C2107 Tutorial 1 (Intro & Encryption)

School of Computing, NUS

February 1, 2021

Remark: Not all questions would be covered by the instructors during tutorial.

1. Alice was the Web administrator of the company VIC[1]. A malicious attacker sent an email to Alice. The email instructed Alice to click on a link so as to login to the HR's system to view a report. In the email, information on the "sender" indicated it was from the HR manager in VIC. Alice wrongly believed that the email was indeed sent by the manager and followed the instructions. In doing so, she revealed her password to the attacker. Using Alice's password, the attacker logged-in to the web-server, and invoked many processes. As a result, the server was overloaded.

   With respect to the security requirements mentioned in the lecture (confidentiality, integrity, authentication, availability, etc), discussed what aspects of security were compromised.

Solution

The violated security aspects and offending actions are as follows.

Confidentiality:

A3. Alice revealed her password.

Authenticity:

A1. The attacker spoofed the email.

A2. Alice visited and interacted with the spoofed website specified in the link.

A4. The attacker logged-in to the Web server.

Availability:

A6. The Web server got overloaded.

Integrity:

A5. The attacker invoked many processes on the Web server (*Remark*: a violation of the server's process integrity).

2. Suppose it takes 512 clock cycles to test whether a 64-bit cryptographic key is correct, when given a 64-bit plaintext and the corresponding ciphertext.

   (a) How long does it take to exhaustively check all the keys using a 4 GHz (single-core) processor?

   (b) How long does it take on a cluster of 1024 servers, each with a quad-core 4Ghz processor.

   (Hint: For simplicity, you can take 1 year $\approx 2^{25}$ seconds. We follow the notations where 1K $= 2^{10}$, 1M $= 2^{20}$, 1G $= 2^{30}$.)

   Solution for 1st part

   Notice that a 4GHz processor has $2^2 \cdot 2^{30} = 2^{32}$ cycles per second.
   From the problem description, testing 1 key takes $512 = 2^9$ cycles.
   In 1 second, the processor can thus check $2^{32} / 2^9 = 2^{23}$ keys.
   To check all 2^{64} keys, the processor needs $2^{64} / 2^{23} = 2^{41}$ seconds.
   Since 1 year $\approx 2^{25}$ seconds, the total time needed is therefore: $2^{41} / 2^{25} \approx 2^{16} \approx 2^6 \cdot 2^{10}$ years $\approx$ 64K years.

   Solution for 2nd part

   Given 1024 servers, each with a quad-core processor, we thus have $1024 \cdot 4 = 2^{10} \cdot 2^2 = 2^{12}$ processors.
   The total time needed is now reduced by a factor of 2^{12} to become: $\approx 2^{16} / 2^{12} \approx 2^4 \approx 16$ years.

3. Suppose it takes 512 clock cycles to test whether a 36-bit cryptographic key is correct, when given a plaintext m and the corresponding ciphertext c. Length of plaintext irrelevant in this question.

   How long does it take to exhaustively check all the keys using a 4GHz (single-core) processor? Using exhaustive search, is it then possible to crack a ciphertext and obtain its plaintext in *realtime*, say within 0.1 second?

Solution

Notice that a 4GHz processor has $2^2 \cdot 2^{30} = 2^{32}$ cycles per second.
From the problem description, testing 1 key takes $512 = 2^9$ cycles.
In 1 second, the processor can thus check $2^{32} / 2^9 = 2^{23}$ keys.
To check all 2^{36} keys, the processor needs $2^{36} / 2^{23} = 2^{13} = 8192$ seconds ≈ 137 minutes.
The plaintext thus cannot be recovered in realtime.
(*Note*: Nevertheless, the calculation shows a 36-bit key can be broken rather easily by the processor).

A walkie-talkie system *realtime Secure Walkie Talkie* (rSWT)[1] secures its communication using symmetric keys encryption. rSWT uses two encryption schemes, AES block cipher, and another fast stream cipher developed by the company called FAST. The cipher FAST is really fast, but its key length is only 32 bits.

During installation, the user enters k, a 128-bit *master key*, into each walkie-talkie. After installation, when a walkie-talkie wants to send a plaintext m to another, the sent signal is computed in the following ways:

(a) A 36-bit v is randomly chosen.

(b) Computes $t = \mathrm{AES}_{\mathrm{enc}}(k, v)$, where $\mathrm{AES}_{\mathrm{enc}}$ is encryption of AES block cipher (without mode of operation) and v is padded with zeros.

(c) Obtains $\widetilde{k}$, which is the first 36 leading bits of t. This $\widetilde{k}$ is called the *temporary* key.

(d) Computes $c = \mathrm{FAST}_{\mathrm{enc}}(\widetilde{k}, \mathbf{0}_{64} \| m)$, where $\mathbf{0}_{64}$ is a string of 64 zeros, $\|$ is string concatenation, and $\mathrm{FAST}_{\mathrm{enc}}$ is the deterministic encryption of FAST. Note that c does not contain initial value.

(e) Sends $(v \| c)$ over the air.

We consider attackers who can eavesdrop the ciphertexts (both v and c) transmitted over the air.

We know that 36-bit is too short and the key can be broken, but, as calculated before, it would take very long time. In their marketing efforts, rSWT claims that 36-bit is sufficient for realtime applications. This is what appeared in their advertisement:

> "*36-bit is sufficient. By the time the message is maliciously decrypted, the message becomes useless*".

Now, you want to design a hand-held device that is able to crack and obtain the plaintext in *realtime*. Specifically, when given the v and c, the device should derive the 36-bit session key readily within 0.1 second. The

[1] Companies are fictional.

hand-held device can have computing resource comparable to a laptop. Suggest a way to get the temporary key.
(Hint: Use storage to help. Here, we assume that the hand-held device can hold a large, say 1TB, of pre-computed table whereby the key can be looked up. We use the notations where 1K = 2^{10}, 1M = 2^{20}, 1G = 2^{30}.)

Solution

Tradeoff the time with space, i.e. solving the problem in less time by using more storage space, as follows:
Construct a table of $\text{FAST}_{\text{enc}}(\widetilde{k}, 000\ldots000)$ for all possible 2^{36} values of $\widetilde{k}$. Notice that there are 2^{36} entries in the table, where each entry is 64-bit long as the result of encrypting the 64-bit of zeros. Hence, the derived table will take $2^{36} \cdot 64$ bits $= 2^{36} \cdot 8$ bytes $= 2^{36} \cdot 2^3 = 2^9 \cdot 2^{30} bytes = 512$GB.
To break a SWT communication, first extract the first 64 bits of the captured ciphertext, then perform a lookup operation on the constructed table in order to determine the employed $\widetilde{k}$.
Given $\widetilde{k}$, the attacker can perform a decryption process using the stream cipher, thus recovering the plaintext in realtime.

Remark:

(a) It is not necessary to store $\widetilde{k}$.

(b) To further reduce space, we don't need to store the full 64-bit of ciphertext. First 40 bits of the ciphertext is sufficient.

(c) There is another technique know as *time-space tradeoff* to further tradeoff time with space. Yet another technique *Rainbow table* further improve the efficiency.

4. Lecture 1 mentioned that Winzip encrypts the compressed file. Why it is meaningless to carry out the two operations in the other way, that is, encrypts the file, and then compresses the encrypted file?
(Hint: Consider the effectiveness of compression on "random" sequences, and a requirement of encryption scheme.)

Solution

Compressing an encrypted file will yield very little or no compression gain.
This is since the encrypted file will resemble a "random" sequence (due to a requirement of a good encryption scheme).
A compression algorithm, which takes advantage of repeating patterns, therefore will not work well on an encrypted file.

5. Bob encrypted a music mp3 file using Winzip, which employs the 256-bit key AES. He chose a 6-digit number as password. Winzip generated the 256-bit AES key from the 6-digit password using a (deterministic) function, say SHA1.

   Alice obtained the ciphertext. Alice also knew that Bob used a 6-digit password and knew how Winzip generated the AES key.

   (a) Give a 256-bit string, can Alice determine whether this string was indeed the correct AES key?

   (b) How many guesses did Alice really need in order to get the mp3 file?

   Solution

   Despite the use of the 256-bit AES key, the total number of guesses needed is only $10^6 = 1$ million.
   (Note: Why not 2^{256}?)

6. Compare Symantec Internet Security Threat Report in 2019 and 2009. Discuss what are new and what remain over 10 years. (Open-ended discussions. No right or wrong. The link for 2009 in lecture note is broken. Try `http://www.securityprivacyandthelaw.com/uploads/file/symantec%202009.pdf`)

7. Find out more about these terminologies:

   (a) *Cryptology, Cryptanalysis, Cryptography,*

   (b) *NSA, NIST, cryptography backdoor, Key Escrow, Decryption order*

   Find out more about the following well-known persons in cryptography: *Whitfield Diffie, Ron Rivest, Alice, Bob, Eve, Mallory and Trent.*

   (optional) Can NSA break AES? Can NSA by-pass cryptography?