HY-457

Assignment 2 - Tutorial

Deyannis Dimitris
Spiridakis Konstantinos
Arakas Ioannis

Assignment 2.1

In this assignment you are asked to implement a simple bash script

- This script must contain an **ssh invocation** but at the same time **“steals”** user’s password

- Host must be given as input parameter while calling the script in the cmd
  e.g my_ssh.sh <*destination*>

- Your script must behave like normal ssh but will be a bit slower

Assignment 2.1

Key ideas

- Monitor system calls and their arguments.
  Use strace(1) to complete this step
  e.g strace <*command* or *process*>

- Filter and parse monitored system calls and find which of them may contain the password
  - You are free to use any methods necessary

- Store the password (not in plain sight)

Assignment 2.1

<u>This part will NOT be graded</u>
!! For this part, locking home directory and spare is required !!
Suppose that you managed to inject *my_ssh.sh* in a user's directory.

The .bashrc file is a script file that is executed when a user logs in. The file itself contains configurations for setting up or enabling coloring, auto-completion, shell history, **command aliases** and more. The main function of this file is loading the terminal preferences and environmental variables and saving them.

Try to set your *my_ssh.sh* as an **alias** for the original ssh so that when the user calls ssh from the command line, *my_ssh.sh* is called instead and the credentials are stolen.

Assignment 2.1

- Open and edit .bashrc (hidden file) and add an alias for ssh
- alias ssh= */path/to/your/my_ssh.sh*
- To enable the changes `$source .bashrc`
- Run ssh in cmd
- When this part is complete do not forget to remove the ssh alias

Assignment 2.2

In this assignment you need to Implement a simple Intrusion Detection System (IDS)

Your code should do the following:

- Read a .txt file that contains all of your rules
- Read .pcapng file for traffic
- Detect packets that match one or more rules of the txt file
- Alert the user

Assignment 2.2

Your rules should have the following form:

- <src IP address> <src port> <dst IP address> <dst port> “ALERT MESSAGE”

For example:

- 192.168.1.2 55 192.168.1.7 55 “this is one rule”
- 192.168.1.0/24 8080 192.168.1.0/24 1234 “this another rule”

Assignment 2.2

CIDR Notation

- 192.168.1.0/24 is equivalent to 192.168.1.* or 192.168.1.0 - 255
- In more detail, 192.168.1.0/24 in binary is 11000000.10101000.00000001.*
- The first 24 bits stay the same and the last 8 can take any value between 00000000 - 11111111

Assignment 2.2

Reading .pcapng files

- Use pcap_loop() from pcap library
- int pcap_loop(FilePointer, 0, packetHandler, NULL)
- void packetHandler(u_char *useless, const struct pcap_pkthdr* pkthdr, const u_char* packet)

Assignment 2.2

- packetHandler(... , … , const u_char* packet) where *packet* points to the first u_char of the header.

- You will need to parse the header to get the source ip, source port, destination ip and destination port

- The headers you need to parse are:
  - ethernet
  - ip
  - tcp

Assignment 2.3

- In this assignment you need to implement a secret sharing mechanism between three entities: Alice, Bob and Carol

- The three people will share a secret number “S” by each taking a piece of the number

- Only when all three pieces are presented then all of them are able to reconstruct the secret number “S”

Assignment 2.3

- This will be achieved by constructing the polynomial f(x) = a*x^2 + b*x + c
  - “c” will be the secret number “S”
- Each will take a point of the polynomial
  - Alice: (1, f(1))
  - Bob: (2, f(2))
  - Carol: (3, f(3))
- When all 3 points are presented, they reconstruct the polynomial to retrieve “c” which is the secret number “S”
- If the secret has to be reconstructed by K entities, the polynomial degree must be K-1

Assignment 2.3

Example

- Let's assume that Alice and Bob want to share the secret number 72
- Since they are 2, the polynomial degree is 1. The chose a randomly to be 14 and b is the secret number
  - f(x) = a*x + b => f(x) = 14*x + 72
- They calculate f(1) = 86 and f(2) = 100
- Alice gets (1, 86) and Bob (2, 100)

Assignment 2.3

Example (cont)

- To reconstruct the secret, they present their points and reconstruct the polynomial

$$\left.\begin{matrix} 86 = a + b \\ 100 = 2a + b \end{matrix}\right\} \left.\begin{matrix} a = 86 - b \\ 100 = 2(86 - b) + b \end{matrix}\right\} \left.\begin{matrix} 100 = 172 - 2b + b \end{matrix}\right\} \left.\begin{matrix} -72 = -b \end{matrix}\right\} \mathbf{b = 72}$$

Assignment 2.3

Develop a program called *secret_sharing* using C that does the following:

1. Using the *split* option and a secret number the program generates the 3 points

   e.g. $ ./secret_sharing split 72
   > (1, f(1)), (2, f(2)), (3, f(3))

2. Using the *join* option and the 3 points, the program reconstructs the secret number

   e.g. $ ./secret_sharing join (1, f(1)), (2, f(2)), (3, f(3))
   > 72

3. Generalize your solution for N friends. When **any three** present their points they are able to reconstruct the secret

Assignment 2.4

- In this assignment, you need to develop an anonymous voting method using the previous approach

- For this reason, “c” in **f(x) = a*x^2 + b*x +c** will be each person’s vote where c == 1 means “yes” and c == 0 means “no”

- Alice, Bob and Carol create a polynomial each
  - A(x) = a2 * x^2 + a1 * x + a0
  - B(x) = b2 * x^2 + b1 * x + b0
  - C(x) = c2 * x^2 + c1 * x + c0

Assignment 2.4

- Alice computes A(1) and gives A(2) to Bob and A(3) to Carol
- Bob computes B(2) and gives B(1) to Alice and B(3) to Carol
- Carol computes C(3) and gives C(1) to Alice and C(2) to Bob
- Alice computes D(1) = A(1) + B(1) + C(1)
- Bob computes D(2) = A(2) + B(2) + C(2)
- Carol computes D(3) = A(3) + B(3) + C(3)
- Alice, Bob, and Carol contribute their points (1, D(1)), (2, D(2)), and (3, D(3)) to create polynomial D(x)= d2*x^2 + d1*x + d0. Note that d0 is equal to a0+b0+c0: the sum of their votes.

Assignment 2.4

- Write a program called *anon_voting* using C that implements this anonymous voting
- Generalize your program for N voters.
- In the README file, explain why d0 is the sum of the votes