CS2107 Tutorial 2 (Encryption)

E.-C. Chang, School of Computing, NUS

August 13, 2025

1. You have intercepted two ciphertexts C_1, C_2 generated by a stream cipher using the same secret key. The first 4 bits of the ciphertext form the IV.

$$\begin{aligned} C_1 &= 0111\ 11011011 \\ C_2 &= 0111\ 00101011 \end{aligned}$$

   You know that the plaintext must be among the following 4 sequences:

$$P_1 = 00000000, P_2 = 11111111, P_3 = 00001111, P_4 = 11000011$$

   What are the possible plaintexts of C_1 and C_2?

2. (*Meet-in-the-middle*) Instead of applying DES three times, Bob wants to apply it four times with 4 different 56-bit keys k_1, k_2, k_3 and k_4. By using meet-in-the-middle attack, what is the of number of cryptographic operations (including encryption and decryption) required for known-plaintext attack? Give your answer in the form of 2^k and approximation (within a multiplicative factor of 2) is suffice. (Remark: Lecture note mentioned that there is a more efficient meet-in-the-middle attack. Here, the simple meet-in-the-middle in the lecture note is suffice).

3. Alice sends instructions to Bob daily using mobile phone in the following way. Each instruction is represented as ASCII string, and follows the format:

$$\textit{action}{:}\textit{date}$$

   The date is the 8-byte "dd/mm/yy" format, and actions are "`buy`", "`sell`", "`sell_everything`", and "`hold_and_see`". Example

$$\texttt{buy:02/01/22}, \qquad \texttt{sell\_everything:03/01/22}.$$

   The message will be using AES under some mode-of-operation. The ciphertext, which is a binary string, is then converted to a text message using some tools, e.g. `uuencode`. The text message is then sent to Bob using SMS.

   The mobile phone, after each re-start, will set the IV to be the string of all zeros, and then increases it by one (i.e. treat it as binary number and increment by 1, similar to the CTR mode) for every new encryption. An attacker is able to sniff the SMS channel between Alice and Bob.

   Let us consider two settings:

(a) Suppose the mode-of-operation is CBC mode. What information regarding the plaintext can be inferred by the attacker? Note that under CBC mode, message padding is required if the message length is not multiples of 16-byte. In this question, the padding is simply done by appending bytes of zeros at the end of the message string.

(b) Suppose the mode-of-operation is CTR mode. What information regarding the plaintext can be inferred by the attacker?

4. (*Padding Oracle*) Consider the padding oracle attack described in the lecture note. Suppose the attacker knows that the 16-byte plaintext is the sequence $\langle b_1, b_2, b_3, b_4, b_5, b_6, b_7, b_8, b_9, 00, 00, \texttt{FF}, \texttt{04}, \texttt{04}, \texttt{04}, \texttt{04}\rangle$, where the numbers are in hexadecimal representation, and the attacker does not know the value of the b_i's. Describe how the attacker determine the value of b_9. In particular, describe how to decide the value of v in the lecture note.

5. The lecture notes assume the attacker knows the number of padding bytes. Now, consider a scenario where this information is unknown. Suppose the attacker has access to the one-block IV and a two-block ciphertext, but does not know how many padding bytes are present. Describe a method the attacker can use to determine the number of padding bytes, using as few oracle queries as possible.

6. (*Padding oracle attack is practical*) Search the CVE database for a known vulnerability that is based on AES-CBC padding oracle attack. (note: there are also padding oracle attack on other encryption scheme).

7. Find out more about these terminologies:
   end-to-end encryption
   Does Whatapps provide end-to-end encryption? Wechat? The live CS2107 online lecture? Zoom meeting?

Hands-on Exercise: Using OpenSSL for primitive crypto operations

Command line of OpenSSL (`https://www.openssl.org/`) is a useful tool for the assignment.

OpenSSL is a full-featured toolkit for the Transport Layer Security (TLS) and Secure Sockets Layer (SSL) protocols, and contains a comprehensive cryptography library. The toolkit comes with the **openssl command-line binary**, which supports a wide range of cryptographic operations.

In other words, an user could invoke the openssl command-line tool, and then ask the tool to carry out some cryptographic operations such as encryption. So, ironically, although it is implemented for SSL, we could use it to carry out basic crypto operation even when we are not using SSL. This is kind of "over-killed" but convenient.

Installation. MacOS should already have OpenSSL preinstalled. For Linux, follow the installation step below.

To install the OpenSSL binary toolkit, install the OpenSSL package using the following command:
`$ sudo apt-get install openssl`
Refer to the following documentation on installation of OpenSSL on Ubuntu: `https://help.ubuntu.com/community/OpenSSL`.

Simple task. Once the package is installed, try running the following command to test the installed OpenSSL and check its version:
`$ openssl version`
To list all available OpenSSL sub-commands, run:
`$ openssl help`

Then, run the following openssl command to benchmark the system's performance on all cryptographic algorithms:
`$ openssl speed`
Based on the output, answer the following question: "Which one is faster: RSA signing operation or verification operation?"

More info. To find out the details of various cryptographic-related OpenSSL operations, read the following "OpenSSL Command-Line HOWTO": `https://www.madboa.com/geek/openssl/`. Refer to the following manual page of various openssl's (sub) commands: `https://www.openssl.org/docs/manmaster/man1/`.