: when comparing different schemes, need to know what kinds of security they provide

What Does A Security Definition Looks Like?

- Define what is insecurity (i.e., what is considered to be a break)
- Define what is the power of the adversary

- A cryptographic scheme for a given task is secure if no adversary of a specific power can achieve a specified break.

Defining Secure Encryption

- Adversary should not be able to
  1. Recover the key
  2. Find the plaintext corresponding to a ciphertext
  3. Cannot determine any character of the plaintext
  4. Can derive any meaningful information about the plaintext
  5. Can compute any function of the plaintext

Adversarial Models for Ciphers

- The language of the plaintext and the nature of the cipher are assumed to be known to the adversary.
- **Ciphertext-only attack**: The adversary knows only a number of ciphertexts.
- **Known-plaintext attack**: The adversary knows some pairs of ciphertext and corresponding plaintext.
- **Chosen-plaintext attack:** The adversary can choose a number of messages and obtain the ciphertexts
- **Chosen-ciphertext attack:** The adversary can choose a number of ciphertexts and obtain the plaintexts.

What kinds of attacks have we considered so far?

When would these attacks be relevant in wireless communications?

Reliance on Precise Assumptions

- Assumptions (under which a scheme is secure) must be precisely stated
  - To validate the assumption
  - To compare different schemes; it is desirable to rely on weaker assumptions
  - To facilitate formal security proofs

How to Tell Whether a Definition is Good

- Needs to tell whether the mathematical formulation matches the real world situation
  - Whether in real world the adversary have more power.
    - E.g., power analysis attacks, side channel attacks
  - Whether the adversary is able to achieve a different goal, which should be considered to be a break
    - E.g., data privacy: k-anonymity,
- Use the following tools
  - Appeal to intuition
  - Prove equivalence
  - Use examples

Rigorous Proofs of Security

- Intuitions can often be wrong when considering security/cryptography
  - Bugs/errors can be very subtle
- The reductionist approach
- A Theorem looks like: Assume that X is true (e.g., certain problem is hard), Construction Y is secure according to the given definition,
- Proof looks like: Given an adversary A that breaks Y according to the definition, using A we can construct something that falsifies X

Towards Computational Security

- Perfect secrecy is too difficult to achieve.
- The computational approach uses two relaxations:
  - Security is only preserved against **efficient** (computationally bounded) adversaries
    - Adversary can only run in feasible amount of time
  - Adversaries can potentially succeed with some **very small probability** (that we can ignore the case it actually happens)

The Concrete Approach

- Quantifies the security by explicitly bounding the maximum success probability of adversary running with certain time:
  - "A scheme is (t,ε)-secure if **every** adversary running for time at most t succeeds in breaking the scheme with probability at most ε"
    - One may also bound t number of computations, CPU cycles, etc.
  - Example: a strong encryption scheme with n-bit keys may be expected to be $(t, t/2^n)$-secure.
    - N=128, $t=2^{60}$, then $\varepsilon= 2^{-68}$. (# of seconds since big bang is 2^{58})
- Makes more sense with symmetric encryption schemes.

The Asymptotic Approach

- A cryptosystem has a security parameter
  - E.g., number of bits in the RSA algorithm (1024,2048,…)
  - Typically, the key depends on the security parameter
  - The bigger the security parameter, the longer the key, the more time it takes to use the cryptosystem, and the more difficult it is to break the scheme
  - The crypto system runs in time polynomial in the security parameter
    - Security parameter is often written as an input 1^n
  - "A scheme is secure if every PPT adversary succeeds in breaking the scheme with only negligible probability"

Efficient Computation

- Efficient computation is equated with Probabilistic Polynomial Time (PPT)
  - The algorithm has access to sequence of unbiased coins
  - Often times, the time is polynomial in the security parameter
- Both the crypto scheme and the adversary are assumed to be PPT

Negligible Probability

- Want the adversary's success probability to be small, but the probability is a function of the security parameter n
- Wants to say that a function f(n) is small when n grows.
  - What functions is very small when n grows?
  - 1/f(n) should be a function that increases fast with n
- A function f is negligible if for every polynomial p(.) there exists an N such that for all integers n>N, it holds that f(n)<1/p(n)

Examples of Negligible Functions

- Examples:
  - 2^{-n}; $2^{-\sqrt{n}}$; $n^{-\log n}$
- Given two negligible functions f and g
  - The function f+g is negligible
  - The function p(n) f(n) is negligible for any polynomial p(n)
- Given a negligible function f, one can choose a security parameter n that is not too large to make f(n) so small that it can be safely ignored

Symmetric-key Encryption

- A symmetric-key encryption scheme is comprised of three algorithms
  - **Gen** Input: security parameter 1^n
    - $k \leftarrow$ **Gen**(1^n) Assume, wlog, that |k| > n
  - **Enc** Input: key k, plaintext m
    - c $\leftarrow$ **Enc**$_k(m)$
  - **Dec** Input: key k, ciphertext c
    - m := **Dec**$_k(m)$

Requirement: $\forall k \, \forall m \;\; [\, \mathbf{Dec}_k(\mathbf{Enc}_k(m)) = \mathrm{m} \,]$

  - If for k output by **Gen**(1n), **Enc** is defined only for messages of length ℓ(n), this is called a fixed-length encrpytion scheme

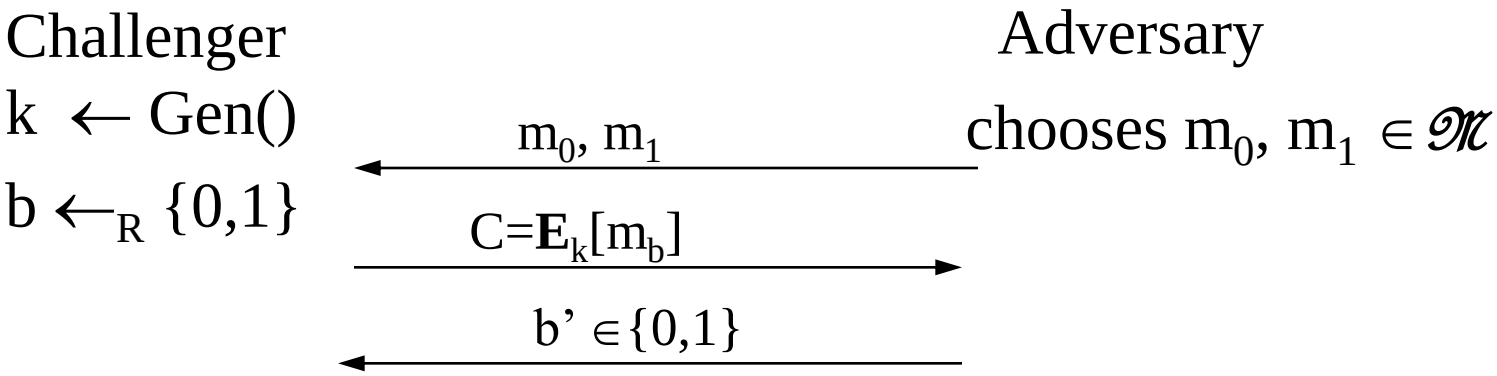
Defining Security

- Desire “semantic security”, i.e., having access to the ciphertext does not help adversary to compute any function of the plaintext.
  - Difficult to use

- Equivalent notion: Adversary cannot distinguish between the ciphertexts of two plaintexts

Recall: Perfect Secrecy via Adversarial Indistinguishability

- Define an experiment called **PrivK^{eav}**:
  - Involving an Adversary and a Challenger
  - Instantiated with an Adv algorithm A, and an encryption scheme Π = (Gen, Enc, Dec)



PrivK^{eav} = 1 if b=b', and PrivK^{eav} = 0 if b ≠ b'
For every adversary, PrivK^{eav} = 1 holds with prob 1/2

Towards IND Security

- Modify the formulation of perfect secrecy using **PrivK^{eav}** in the following ways
  - Adversaries run in polynomial time
  - Adversaries might determine which message is encrypted with probability negligibly better than ½
  - Require two messages m_0 and m_1 to be the same length
    - Most encryption schemes do not hide length of messages

IND Security

- An encryption scheme Π = (Gen, Enc, Dec) has indistinguishable encryptions in the presence of an eavesdropper if for all PPT adversary A, there exists a negligible function negl such that
  - $\Pr[\mathbf{PrivK}^{\mathbf{eav}}_{\mathbf{A},\Pi}=1] \leq \frac{1}{2} + \text{negl}(n)$

- Equivalently, any adversary would behave the same way whether it sees the encryption of m_0 or m_1
  - $|\Pr[\text{output}(\mathbf{PrivK}^{\mathbf{eav}}_{\mathbf{A},\Pi}(n,0)) = 1] - \Pr[\text{output}(\mathbf{PrivK}^{\mathbf{eav}}_{\mathbf{A},\Pi}(n,1)) = 1]| \leq \text{negl}(n)$

Coming Attractions …

- Pseudorandomness
- Pseudo Random Number Generator
- Stream Ciphers

- Reading: Katz & Lindell: 3.3 and 3.4