Ýmir Þórleifsson

HW10

# 10.1.1

Two users A and B have the following privileges:

* A can read and write a file f, and read, write, and execute a service program p.
* B can only read f and execute p.

An intruder has managed to perform one of the violations:

* Guess A's password.
* Guess B's password.
* Place a passive wiretap on A's terminal line and intercept all data communication, except for the password, which is protected through encryption.
* Place an active wiretap on A's terminal line, which allows the interception of all communication, expect the password, but also allows insertion of arbitrary new messages into the communication stream.

1. For each of the 4 cases, describe a simple scenario, if one exists, that would lead to each of the following violations:
2. Information disclosure.

The intruder could leak all information he gathered.

1. Information modification or destruction.

The intruder could write to the service program or destroy from it.

1. Unauthorized use of services.

The intruder could read f and execute p.

1. Denial of service.

The intruder could delete file f and program p.

# 10.2.2

An OS uses salting to prevent password guessing. Encrypting one password takes h μs. Encrypting one password with salt takes 10 times longer. Looking up and comparing a value in the password file takes c μs. The password file has n entries. An intruder is using a dictionary of m words to try to guess a valid name/password combination.

1. Determine the time to check if any of the dictionary words is a valid password of any user:

* without salt

h\*m\*c μs.

* with salt

10h\*m\*c\*n μs.

1. Compute the actual values for the two cases when h = 1 μs, c = 0.01 μs, n = 1000, and m = 100,000.

1\*100,000\* 0.01 μs = 1,000 µs.

1,000 µs \* 10n = 10,000,000 µs.

# 10.2.3

The one-way function H(c) = c³ mod 100, where c is the decimal value of an ASCII character, is used to generate a series of one-way passwords. H is applied separately to each character of a password.

1. Starting with the initial password "Cat," generate a sequence of 5 one-time passwords.

Pwd 1: ?I`T

Pwd 2: /$

Pwd 3: 8@

Pwd 4: Ca,

Pwd 5: ?I`T

# 10.2.5

A biometric authentication system using fingerprints returns a value between 0 and 1 for each attempted match. The system was tested with 1000 genuine fingerprints and 1000 imposter fingerprints. The histogram shows the numbers of genuine and imposter attempts for different ranges.

Ex: 88 of the 1000 genuine fingerprints returned a value of n between 0.4 and 0.5, while only 3 imposter fingerprints returned a value in the same range.

|  |  |  |
| --- | --- | --- |
| Range of n | Imposter fingerprints | Genuine fingerprints |
| 0 ≤ n < 0.1 | 643 | 0 |
| 0.1 ≤ n < 0.2 | 319 | 1 |
| 0.2 ≤ n < 0.3 | 29 | 15 |
| 0.3 ≤ n < 0.4 | 5 | 49 |
| 0.4 ≤ n < 0.5 | 3 | 88 |
| 0.5 ≤ n < 0.6 | 1 | 133 |
| 0.6 ≤ n < 0.7 | 0 | 158 |
| 0.7 ≤ n < 0.8 | 0 | 176 |
| 0.8 ≤ n < 0.9 | 0 | 188 |
| 0.9 ≤ n ≤ 1 | 0 | 192 |

1. Determine the threshold value of n such that less than 1% of imposter attempts are accepted. How many false alarms are generated as a result?

n = 0.3, 16 false alarms.

1. Determine the threshold value of n such that less than 1% of genuine attempts are rejected. How many imposter attempts are accepted as a result?

n = 0.2, 38 imposter attempts.

1. Determine the threshold value of n such that no genuine attempts are rejected. How many imposter attempts are accepted as a result?

n = 0.1, 357 imposter attemps.

# 10.3.1

A system consists of 200 domains and 10,000 objects, for a total of 2,000,000 entries in the access matrix.

* 95% of the objects are accessible from only a single domain.
* 4% of the objects are accessible from an average of 3 domains.
* 1% of the objects are accessible from all domains.

1. Determine the number of entries necessary if the matrix is implemented as a set of access lists.

100\*200 + 400\*3 + 9500\*1 = 30,700.

1. Determine the number of entries necessary if the matrix is implemented as a set of capability lists.

30,700.

# 10.3.2

A system consists of 4 domains, D1 through D4, and 3 objects, O1 through O3. Processes in the 4 domains need to access the resources as follows:

* D1 owns, reads, writes, and executes O1
* D1 reads and executes O3
* D2 reads and writes O1
* D2 owns, reads, writes, and executes O2
* D3 executes O2
* D4 reads O1
* D4 reads O2
* D4 owns, reads, writes, and executes O3

|  |  |  |  |
| --- | --- | --- | --- |
|  | O1 | O2 | O3 |
| D1 | rwxo |  | rx |
| D2 | rw | rwxo |  |
| D3 |  | X |  |
| D4 | r | r | Rwxo |

1. Show the necessary permissions as access lists.

Read lines.

1. Show the necessary permissions as capability lists.

Read columns.

1. Assume that all x-rights are marked with the c-right.

Determine in which domain a process can enable processes in domain D1 to:

* read O2

D2, D3.

* execute O2

D2, D3.

# 10.4.1

The following is a simple encryption/decryption scheme for character strings in ASCII code:

* The plaintext and the ciphertext are divided into blocks of n ASCII characters.
* The encryption/decryption key K is a string of n ASCII characters.
* The encryption and decryption functions are both the bitwise exclusive OR function, XOR, applied to the blocks of ASCII characters and the key:

E(P, K) = XOR(P, K) and D(C, K) = XOR(C, K), where P is a block of n plaintext characters, C is a block of n ciphertext characters, and K is the key.

1. Decrypt the ciphertext TF%TO6QE5GEv using the key 5$W.

5$W = 00110101 00100100 01010111

XOR(TF%, 5$W) = 01100001 01100010 01110010 = abr

XOR(TO6, 5$W) = 01100001 01101011 01100001 = aka

XOR(QE5, 5$W) = 01100100 01100001 01100010 = dab

XOR(Gev, 5$W) = 01110010 01100001 00100001 = ra!

Message: abrakadabra!

1. If one encryption takes 0.01 μs, determine the time needed to try all possible keys on a ciphertext of 30 characters.

30 \* C(127, 3) \* 0.0000001 = 0.1 s

1. How long would the key have to be to increase the search time to more than one month?

s in month = 2,628,000

C(127, x) \* 30 \* 0.0000001 >= 2,628,000

C(127, x) >= 876,000,000,000

Test x=4, 10,334,625 >= 876,000,000,000 FALSE

Test x=5, 254,231,775 >= 876,000,000,000 FALSE

Test x=6, 5,169,379,425 >= 876,000,000,000 FALSE

Test x=7, 89,356,415,775 >= 876,000,000,000 FALSE

Test x=8, 1,340,346,200,000 >= 876,000,000,000 TRUE

Key would have to be 8 characters.

# 10.4.2

The following is a simple public-key cryptosystem for character strings in ASCII code:

* The decryption function is P = C⁵ mod 35. The corresponding encryption function C = Pᵉ mod 35 is kept secret.
* Each hexadecimal code of an ASCII character has 2 digits. Each digit of the pair is encrypted separately using the above function.

Ex: The hexadecimal code for the ASCII character B is 42. The two digits 4 and 2 are encrypted as (4ᵉ mod 35, 2ᵉ mod 35).

1. Decrypt the ciphertext:

(9,4)(32,0)(6,15)(7,7)(6,10)(32,0)(7,4)(6,15)(7,10)(32,0)(32,9)(33,1)(32,17)(33,0)(33,0)(33,0)(32,17)(33,0)(33,0)(33,0)(32,1).

(The parentheses and the commas are not part of the ciphertext but are used only to increase readability.)

(9,4) = mod(95, 35) = 4, mod(45, 35) = 9 🡪 “I”

Doing this to the whole sequence gives the following result:

“I owe you $1,000,000!”

1. The RSA code is based on 2 principles:

* n is the product of 2 prime numbers, p and q
* e \* d mod (p - 1) \* (q - 1) = 1

Break the above simple encryption scheme by finding the corresponding encryption function by using the two underlying principles of the scheme.

p = 7

q = 5

e \* d mod 24 = 1

e = 5

d = 5

# 10.4.3

A message with the text "Let's be friends," has been received, along with the digital signature (23, 6, 17, 6). The message was sent by a user, whose public decryption function is P = C⁷ mod 35. The signature was produced using the following known conventions:

* The digest d of the message is produced by dividing the text into blocks of 4 characters. The decimal values corresponding to the 4 ASCII characters are added together modulo 10.
* The digital signature is produced by encrypting each digit of the digest with the sender's private function C = Pᵉ mod 35, where e is kept secret.

1. Determine whether the received message is genuine and could not be repudiated by the sender.

“Let’” “s be” “ fri” “ends”

mod(332,10) = 2 🡪 mod(27,35) = 23. Checks out.

mod(346,10) = 6 🡪 mod(67,35) = 6. Checks out.

mod(353,10) = 3 🡪 mod(37,35) = 17. Checks out.

mod(426,10) = 6 🡪 mod(67,35) = 6. Checks out.

Message is genuine.