Exercise 03

Feb 20, 2023

# Overview

This exercise provides hands-on experience with Cryptography to perform secure exchange of messages between two parties using symmetric key. Both techniques of Block Cipher and Stream Cipher will be used in this exercise.

# Learning Objectives

* Understand basics of Stream Cipher
* Understand working and implementation of RC4
* Ability to implement the algorithm and communicate message securely
* Ability to work in a small team

# Reading Material

1. Chapter 20, Principles of Computer Security 4th ed, Stallings and Brown
2. Using netcat to transfer data across network.

<https://www.digitalocean.com/community/tutorials/how-to-use-netcat-to-establish-and-test-tcp-and-udp-connections>

# Prerequisites and environment familiarity

Familiarity with Linux and command line terminal usage.

Familiarity with network communication between two systems connected via a network. This could be either using socket programming or using netcat (nc) to communicate across the network between two hosts

Familiarity with tcpdump (or wireshark) to capture and analyze the packets.

As a team of two you have access to 2 systems connected via a network. If you would like, you can work independently as well though it would be preferred to have a team. If you are not able to find your team mate, get in touch with Sachin (TA) and he will randomly assign the team mate to you.

# Description

This exercise involves sending a message securely from one system to another using RC4 stream cipher. Thus, this assignment involves a bit of programming (in your preferred programming language). You need to write a program to implement RC4 algorithm, Take the input data as a stream, encrypt the streaming data, send it to another machine, and where you decrypt the data.

## Assignment details

1. Make a team of two.
2. Agree upon the secret key among yourselves e.g. “Principles of Computer Security” or any other key as per your choice.
3. Assuming two machines are vm1 and vm2. So, we will assume that message (m) will be sent from vm1 to vm2 and reply (r) will be sent back from vm2 to vm1.
4. Consider that message m as m=”Hello, it is a nice sunny day and we should enjoy the weather” and reply r as r=”Busy with assignment right now.”.
5. On vm1, implement RC4 and send the message m to vm2. This can be done using socket programming. Note, if you aren’t familiar with socket programming, you can use nc (netcat) to transmit the data. Netcat can be used in the following way.
   * + On vm 2, run the following command.
       1. nc -l 12345
     + On vm 1, run the following command.
       1. nc “ip address of vm2” 12345

Whatever is typed on vm1 terminal will appear on vm2 terminal and vice versa.

Rather than typing on terminals, the program p1 on vm1 needs to communicate with program p2 on vm2 where prog p1 outputs on stdout and program reads data from stdin, i.e. p1 on vm1 sends data to p2 on vm2, then you invocatio of nc will invoke unix pipe redirection as given below.

* + - On vm 2, run the following command.
      1. nc -l 12345 | ./p2
    - On vm 1, run the following command.
      1. ./p1 | nc “ip address of vm2” 12345

The above mechanism provides redirection in one way only i.e. on vm2 from output of nc to input of p2, and on vm1 output of p1 to input of nc. However, if both program p1 and p2 needs to communicate in both direction, this can be done using creating pipes as given below

* + - On vm 2, run the following command.
      1. mkfifo vm2pipe
      2. nc -l 12345 <vm2pipe | ./p2 >vm2pipe
    - On vm 1, run the following command.
      1. mkfifo vm1pipe
      2. ./p1 <vm1pipe | nc “ip address of vm2” 12345 >vm1pipe

1. Assuming your RC4 implementation is done using python program “rc4s.py” (for sender) and "rc4cr.py" (for receiver) which takes key and the text as parameter, and you plan to use netcat (assuming you are not comfortable with network socket programming) then you can run the following command. This assumes that you know how to process the command line arguments in python. It is similar to using *main(int argc, char \*argv[])* in C, or using *getopt()* function in C. Command on vm1 would be as below (alternatively you can make use of socket programming in python to communicate and don’t need to use netcat). This assumings that unix pipe vm1pipe and vm2pipe have been created.
   * + 1. python rc4.py -k “key” -m ”msg” <vm1pipe | nc “ip addr of vm2” “port on vm2” <vm1pipe

Command on vm2 would be similar to

1. nc -l “port on vm2” <vm2pipe | python rc4r.py >vm2pipe

this should display the message received from vm1 after decryption.

1. Do a tcpdump capture on both vm1 and vm2 to ensure that encrypted message is being transmitted. For example, assuming application on vm2 is running on port 12345, the command to capture the dump on vm1 and vm2 respectively would be as follows.
   * + 1. tcpdump -n -i <ethernet interface> -w vm1capture.pcap host <vm2 ip address> and port <port on vm2>

Note: The ethernet interface would be like ens160.

* + - 1. tcpdump -n -i <ethernet interface> -w vm2capture.pcap host <vm1 ip address> and port <vm2 port>
      2. Once communication is done, press Ctrl-C (^C) to terminate wireshark capture.
      3. Download the program

## Explanation and Hints

1. Revise your network programming concepts. If you haven’t done network programming, get familiar with using netcat (nc).
2. Refresh/revise your wireshark/tcpdump skills. In case you haven’t used wireshark before, install/use the same and become familiar with it. tcpdump is preinstalled on linux.
3. First just output the key from RC4 sender and manually encrypt the data being sent character by character by XORing it with pseudorandom number key k.
4. Verify that same encrypted value is being transmitted over the network.

# Assessment and Rubric

Please do submit the following

1. Readme.txt file which will contain the following information
   1. Team information i.e. names of both team mates with their university id.
   2. Secret key being used.
   3. Commands invoked on both the vms for invoking the program.
   4. Challenges faced and how did you address these.
   5. Summary of your overall learning.
   6. References. Any website/resource that you used to took help.
2. Both the programs on vm1 and vm2 along with their respective tcpdump/wireshark capture.

**Rubric for assessment (20 marks)**

* 1. 4 marks for a Readme.txt file containing all the required information.
  2. 12 marks for Implementing RC4 algorithm and network communication.
  3. 4 mark for tcpdump/wireshark capture

# Note

Any plagiarism activity will result in penalties of being awarded 0 marks. If you are using the sample program as shown in the class, please attribute the same.

<end of Exercise 03>