Lecture 2

JANUARY 12 → JANUARY 24

Instructor: Fang Song

Disclaimer. Draft note. No guarantee on completeness nor soundness. Read with caution, and shoot me an email at fsong@pdx.edu for corrections/comments (they are always welcome!)
Logistics.

- D2L: forward d2l email to your email account. Solutions will also be posted on d2l.
- DRC: "Would you like to make money for taking notes? If you are willing to share your notes, you will be paid up to $25 per credit hour to scan and upload your notes to DRC Online. If you're interested go to the DRC homepage: pdx.edu/drc, select DRC Online, and follow the link to sign up as a note taker. You can find answers to most questions by selecting "Note Takers" in the top menu. Thanks."
- Background survey: Glad to see many of you actually showed interest, not just "to get the credit". Level of confidence: math proofs and probability have the lowest overall scores. **Suggested readings**: Resource page – useful links: tips on math proofs; Probability: MIT course, draft book[1], Part IV Chapters 17 - 20; CLRS[2] Appendix Part C in particular.
- Clarification: reading preferable before class but not required. Quizzes are based on materials in lectures already given.

Last time. Overview, math background, perfect secrecy.
Today. Perfect secrecy: equivalence, example (one-time pad), limits. Computational secrecy.

Note: "Random" in this course refers to some general *random* experiment/phenomenon, and we will explicitly say "uniformly random" to mean a uniform distribution.

Recall an private-key encryption scheme $\Pi = (G, E, D)$ with $E : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$ & $D : \mathcal{K} \times \mathcal{C} \to \mathcal{M}$. G is a *randomized* key generating algorithm. E, D can be either deterministic or randomized. Let K, M, C be random variables taking values in $\mathcal{K}, \mathcal{M}, \mathcal{C}$ respectively. K is distributed according to the key-gen algorithm G, and M represents *a priori* the distribution of the messages being sent. C is deduced by $E_K(M)$, denoting the resulting ciphertext from random key and message. We assume that K and M are independent, and the distribution of M is known to the attacker.

1 Perfect secrecy

1.1 Equivalent Definitions

Definition 1 (KL-Def.2.3)**.** Π is perfectly secret if for every

$$\Pr(M = m | C = c) = \Pr(M = m). \tag{1.1}$$

[1] https://courses.csail.mit.edu/6.042/spring16/mcs.pdf
[2] https://mitpress.mit.edu/books/introduction-algorithms

Definition 2. Π is perfectly secret if for every $m, m' \in \mathcal{M}$ and every $c \in \mathcal{C}$

$$\Pr(E_K(m) = c) = \Pr(E_K(m') = c)\,. \tag{1.2}$$

Lemma 3. *Definition 1 is* equivalent *to Definition 2 .*

Proof. We need to show **both** directions: Def. 1 ⇒ Def. 2 (KL-EX.2.4) and Def. 2 ⇒ Def. 1 (KL book).

- (Def. 2 ⇒ Def. 1) Fix an arbitrary $c \in \mathcal{C}$. Observe that for every $m' \in \mathcal{M}$

  $$\Pr[C = c | M = m'] \stackrel{\text{def}}{=} \Pr[E_K(M) = c | M = m'] = \Pr[E_K(m') = c]\,.$$

  But Def. 2 says that $\Pr[E_K(m') = c] = \Pr[E_K(m) = c]$ for every $m, m' \in \mathcal{M}$. This means that for every m', $\Pr[C = c | M = m']$ is a constant and denote it δ_c. For any $m \in \mathcal{M}$ and $c \in \mathcal{C}$ (assuming $\Pr[C = c] \neq 0$), we have

  $$\begin{aligned}
  \Pr[M = m | C = c] &= \frac{\Pr[C = c | M = m] \cdot \Pr[M = m]}{\Pr[C = c]} \\
  &= \frac{\Pr[C = c | M = m] \cdot \Pr[M = m]}{\sum_{m' \in \mathcal{M}} \Pr[C = c | M = m'] \cdot \Pr[M = m']} \\
  &= \frac{\delta_c}{\sum_{m' \in \mathcal{M}} \delta_c \cdot \Pr[M = m]} \\
  &= \Pr[M = m]
  \end{aligned}$$

- (Def. 1 ⇒ Def. 2) For any $c \in \mathcal{C}$ and $m \in \mathcal{M}$, by Baye's theorem we have

  $$\Pr[C = c | M = m] = \Pr[M = m | C = c] \cdot \Pr[C = c] / \Pr[M = m]\,.$$

  But we know by hypothesis that $\Pr[M = m | C = c] = \Pr[M = m]$, and this implies that $\Pr[E_K(m)] := \Pr[C = c | M = m] = \Pr[C = c] =: \delta_c$ holds for any m and c. Therefore for any $m, m' \in \mathcal{M}$ and $c \in \mathcal{C}$, $\Pr[E_K(m) = c] = \delta_c = \Pr[E_K(m') = c]$.

□

Perfect indistiguishability. Def. 2 can be read as the distributions (over $\mathcal{C}$) resulting from encrypting one message m and encrypting another message m' are *identical*. From the attackers point of view, these two distributions are *indistinguishable*. We can reformulate this by a *experiment* or *game* between an attacker/adversary $\mathcal{A}$ and the so-called *challenger* CH. This gives another equivalent definition of PS. This will also serve as the template for many of our definitions in the future.

Definition 4. $\Pi = (G, E, D)$ is perfectly indistinguishable if for every attacker $\mathcal{A}$, it holds that

$$\Pr[\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi}] = \frac{1}{2}. \tag{1.3}$$

FS NOTE: Draw a game diagram

1. Adversary $\mathscr{A}$ outputs a pair of messages $m_0, m_1 \in \mathscr{M}$.
2. CH generates a key $k \leftarrow G$, and a uniform $b \leftarrow \{0,1\}$. Compute *challenge ciphertext* $c \leftarrow E_k(m_b)$ and give to $\mathscr{A}$.
3. $\mathscr{A}$ outputs a bit b' as the guess of b.
4. Define the output of the experiment to be 1 if $b' = b$, and 0 otherwise. Write $\mathsf{PrivK}^{\mathsf{eav}}_{\mathscr{A},\Pi} = 1$ if the experiment output is 1, in which case we call $\mathscr{A}$ succeeds.

Figure 1: Adversarial indistinguishability experiment $\mathsf{PrivK}^{\mathsf{eav}}_{\mathscr{A},\Pi}$

Lemma 5 (KL-Lemma 2.6)**.** *Perfect secrecy is equivalent to perfect indistinguishability.*

Proof in HW.

Remarks: why care about so many equivalent definitions?. Different characterizations give a more comprehensive understanding. In different situations, some are easier to work with. Don't constrain yourself to one!

2 One-time Pad

We have a nice definition. But can we achieve it? Answer is yes. One-time-pad (OTP) by *Vernam*, patented in 1917. However, the formal analysis, i.e. definition and proof of security, had to wait for about 25 years till Claude Shannon's seminal work. Shannon was a mathematician, electrical engineering and cryptographer. He was the founding person of information theory, and he was also one of the early researcher in artificial intelligence. We are studying his main contribution in cryptography.

KL Construction 2.8. Fix integer $\ell > 0$. $\mathscr{K} = \mathscr{M} = \mathscr{C} = \{0,1\}^\ell$.

- G: sample a uniformly random key $k \leftarrow \mathscr{K}$.
- E: on input $m \in \mathscr{M}$ and $k \in \mathscr{K}$, output $c := m \oplus k$.
- D: on input $c \in \mathscr{C}$ and $k \in \mathscr{K}$, output $m := c \oplus k$.

Figure 2: Construction of one-time-pad

Fact: $\oplus$ is XOR bit-wise. OTP is correct, simple and actually got used quite a lot by national intelligence angencies in mid-20th.

Theorem 6 (KL Theorem 2.9)**.** *The one-time-pad encryption scheme is perfectly secret.*

Idea. Ciphertext is a uniformly random string regardless of the plaintext.

Proof. Use Def. 2. For any $m, m' \in \mathcal{M}$ and $c \in \mathcal{C}$.

$$\Pr[E_K(m) = c] = \Pr_{k \leftarrow \mathcal{K}}[k \oplus m = c] = 1/2^{\ell}.$$

Similarly $\Pr[E_K(m) = c] = \Pr_{k \leftarrow \mathcal{K}}[k \oplus m' = c] = 1/2^{\ell}$. Hence $\Pr[E_K(m) = c] = \Pr[E_K(m') = c]$ for arbitrary m, m' and c. This concludes the proof. □

Crucial observations on one-time-pad.

- Key has equal length as a message. $|k| = |m|$.
- "One-time" means "one-time", seriously! Consider encrypting m and m' under the same key k, we get $c = m \oplus k$ and $c' = m' \oplus k$. Having c and c', one can compute $c \oplus c' = m \oplus m'$. This can reveal a lot of information and potentially both m and m'. [3]

We will see that these are not just shortcomings of a particular encryption scheme, OTP. But actually both are inherent limitations of *perfect secrecy*.

2.1 Limitations of perfect secrecy

Limitation 1: Perfect secrecy cannot avoid long keys.

Theorem 7 (KL Theorem 2.10)**.** *Suppose* $\Pi = (G, E, D)$ *is perfectly secret, then* $|\mathcal{K}| \geq |\mathcal{M}|$.

Proof. Proof by Contradiction. Suppose, Π is perfectly secret but $\mathcal{K} = \{0,1\}^{\ell}$ and $\mathcal{M} = \{0,1\}^{n}$ with $\ell < n$. Consider a uniform distribution over the $\mathcal{M}$, i.e. $\Pr[M = m] = 1/2^n$ for all $m \in \mathcal{M}$. Perfect secrecy then says $\Pr[M = m|C = c] = \Pr[M = m] = 1/2^n$. However, consider an attacker that guesses a key at random, then it will be correct with prob. $1/2^{\ell}$, in which case s/he can recover m from c with certainty. Therefore $\Pr[M = m|C = c] \geq \Pr[\text{guess correct key}] = 1/2^{\ell} > 1/2^n$. □

Attacking the system with prob. $1/2^{\ell}$ may not seem too bad, but this violation can be boosted. HW bonus problem has a strengthen of the proof here.

Limitation 2: "one-time" only. It is inherent too, but more general (applies to computational secrecy too). It has to do with the threat/attack model, i.e., secrecy against eavesdropping is not strong enough to ensure securely encrypting *multiple* messages under the same key.

[3] The most embarrassing example of reuse the same one-time-pad key would be the *Venona Project* (https://en.wikipedia.org/wiki/Venona_project). See a visual demonstration http://crypto.stackexchange.com/questions/59/taking-advantage-of-one-time-pad-key-reuse.

3 Computational secrecy

Resolving limitation 1. To get away with *long* keys, have to appeal to relaxations in perfect secrecy.

Recall the perfect indist. formulation: distinguishing prob. must be exactly 1/2 (i.e. attacker always fails completely) against *any* attacker (i.e. can be computationally unbounded) .

- consider "efficient" attackers only
- accept "small" break of scheme

Both are necessary. Proof of Theorem 7 gives an efficient attack. Achieving small break against unbounded attackers need long key too. KL-Ex. 2.11,2.12.

How to interpret the constraints, two general approaches, **concrete** and **asymptotic**.
Concrete approach.

Template for **concrete** security A scheme is (t,ε)-*secure* if for every adversary running for time at most t succeeds in breaking the scheme with probability at most ε.

Example 8. No adversary running at most 2^{80} CPU cycles can break the scheme with prob. better that 2^{-60}. Modern private-key encryption often assumes **optimal** security: when the key has n bits (i.e. key space size 2^n), any adversary running for time t can break the scheme with prob. at most $ct/2^n$ for some constant c.

Important in practice, but difficult to provide and often misleading: what type of computing power, what algorithm was implemented?.
Asymptotic approach. WE WILL TAKE THIS! Security parameter n – think of it as the key length, and everything is viewed as a function of n rather than concrete numbers: running time of all algorithms (including the adversary) and success probability.

"Efficient" = PPT, and "Small"= *negligible*.

Template for **asympototic** security A scheme is *secure* if for every *probabilisitic polynomial-time* adversary $\mathscr{A}$ carrying out an formally specified attack, the probability that $\mathscr{A}$ *succeeds* in the attack is *negligible*.

More about asymptotics.

Definition 9. A *probabilistic polynomial time* (PPT) algorithm is an algorithm with access to an infintely long random tape, which for all inputs $x \in \{0,1\}^*$ and random tapes halts within $p(|x|)$ steps for some polynomial p. A PPT algorithm A is said to compute $f : \{0,1\}^* \to \{0,1\}^*$ with probability q if for all inputs $x \in \{0,1\}^*$, we have $\Pr[A(x) = f(x)] \geq q$.

A *negligible* function is one that decreases asymptotically faster that any inverse polynomial function.

Definition 10. A function $f : \mathbb{N} \to \mathbb{R}$ is *negligible* if for every positive polynomial p, there is an N such that for all integers $n \geq N$ it holds that $f(n) < \frac{1}{p(n)}$.

Example 11. $2^{-n}, 2^{-\sqrt{n}}, n^{-\log n}$ are all negligible.

We usually denote an arbitrary negligible function by negl. Rule of thumb: n^5 adversary succeeds with prob. at most $2^{-0.5n}$.

In practice need concrete security. Asymptotic does not guarantee security for small n, but usually can be translated into concrete security.

3.1 Defining computational secrecy

Informally, no additional info. can be learned by *efficient* adversary except with *negligible* probability. This was formalized as semantic security, the first definition for computational secrecy analogous to Def. 2 for perfect secrecy, by Goldwasser and Micali. But it is more complex and difficult to work with. We adapt the indistinguishablity formulation with our relaxations (proposed in the same paper and was proven equivalent to semantic security later [MicaliRacoffSloan]).

Setup: goal & threat model (eavesdropping one ciphertext).

FS NOTE: Emphasize again threat model only specifies ability but not the strategies.

Distinctions from *Perfect* indist:

- security parameter n.
- challenge messages $|m_0| = |m_1|$, otherwise no constraint on length (of course polynomial if $\mathcal{A}$ runs in polynomial time).

FS NOTE: Draw a game diagram

1. Adversary is given input 1^n, and $\mathcal{A}$ outputs a pair of messages m_0, m_1 with $|m_0| = |m_1|$.
2. CH generates a key $k \leftarrow G(1^n)$, and a uniform $b \leftarrow \{0,1\}$. Compute *challenge ciphertext* $c \leftarrow E_k(m_b)$ and give to $\mathcal{A}$.
3. $\mathcal{A}$ outputs a bit b' as the guess of b.
4. Define $\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n)$ the output of the experiment to be 1 if $b' = b$, and 0 otherwise. We call $\mathcal{A}$ succeeds if $\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n) = 1$.

Figure 3: Adversarial indistinguishability experiment $\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n)$

Definition 12 (KL-Def. 3.8)**.** A private-key encryption scheme $\Pi = (G, E, D)$ is *computationally secret* or has *indistinguishable encryptions in the presence of an eavesdropper* (EAV-secure in short) if for all probabilistic polynomial-time adversaries $\mathcal{A}$ there is a negligible function negl such that, for all n

$$\Pr[\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n) = 1] \leq 1/2 + \mathrm{negl}(n),$$

where the probability is take nover the randomness of $\mathcal{A}$ and the randomness in the experiment (choosing the key and bit b, as well as any randomness in E).

Encryption and Plaintext length: [KL: p56 & Excercise 3.2] (in HW).