COM S 687 Introduction to Cryptography January 24, 2008

Lecture 2: Secure Communication

Instructor: Rafael Pass *Scribe: Ben Kraft*

1 Review

The previous lecture introduced us to cryptography and the classic problem of how to make a message incomprehensible to eavesdroppers but still readable by those with some secret information. We wish to formalize this problem. Two parties, Alice and Bob, wish to communicate privately over an insecure channel monitored by an eavesdropper, Eve. To defend against Eve, Alice and Bob will encrypt their messages with a private key encryption scheme.

Definition 1 $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ *is called a private key encryption scheme over message space* $\mathcal{M}$ *and key space* $\mathcal{K}$ *if:*

1. Gen *is a randomized key-generation algorithm, where* $k \leftarrow \mathrm{Gen}$ *represents* Gen *sampling a key* $k \in \mathcal{K}$.
2. Enc *is an encryption algorithm, either randomized or deterministic, where* $c \leftarrow \mathrm{Enc}_k(m)$ *represents* Enc *yielding ciphertext* c *for key* $k \in \mathcal{K}$ *and message* $m \in \mathcal{M}$.
3. Dec *is a deterministic decryption algorithm where* $m \leftarrow \mathrm{Dec}_k(c)$ *represents* Dec *accepting key* $k \in \mathcal{K}$ *and ciphertext* c *and yielding plain-text message* m.
4. Dec *is the inverse of* Enc. *Formally, for all* $m \in \mathcal{A}$ *and* $k \in \mathcal{K}$,

$$\Pr[k \leftarrow \mathrm{Gen} : \mathrm{Dec}_k(\mathrm{Enc}_k(m)) = m] = 1$$

Alice and Bob meet in advance, run Gen to obtain a key, and use an encryption scheme to communicate over the insecure channel. But how can they be sure the encryption scheme works and Eve cannot read their messages? This requires a definition of secrecy.

2 Secrecy

2.1 Defining Secrecy

Possible ways to define secrecy; Eve should:

1. *Not be able to recover the key or any information about the key.* This approach fails because we might be able to recover the message without learning any information about the key. Consider a trivial encryption scheme where Enc is the identity function and discards the key and returns the original message as the ciphertext. There is no way for Eve to recover the key, but its trivial for her to recover the message.
2. *Not be able to recover a single bit of the message.* Eve, however, might be able to recover probabilistic information about the bit's distribution.
3. *Not be able to recover any probabilistic about any bit.* But what if Eve already knows some information about the message?
4. *Not be able to recover any probabilistic information besides what she knows a-priori.*

2.2 Perfect Secrecy

Definition 2 *An encryption scheme* (Gen, Enc, Dec) *is* perfectly secret *if for every* $m_1, m_2 \in \mathcal{M}$ *and for all c,*

$$\Pr[k \leftarrow \text{Gen} : \text{Enc}_k(m_1) = c] = \Pr[k \leftarrow \text{Gen} : \text{Enc}_k(m_2) = c]$$

That is, the probability of getting ciphertext c is the same for all possible messages in $\mathcal{M}$. Since c carries no information about the message, this definition holds even if Eve has A-priori information.

The Caesar and substitution ciphers introduced in the first lecture are perfectly secret for one-letter messages, but not perfectly secret for any larger messages.

2.3 One-time-pad: a perfectly secret encryption scheme

$$\begin{aligned}
\mathcal{M} &= \{0,1\}^n \\
\mathcal{K} &= \{0,1\}^n \\
\text{Gen} &= k \in \{0,1\}^n \\
\text{Enc}_k(m_1m_2...m_n) &= c_1c_2...c_n \qquad \text{where } c_i = m_i \oplus k_i \\
\text{Dec}_k(c_1c_2...c_n) &= m_1m_2...,m_n \qquad \text{where } m_i = c_i \oplus k_i
\end{aligned}$$

Proposition 1 *OTP is perfectly secret.*

Proof. *Consider any message* $m \in \{0,1\}^n$ *and any encryption* $c \in \{0,1\}^n$.

Claim 1: *If* $c \in \{0,1\}^n$*, then* $\Pr[\mathrm{Enc}_k(m) = c] = 2^{-n}$.

Claim 2: *If* $c \notin \{0,1\}^n$*, then* $\Pr[\mathrm{Enc}_k(m) = c] = 0$.

If $c \in \{0,1\}^n$ *and* $\mathrm{Enc}_k(m) = m \oplus k$*, then for every* c *and* m*, there exists a single unique key s.t.* $m \oplus k = c \Rightarrow k = m \oplus c$*, and there is a* 2^{-n} *chance* Gen *will pick this key.*

If $c \notin \{0,1\}^n$*, there is no chance* Enc *will generate* c *for any* k*.*

Since the above holds for any m*, it holds for any pair* $m_1, m_2 \in \mathcal{M}$*,* $|m_1| = |m_2| = n$*, and*

$$\Pr[k \leftarrow \mathrm{Gen} : \mathrm{Enc}_k(m_1) = c] = \Pr[k \leftarrow \mathrm{Gen} : \mathrm{Enc}_k(m_2) = c]$$

as required by perfect secrecy. ■

So OTP obtains perfect secrecy but requires very long keys and would be awkward to use in many situations. The next result shows this to be a requirement of perfect secrecy, and the key length used by OTP to be optimal.

Theorem 1 *If an encryption scheme* $(\mathcal{M}, \mathcal{K}, \mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ *is perfectly secret, then* $|\mathcal{K}| \geq |\mathcal{M}|$*.*

Proof. *Assume* $|\mathcal{K}| < |\mathcal{M}|$*, and pick any message* $m_1 \in \mathcal{M}$ *and key* $k \in \mathcal{K}$*. Let* $c = \mathrm{Enc}_k(m_1)$*. Since* $|\mathcal{K}| < |\mathcal{M}|$*, there must exist some message* $m_2 \in \mathcal{M}$ *s.t.* $m_2 \neq \mathrm{Dec}_{k'}(c)$ *for any key* $k' \in \mathcal{K}$*. Thus*

$$\Pr[k' \leftarrow \mathrm{Gen}, \mathrm{Enc}_{k'}(m_2) = c] = 0$$

and since we know $\mathrm{Enc}_k(m_1) = c$*,*

$$\Pr[k' \leftarrow \mathrm{Gen}, \mathrm{Enc}_{k'}(m_1) > c] = 0$$

so any encryption scheme with $|\mathcal{K}| < |\mathcal{M}|$ *cannot be perfectly secret.* ■

This gives us a valid attack on any encryption scheme with $|\mathcal{K}| < |\mathcal{M}|$. For any ciphertext c:

1. enumerate all keys.
2. encrypt all possible messages with all possible keys $k \in \mathcal{K}$.
3. If a message does not encrypt to c for any key $k \in \mathcal{K}$, we can rule it out.

At first glance it seems we need perfect secrecy; otherwise an adversary can learn information about our message. But at what cost? The running time of this algorithm is exponential in $\mathcal{M}$ and $\mathcal{K}$, and its unlikely more than a few parties in the world could afford to run it for reasonable $\mathcal{M}$'s and $\mathcal{K}$'s. For this reason we will restrict ourselves to efficient adversaries who cannot perform this sort of brute-force attack.

3 Computationally Bounded Adversaries

Algorithm = TM, RAM, etc ... , where input/output is over a string of $\Sigma = \{0, 1\}$.

Algorithm A **computes** f if A on input x returns $f(x)$.

Running Time :

A **runs in time** $t(n)$ if for all x, $A(x)$ halts in $t(|n|)$ steps.

A **runs in *polynomial time*** if there exists a constant c s.t. A runs in time $t(n) = n^c$.

Efficient algorithms run in polynomial time. Why?

1. Because this is how others define(d) this.
2. Poly. time is closed under composition.

Easy Problems : Mult, Div, GCD, ModExp, etc....

Hard Problems :

Halting Problem : proved undecidable over TM's.

Time hierarchical Theorem

SAT [1] and other NP complete problems (the set of problems reducible to it) are conjectured to be hard on the assumption $NP \neq P$.

Randomized Algorithm : a TM with a random tape, or a "button" for a random coin flip. A randomized algorithm A runs in time $t(n)$ if for all x, $A(x)$ halts within $t(|x|)$ steps regardless of the random tape.

PPT Probabilistic polynomial time algorithm.

A randomized algorithm A **computes** f if for all x,

$$\Pr[A(x) = f(x)] = 1$$

A randomized algorithm A **computes** f **with probability** p if for all x,

$$\Pr[A(x) = f(x)] = p$$

Proposition 2 *If a PPT A computes f w.p. $\frac{2}{3}$, then there exists a PPT A' s.t. A' computes f w.p. $1 - 2^{-n}$.*

A' will be constructed by repeatedly running A and returning the majority output. The proof will use the Chernoff bound and methods similar to those used in the first homework.

[1] determine whether a satisfying assignment exists for a set of Boolean formulas