Security Issues in Cloud Computing

Introduction to Cryptography

Peter Schwabe

October 7, 2011

What is cryptography?

The word cryptography comes from Greek “kryptos” (κρυπτός) for “secret” and graphein (γράφειν) for “to write”.
Traditionally cryptography deals with *encrypting* messages, i.e. transforming message *plaintexts* into *ciphertexts* under the use of a *key.* This is done in a way that makes it easy to obtain the message plaintext from ciphertext and key, but hard (impossible) to obtain the message plaintext from the ciphertext without knowledge of the key.
Cryptanalysis deals with breaking an encryption, that is obtaining the plaintext from the ciphertext without knowledge of the key. *Cryptology* is the area that encompasses cryptography and cryptanalysis.
Modern cryptography addresses broader topics than this:

> “Cryptography is the study of mathematical techniques related to aspects of information security such as confidentiality, data integrity, entity authentication, and data origin authentication.” (Menezes, van Oorschot, Vanstone: Handbook of Applied Cryptography).

In cryptology we usually use the following communication model: Alice (A) communicates with Bob (B). Eve is a passive eavesdropper of this communication (she does not send anything, she does not modify anything). Oscar is a potentially active opponent (attacker, adversary) who may send messages or modify messages.

Confidentiality: Ensure that only Bob (and not Eve) can read Alice’s messages.

Data integrity: Ensure that Bob receives exactly the messages sent by Alice, and not a message modified by Oscar. Think, for example of software downloads from the internet.

Entity authentication: Ensure that Alice is really who she claims to be (and not Oscar saying “I’m Alice”). Think of logins.

Data-origin authentication: Ensure that Bob can be sure that a message has indeed be sent by Alice (and, if desired, also give him the possibility to prove this to others).

A simple cipher: Caesar's cipher (100-44 BC)

Encrypt messages by shifting the alphabet. For example, shift by 3:
`ABCDEFGHIJKLMNOPQRSTUVWXYZ` maps to
`DEFGHIJKLMNOPQRSTUVWXYZABC`

Plaintext message:	`THISISSECRET`
Key: 3	
Ciphertext:	`WKLVLVVHFUHW`

How to break this cipher?

There are just 26 keys (one being highly insecure), we can just try them all.

Why did the system work more than 2000 years ago?

Enemies did not know how the system works, they might have thought it was some text in a foreign language. Generally this approach to security ("security by obscurity") is a bad idea.

Kerckhoffs' principle

Auguste Kerckhoffs stated in 1883 that a cryptosystem should be secure even if everything about the system, except the key, is public knowledge. This has various advantages:

- Keys are generally easy to change, systems or algorithms are not (think of hardware implementations);
- public knowledge is required for standardization;
- all modern cryptosystems follow Kerckhoffs' approach;
- secret services still do not reveal the systems they are using.

Another way to describe Caesars Cipher

Write the alphabet as $0, 1, 2, \ldots, 25$ (for A,B,C,D.,Z)
Encryption: - Add the key
- Substract 26 if the result is larger than 25
More general: *Modular Arithmetic*
Consider the set of integers $\{0, 1, 2, , n - 1\}$

Addition:	- Add two numbers - Subtract n if the result is Larger than $n-1$
Subtraction:	- Subtract two numbers - Add n if result is negative
Multiplication:	- Multiply two numbers - Subtract n until the result is smaller than n. (the second step is the same as taking the remainder of a division by n)

For this kind of arithmetic *modulo* n we use the following notation:

$$\begin{aligned} a+b &\mod n \\ a-b &\mod n \\ a\cdot b &\mod n \end{aligned}$$

Better than Caesars cipher: Substitution cipher

For better security we need more keys (make exhaustive search infeasible).
Idea: Replace each letter of the alphabet by another letter.

Example:
`ABCDEFGHIJKLMNOPQRSTUVWXYZ` maps to
`WXICSRYAEVBUKODLGTZFPYQHMN`
Keys are permutations of the alphabet. Some keys are obviously insecure (e.g., the unpermuted alphabet)
Number of keys: $26 \cdot 25 \cdot 24 \cdot \ldots 2 \cdot 1 = 26! = 403291461126605635584000000$
Exhaustive search is infeasible, even on a modern computer!

How to break this system?

Idea: Letters appear with different frequencies in natural language, so count the frequencies of letters in the ciphertext and draw conclusions about plaintext letters.
Example: E is the most frequent letter in English, followed by T,A,O,I,N. In a long ciphertext obtained from the above permutation (key), the most frequent letter is probably is S, followed by F,W,D,E,O.
More advanced: Analyze bigrams, trigrams (frequent sequences of 2 or 3 letters) e.g., “AND”, “IS”

Better than Substitution Cipher: Vigenère Cipher (1523-1596)

Defense against frequency analysis needs to map the same plaintext letters to different ciphertext letters.
Idea: Add keyword to plaintext.
Example: Encrypt the plaintext “CRYPTOLOGY” with the keyword “BOARD”:

$$\begin{array}{cccccc|ccccc}
 & C & R & Y & P & T & O & L & O & G & Y \\
\oplus & B & O & A & R & D & B & O & A & R & D \\
\hline
 & D & F & Y & G & W & P & . & . & . &
\end{array}
\qquad
\begin{array}{cccccc|c}
 & 2 & 17 & 24 & 15 & 19 & 14 \\
\oplus & 1 & 14 & 0 & 17 & 3 & 1 \\
\hline
 & 3 & 5 & 24 & 6 & 22 & 15
\end{array}$$

How to break this system?

Assume that the attacker knows the length k of the key. Write the ciphertext $(c_1, c_2, \ldots, c_n)$ as matrix:

$$\begin{pmatrix} c_1 & c_2 & \cdots & c_k \\ c_{k+1} & c_{k+2} & \cdots & c_{2k} \\ \vdots & & & \vdots \\ c_{sk+1} & & & c_n \end{pmatrix}$$

Each column is encrypted by the same keyword letter (as in Caesers cipher), we can apply frequency analysis columnwise.
Babbage and Kasiski independently found a method to estimate the key length of Vigenère ciphertext. Their idea was the following: Frequent equal sequence of letters in the ciphertext probably come from equal sequences in the plaintext. Key length is divisor of the distance of these sequences.
Example: The text "To be or not to be" encrypted with the Vigenère cipher with key `HAM` leads to:

	T	O	B	E	O	R	N	O	T	T	O	B	E
	H	A	M	H	A	M	H	A	M	H	A	M	H
⊕	A	O	N	L	O	D	U	O	F	A	O	N	L

The sequence AONL appears twice in the ciphertext, both stem from the same plaintext words (TOBE). The distance of the two sequences is 9, the key length is 3.

Better than Vigenère: Vigenere with running key

Choose key of the same length as the message length (e.g., specify a book, page, start word). This is called *running key*, or *key stream.*

How to attack this system?

The 6 most frequent letters in English E,T,A,O,I,N appear with frequency 51% for both plaintext and key. More than 26% of the ciphertext characters stem from combinations of these 6 letters. We can apply joint or parallel frequency analysis to obtain plaintext and key.

Better than Vignère with running key: Vernam (1917)

The idea of Vernam's cipher is the following:

- Choose a key of the same length as the message.
- Dont take key from natural language, but choose each letter of the key of random (with the same probabilities for each letter), uniform distribution.
- Never use the same key twice ("one-time pad")

Advantage Information-theoretic proof of perfect secrecy for fixed-length messages (Shannon, 1949)

Disadvantages

- Key generation: True randomness is hard to get.
- Key distribution: Need to securely transmit a key of the same length of the message, this is not easier than securely transmitting the message.

The one-time pad is usually used by transmitting a huge amount of random letters once (e.g. hard disk). Afterwards messages can be securely encrypted until all random letters have been used as key.
The Vernam cipher has been used by secret services during the Cold War (and is maybe still used by them today).

Types of attacks and attackers

Known-ciphertext attack: Attacks considered so far. Attacker only knows ciphertext(s), and tries to obtain knowledge about the plaintext (or key).

Known-plaintext attack: Attacker knows plaintext-ciphertext pairs, tries to obtain knowledge that can be used to decipher other ciphertexts (e.g. the key).

Chosen-plaintext attack: Attacker gets access to an "encryption oracle", i.e., he can generate ciphertexts from plaintexts. Target: typically the key. Two variants of this kind of attack:

- Non adaptive: attacker chooses the messages, encrypts them all, then access to the oracle is removed.
- Adaptive: Attacker can choose messages depending on ciphertexts he previously obtained from the oracle.

Chosen-ciphertext attack: Attacker has access to a "decryption oracle", he can generate plaintexts from ciphertexts. Target: typically the key.
Again: Non-adaptive and adaptive versions.

All systems of todays lecture can be broken by all of the above attacks (except Vernam's cipher). Generally, adaptive chosen-ciphertext attacks are hardest to defend.