C2107 Tutorial 2 (Encryption+Password)

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

February 9, 2021

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?

Solution

This is an example of the ciphertext xor-ing attack described in Lecture 1:
$C_1 \oplus C_1$ (by omitting the IV) $= P_2 \oplus P_3 = 11110000$.
Notice that the attack is applicable because: (a) a stream cipher is employed; (b) the same secret key and IV are used for generating the two ciphertexts.

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 is suffice.

Solution

For each pair of k_1 and k_2, we need 2 encryptions and 2 decryption. There is a total of 2^{112} pairs. Using a straightforward method of applying 2 encryptions and 2 decryptions for each pairs, the overall is 2^{114} cryptographic operations.
We can do slightly better that that. One can exhaustively enumerate k_1, and for each k_1, exhaustively search all k_2. So, the number of encryptions will be (number of encryptions using k_1) + (number of encryptions using k_2) $= 2^{56} + 2^{112} \approx 2^1 12$. We need the same number of operations for k_3 and k_4. So total is approximately $2^{112} \times 2 = 2^{113}$.
Remarks:

(a) If applied 3 times, meet-in-the-middle needs approximately 2^{112} operations. So, increase from 3 to 4 times only increase the "difficulty" from 2^{112} to 2^{113} (or "bit-strength" from 112 to 113).

(b) What about applying $2t - 1$ times vs applying it $2t$ times, when $t = 3, 4, \ldots$.ß

3. (*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 b_9. In particular, describe how the value of v in lecture 1 slide 69 is to be computed.

Solution

$v = IV \oplus \langle 0, 0, 0, 0, 0, 0, 0, 0, t, 08, 08, \texttt{F7}, \texttt{0C}, \texttt{0C}, \texttt{0C}, \texttt{0C}\rangle$ and output $t \oplus 08$. That is,

(a) For t= 0 to FF

(b) let $v = IV \oplus \langle 0, 0, 0, 0, 0, 0, 0, 0, t, 08, 08, \texttt{F7}, \texttt{0C}, \texttt{0C}, \texttt{0C}, \texttt{0C}\rangle$.

(c) send $v \| c$ to oracle.

(d) If `yes`, output $t \oplus 08$.

To understand the algo, you might want to draw the decryption process (i.e. the flow chart of $D_k(c) \oplus v$), assume b_9 is some value (e.g. 04) and then trace the loop.

4. Consider two password authentication systems S-I and S-II:

   S-I. After a user has entered a userid, the system sleeps for 0.5 second, and then checks whether the userid is in the database. If it is not in, the system sleeps for another 0.5 second, displays an error message, and then prompts for the new userid; Otherwise, the system prompts for the password. If the password is wrong, the system sleeps for another 0.5 second and then prompts for the new userid.

   S-II. After a user has entered a userid and password, the system sleeps for 0.5 second, and then checks whether the information is correct. If it is wrong, the system sleeps for another 0.5 second, display an error message, and then prompt for the new userid and password.

   What are the security implications?

   Solution

   Suppose an attacker needs x guesses for the userid, and y guesses for the password, in order to log in as a valid user. The attacker will thus need $x+y$ guesses on S-I, whereas he will need $x \cdot y$ guesses on S-II.
   (*Note:* Notice that, in practice, userid is not a secret. But in a layered defense, it is good to hide it as much as possible. In the worst case scenario that the attacker is able to get all public information, the security still relies on the password's strength.)

5. (*Online vs offline attack.* *To be discussed next week, after SHA3 is covered in lecture.*) You are assessing the system [S-II] in question 2.

   (a) The system recommends having password of at least 10 alphanumeric (including uppercase and lowercases) characters. Based on the guideline by RFC4086 (lecture note), is 10 sufficient?

   (b) After further investigation, you found that the verification is a protocol between a prover and a verifier.

      i. A prover asks for userid u and password p from the user. Next, the prover sends u to the verifier.
      ii. The verifier randomly generates a 128-bit string r and sends r to the prover.
      iii. The prover computes $h = SHA3(r\|p)$ and sends h to the verifier.
      iv. The verifier checks that indeed $h = SHA3(r\|p)$. If so, then the prover is "authentic".

      You also found that potentially, an eavesdropper can obtain the above interactions. Hence, the eavesdropper can get r, u, h. Now, is the 8-character p sufficient?

      (optional: what is the role of r?)

6. (*Usability vs Security*) A university library provides a web-based service for the students to renew books. To get authenticated, a student keys in (a) student ID, (b) date of birth in DDMMYYYY format, and (c) family name. After authenticated, the list of books borrowed by the student is displayed, and the student can choose which book to be renewed. No other action can be performed through this service.

   (a) What are the advantages of the above compare to the typical password authentication?

   (b) What are the weaknesses of the above system? Are there any concern on privacy (note that beside book information, there is another subtle leakage of personal information)? Do you prefer the university using the above, or the typical password authentication?

Solution

(a) There is no bootstrap process needed to setup user passwords. Additionally, the pieces of information needed are easy to remember.

(b) There is a potential privacy leak:

   i. If an adversary knows the social information, then he can log-in to see the books being borrowed;

   ii. If the adversary knows both student ID and family name, then he can probe the login screen to find out the birthday.

   (Notice that the National Library Board used to have such a system for many years. Now, a password-based system is used.)

(c) It is a trade-off between usability and security.

7. (*Security Analysis: comparing two systems*)

   An IT team is planning to deploy password+sms 2FA for an online-banking service. To use the service, a user first logs-in using the password (without the sms) via a PC. After the user has logged in, the user's account number would be displayed on the PC, together with a few options. The connection from the PC to server is through HTTPS[1]. If the user

[1] We would study HTTPS later. HTTPS is supposed to be secure, that is, the server is authentic, and no sniffing and spoofing of the channel. Nonetheless, while not common, there are still circumstances that it got compromised. In this security analysis, you can also include scenarios where an attacker is able to act as a man-in-the-middle. Also note that a malware in PC is stronger than an attacker who compromised the channel, and thus not necessary to consider that.

wants to transfer money to another account, the following steps are to be carried out:

(a) The user enters transaction information (account number and amount) to the PC, which in turns sends the information to the server.

(b) The server sends a OTP to the user via sms. The sms will be delivered to the user mobile phone by the telecommunications service provider (eg. Singtel).

(c) The user enters the OTP to the PC, which in turn sends the OTP to the server.

(d) After received confirmation from the server, the PC displays a message "transaction completed".

Now, the IT team has to decide what information to be included in the sms in step (b). Below are examples of two choices:

M-I "Enter OTP: 132373"

M-II "You have requested to transfer $10,000 from account 1388293-43-23 to the account 12398-234-A2, enter OTP: 132373".

- Give a situation where M-II is preferred. *(consider an attack scenario where the browser could be malicious, e.g. the user is using a PC in an Internet cafe to carry out the transaction.)*
- Give another situation where M-I is preferred. *(use the fact that sms are not encrypt "end-to-end", and there are untrusted entities who can sniff the sms.)*

You are in the IT team. Which would you choose?

Solution

Before answering the questions, notice that arguing "*a system M_1 is more secure than M_2*" requires us to find an attack that M_1 can prevent, but not M_2 .
Also pay attention to the meaning of "end-to-end encryption", that is only Alice & Bob can know the original message.

(a) *M-II is more secure*: (Integrity/Authenticity) if (1) a malware resides is planted in the untrusted PC or (2) an adversary successfully redirects the user to a spoofed webpage, then the message displayed to the user can be different from the transaction actually sent to the server. M-II can help the user verifies if his/her money is transferred correctly.

(b) *M-I is more secure*: (Confidentiality) If the mobile phone is lost or somehow an adversary can get hold of the mobile phone, and the previous messages are not deleted, then the adversary can see the undeleted transaction details. Alternatively, since SMS does not provide end-to-end encryption, potentially some adversaries can tap into the SMS communication channel and capture the transaction history.

(c) You can decide on the message format. How about showing partial account information?

8. A company has installed a fingerprint door access system for their server room, and gym. The two systems are the same, but the company can set different thresholds to adjust the FNMR/FMR (lecture 2). Suppose the threshold for the server room is set at 0.5, would a reasonable threshold for the gym be larger, smaller, or equal to 0.5?

Solution

Smaller than 0.5 in order to be more accepting than the server room.
Recall again that when threshold is 0, the system accepts everyone; and when it is 1, the system accepts no one, i.e. rejects all.

9. Recently, many mobile phone use face detection to unlock the phone. Discuss why face detection is preferred over password. Is it more "secure"?

Solution

(a) **Usability:** It is more convenient. User doesn't need to remember, and no need to key in.

(b) **Security:** An additional "attack vector" for attacker to access the phone. Thus, not more secure, and possibly less secure.

Suppose face recognition replaces password and is the only way to get access, or is combined with password (i.e. password correct, and the correct face), then,

(a) **Security:** It provides an additional factor of "who-you-are", on the assumption that attackers are unable to spoof face, or hack the physical devices and feed images to the camera.

(b) **Usability:** There would be false reject.

10. Find out more about these terminologies:
Graphical Passwords, covert channel, side channel attack, end-to-end encryption

Hands-on Exercise: OpenSSL Installation on Your Linux Host

Last week, you were already asked to set up a Linux host. In this week, you will install **OpenSSL** (`https://www.openssl.org/`) on your Linux host so that you can use `openssl` command.

OpenSSL is a full-featured toolkit for the Transport Layer Security (TLS) and Secure Sockets Layer (SSL) protocols, and is also **a general-purpose cryptography library**. The library additionally comes with the **openssl command-line binary**, which allows you to handily perform a wide range of cryptographic operations.

To install the OpenSSL binary toolkit, install the OpenSSL package using the following command:
`$ sudo apt-get install openssl`
If needed, you may also refer to the following documentation on how you can install OpenSSL on Ubuntu: `https://help.ubuntu.com/community/OpenSSL`.

Once the package is installed, you can try running the following command to test the installed OpenSSL and check its version:
`$ openssl version`

To list all available OpenSSL sub-commands, you can run:
`$ openssl help`

Then, run the following openssl command to benchmark your 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?"

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

If you have any issues and need help with your Linux set-up and OpenSSL installation, your TAs will open an open consultation session after releasing Assignment 1 later. Please do your own self exploration first in setting up your Linux system and installing OpenSSL.