Crypto!!!

Contents

Some Notations and Terminology.

Suppose Alan wants to send Mona a message m over an insecure **channel**.

Some Notations and Terminology.

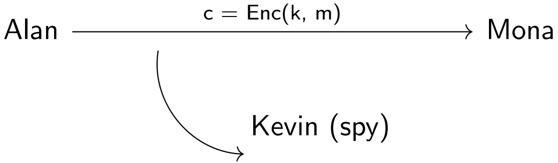
Suppose Alan wants to send Mona a message m over an insecure **channel**. The message m is going to be encrypted with a key k.

Some Notations and Terminology.

Suppose Alan wants to send Mona a message m over an insecure **channel**. The message m is going to be encrypted with a key k. The encoding/encryption of m using k is denoted by Enc(k, m).

Some Notations and Terminology.

Suppose Alan wants to send Mona a message m over an insecure **channel**. The message m is going to be encrypted with a key k. The encoding/encryption of m using k is denoted by Enc(k, m).



Problem

Suppose Kevin is spying on the channel between Alan and Mona. In order to encrypt the messages, Alan and Mona have to agree on a key for encrypting and decrypting their messages. But with Kevin spying on the channel, how do they send each other a key?

Problem

Suppose Kevin is spying on the channel between Alan and Mona. In order to encrypt the messages, Alan and Mona have to agree on a key for encrypting and decrypting their messages. But with Kevin spying on the channel, how do they send each other a key?

Solution. Computational complexity!!

Fact.

Suppose you're given a prime p, and you pick some number a modulo a prime p. Kevin is given the equation

$$a^x \equiv d \pmod{p}$$

and asked to solve for x. Can he do it?

Fact.

Suppose you're given a prime p, and you pick some number a modulo a prime p. Kevin is given the equation

$$a^x \equiv d \pmod{p}$$

and asked to solve for x. Can he do it?

This is called the **discrete log problem**.

Fact.

Suppose you're given a prime p, and you pick some number a modulo a prime p. Kevin is given the equation

$$a^x \equiv d \pmod{p}$$

and asked to solve for x. Can he do it?

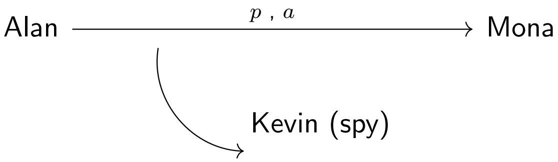
This is called the **discrete log problem**.

Turns out that the discrete log problem is actually super hard, in fact it is NP hard. So unless Kevin can solve NP hard problems, I would wager a guess that he has no chance of solving the problem. But we still love him anyway.

Back to our problem.

Back to our problem. Alan sends a prime p and a number $a \in \{1, \ldots, p-1\}$ to Mona.

Back to our problem. Alan sends a prime p and a number $a \in \{1, \ldots, p-1\}$ to Mona.



Alan generates a random number m in $\{1, \ldots, p-1\}$.

Alan generates a random number m in $\{1, \ldots, p-1\}$.

$$
\begin{tikzcd}
\text{Alan} \arrow[rr, "a^m \pmod{p}"] \arrow[dr, bend right] & & \text{Mona} \\
& \text{Kevin (spy)} &
\end{tikzcd}
$$

Mona generates a random number n in $\{1, \ldots, p-1\}$

Mona generates a random number n in $\{1, \ldots, p-1\}$

$$
\begin{tikzcd}
\text{Alan} & & \text{Mona} \arrow[ll, "a^n \pmod p"'] \\
& \text{Kevin (spy)} &
\end{tikzcd}
$$

What information do they have?

1. **Alan.** $m, a^n \pmod p$ — so he can figure out $a^{mn} = (a^n)^m \pmod p$

What information do they have?

1. **Alan.** $m, a^n \pmod p$ — so he can figure out $a^{mn} = (a^n)^m \pmod p$
2. **Mona.** $n, a^m \pmod p$ — so she can figure out $a^{mn} = (a^m)^n \pmod p$

What information do they have?

1. **Alan.** $m, a^n \pmod p$ — so he can figure out $a^{mn} = (a^n)^m \pmod p$

2. **Mona.** $n, a^m \pmod p$ — so she can figure out $a^{mn} = (a^m)^n \pmod p$

3. **Kevin.** $a^m, a^n \pmod p$ — but he doesn't know m or n, so he can only figure out $a^m \times a^n = a^{m+n}$. Better luck next time Kevin :(

What information do they have?

1. **Alan.** $m, a^n \pmod p$ — so he can figure out $a^{mn} = (a^n)^m \pmod p$

2. **Mona.** $n, a^m \pmod p$ — so she can figure out $a^{mn} = (a^m)^n \pmod p$

3. **Kevin.** $a^m, a^n \pmod p$ — but he doesn't know m or n, so he can only figure out $a^m \times a^n = a^{m+n}$. Better luck next time Kevin :(

So Alan and Mona can now use $a^{mn} \pmod p$ as their key, and Kev would be none the wiser.

Formalizing Encryption — Syntax

1. Key Space K.

Formalizing Encryption — Syntax

1. Key Space K.
2. Message space M.

Formalizing Encryption — Syntax

1. Key Space K.
2. Message space M.
3. Ciphertext space C.

Formalizing Encryption — Syntax

1. Key Space K.
2. Message space M.
3. Ciphertext space C.

$$\mathsf{Enc} : K \times M \to C$$

$$\mathsf{Dec} : K \times C \to M$$

Formalizing Encryption — Correctness

For all keys $k \in K$, messages $m \in M$, we have Dec(k, Enc(k, m)) $=$ m.

Formalizing Encryption — Semantic Security

- Plaintext comes from an arbitrary distribution

Formalizing Encryption — Semantic Security

- Plaintext comes from an arbitrary distribution
- Adversary initially has some information about the plaintext

Formalizing Encryption — Semantic Security

- Plaintext comes from an arbitrary distribution
- Adversary initially has some information about the plaintext
- Seeing the ciphertext should not reveal any more information

Formalizing Encryption — Semantic Security

- Plaintext comes from an arbitrary distribution
- Adversary initially has some information about the plaintext
- Seeing the ciphertext should not reveal any more information
- Model unknown key by assuming it is chosen uniformly at random

Imitiation Game.

Sometimes it is the people no one imagines anything of who do the things that no one can imagine. - Alan Turing, re Kevin (?)

Substitution Cipher.

1. Key space $K = S_{26} =$ permutations of $\{a, \ldots, z\}$

Substitution Cipher.

1. Key space $K = S_{26} =$ permutations of $\{a, \ldots, z\}$
2. Message space $M = \{a, \ldots, z\}^*$

Substitution Cipher.

1. Key space $K = S_{26}$ = permutations of $\{a, \ldots, z\}$
2. Message space $M = \{a, \ldots, z\}^*$
3. Ciphertext $C = \{a, \ldots, z\}^*$

Substitution Cipher.

1. Key space $K = S_{26} =$ permutations of $\{a, \ldots, z\}$
2. Message space $M = \{a, \ldots, z\}^*$
3. Ciphertext $C = \{a, \ldots, z\}^*$

Encoding and Decoding are both obvious, and it's clear that correctness holds.

Substitution Cipher.

1. Key space $K = S_{26}$ = permutations of $\{a, \ldots, z\}$
2. Message space $M = \{a, \ldots, z\}^*$
3. Ciphertext $C = \{a, \ldots, z\}^*$

Encoding and Decoding are both obvious, and it's clear that correctness holds.

VB V NVRRWO LC CVKR, U RJUFE RJVR EWYUF UB V XFXBXVQ EUFZ LC GWOBLF
Hint: Y = V

Transposition Cipher

1. Key space $K = S_n =$ permutations of $\{1, \ldots, n\}$
2. Message space $M = \{a, \ldots, z\}^n$
3. Ciphertext $C = \{a, \ldots, z\}^n$

Transposition Cipher

1. Key space $K = S_n =$ permutations of $\{1, \ldots, n\}$
2. Message space $M = \{a, \ldots, z\}^n$
3. Ciphertext $C = \{a, \ldots, z\}^n$

Use the inverse permutation to decrypt!

One Time Pad

.

1. Key space $K = \{0,1\}^n$
2. Message space $M = \{0,1\}^n$
3. Ciphertext $C = \{0,1\}^n$

One Time Pad

.

1. Key space $K = \{0, 1\}^n$
2. Message space $M = \{0, 1\}^n$
3. Ciphertext $C = \{0, 1\}^n$

Use a different substitution for each character, never use the same key twice.

One Time Pad is extremely secure. But what happens if we use the same key twice?

One Time Pad is extremely secure. But what happens if we use the same key twice?

$$\begin{aligned}\mathsf{Enc}(k, m_0) - \mathsf{Enc}(k, m_1) \\ = (k + m_0) - (k + m_1) \\ = m_0 - m_1\end{aligned}$$

This is sufficient to recover both messages.

Perfect Semantic Secrecy.

Definition. A scheme (Enc, Dec) is perfectly semnantically secure if, for all:

Perfect Semantic Secrecy.

Definition. A scheme (Enc, Dec) is perfectly semnantically secure if, for all:

- Distributions D on M (Plaintext distribution)

Perfect Semantic Secrecy.

Definition. A scheme (Enc, Dec) is perfectly semnantically secure if, for all:

- Distributions D on M (Plaintext distribution)
- Functions $I : M \to \{0,1\}^*$ (Info Kevin (the adversary) gets)

Perfect Semantic Secrecy.

Definition. A scheme (Enc, Dec) is perfectly semnantically secure if, for all:

- Distributions D on M (Plaintext distribution)
- Functions $I : M \to \{0,1\}^*$ (Info Kevin (the adversary) gets)
- Functions $f : M \to \{0,1\}^*$ (Info Kevin tries to learn)

Perfect Semantic Secrecy.

Definition. A scheme (Enc, Dec) is perfectly semnantically secure if, for all:

- Distributions D on M (Plaintext distribution)
- Functions $I : M \to \{0,1\}^*$ (Info Kevin (the adversary) gets)
- Functions $f : M \to \{0,1\}^*$ (Info Kevin tries to learn)
- Functions $A : C \times \{0,1\}^* \to \{0,1\}^*$ (Mathematical description of Kevin/adversary)

Perfect Semantic Secrecy.

Definition. A scheme (Enc, Dec) is perfectly semnantically secure if, for all:

- Distributions D on M (Plaintext distribution)
- Functions $I : M \to \{0,1\}^*$ (Info Kevin (the adversary) gets)
- Functions $f : M \to \{0,1\}^*$ (Info Kevin tries to learn)
- Functions $A : C \times \{0,1\}^* \to \{0,1\}^*$ (Mathematical description of Kevin/adversary)

There exists a function (called the **simulator**) $S : \{0,1\}^* \to \{0,1\}^*$ such that

$$\Pr[A(\mathsf{Enc}(k,m), I(m)) = f(m)]$$

$$= \Pr[S(I(m)) = f(m)]$$

where probabilities are taken over $k \in K, m \in D$.

What are the two main issues with semantic security?

What are the two main issues with semantic security?

1. We need Kevin in the definition.

What are the two main issues with semantic security?

1. We need Kevin in the definition.

2. The description is unnecessarily complicated (see Kevin)

More Notation

Let us say two random variables X, Y over a finite set S have identical distributions. Then we write

$$X \overset{\mathsf{d}}{=} Y$$

Perfect Shannon Secrecy (Shannon '49)

A scheme (End, Dec) has perfect secrecy if, for any two messages $m_0, m_1 \in M$ we have

$$\mathsf{Enc}(k, m_0) \stackrel{\mathsf{d}}{=} \mathsf{Enc}(k, m_1)$$

Both sides contain a random variable over uniform distribution of the key k.

Semantic Security = Perfect Security

A scheme (Enc, Dec) is semantically secure if and only if it has perfect shannon secrecy.

Semantic Security = Perfect Security

A scheme (Enc, Dec) is semantically secure if and only if it has perfect shannon secrecy.

Proof. Boring probability bash, see lecture slide.

"VB V NVRRWO LC CVKR, U RJUFE RJVR EWYUF UB V XFXBXVQ EUFZ LC GWOBLF"

Solution.

"As a matter of fact, I think that Kevin is a unusual kind of person."

What's Next?

1. Randomized Encryption
2. Limitations of information theoretic security
3. Pseudorandom generators