CYBS
"Cybersecurity"

Practical Work 2: Security analysis and patching

In this practical session, you will be taking on the role of security professionals, charged with maintaining an inter company chat system. You have heard rumours that this chat system isn't secure, and that the company watches and documents all messages sent. Therefore, you have been charged with resolving these issues, thus still allowing you to use the chat system, without the security risk. However, the chat system is completely server based, meaning you have no control or influence over its functionalities, only on the data you send.

Working in groups, you represent the security interests and principals of your individual businesses. However, when it comes to security no one is truly alone, especially when it comes to a shared communication system. Thus, each group will need to collaborate with the others to propose different security solutions, which would still allow you to communicate, with added security protection.

The work here will be evaluated, so each group will be required to deposit a report on moodle. Only one member of each group needs to deposit the report, but don't forget to put the names of all members in the file. You are not required to provide any code, only the report.

1 To chat ... or not to chat

As stated previously, the chat system is completely server based, meaning all actions are performed remotely, with no need for client systems at all. We will simulate this with the server running on the professors laptop, linked to a Wi-Fi network that you will use to simulate your network connection. You will be given the name and password at the start of the session.

To use the chat, open a terminal and use the `telnet` command to connect to the server IP and port provided. You will see that the chat system possesses multiple commands, to change your username, see users online as well as send DM's to other users. Try it out now to see how it works. In a normal scenario, you won't have access to the logs or operations of the server. For this course, and to help you, all traces will be printed on the projector. Don't hesitate to ask if you wish to understand or know more about certain functionalities. When you are ready to move on, exit the `telnet` environment using `ctrl + x` or `ctrl + ]` depending on your system.

For your purposes, a client program written in Python can be found on Moodle. Download this file and check its functionality. Don't forget to set the IP and port number in the (__main__) function before proceeding. It is in this file that your security solutions will be tested.

2 Security audit

Firstly, in your individual groups, start by analysing the functionalities of the chat system. Present it in a short paragraph, as well as how this sort of system is useful for inter business communications.

Based upon your previous analysis, discuss the different security threats you have identified. In doing so, analyse how such a threat would be dangerous to your business, as well as to others.

3 Protecting communications

Here, we will take a look at a few encryption methods to help secure the communications between each other: `Encoding`, `Symmetric Encryption`, as well as `Asymetric Encryption`. Firstly, briefly present each of these methods based upon what we have seen in the course.

From here onwards, you will need to collaborate with all other groups in the elaboration of your security method.

3.1 Encoding

Begin by activating the `Encoding` methods provided in the Python script, by setting `METHOD = ENCODE` at the top of the file. There are many different encoding methods used in security, as well as network communications in general. An example is in web URLs, where special characters are "escaped" to prevent potential attacks (presence of % or
codes in many urls).

In the code provided, a basic encoding method is provided, using character translation. Here we shift each and every character of a message with another, for example a -¿ b, or a -¿ z, etc. In coordination with the other groups, define an encoding method (i.e. number of characters to translate) and set the variable accordingly at the top of the file `TRANSLATE = *put_your_number_here*`. For example `TRANSLATE = 1` corresponds to a -¿ b, `TRANSLATE = 2`, is a -¿ c and so on and so forth. Check the server output and analyse its contents. What is your opinion of this method? Do you foresee any limitations or issues with such a method?

3.2 Encryption

Now replace the method variable definition with `METHOD = ENCRPYT`. Find the section in your file marked `TODO: ENCRYPTION`, set `SYMETRIC = True` to activate symetric encryption and observe the contents. You should see four variables, each with different names. For now, we will only interest ourselves with the variable `GLOBAL_KEY`. Analyse what this key would do and how it would secure your system. Collaborate with the other groups to define a network key to encrypt all communications and test it out.

Analyse the contents on other groups terminals as well as the contents of the server logs. Do you see any limitations or restrictions with this approach. What would happen if one group had a different key, or none at all? Discuss how this limitation could be considered a problem

Now take a look at the other five variables. You should see three variables `PUB_KEYS`, `MY_PUB_KEY` and one variable `MY_PRIV_KEY`. How would these keys add extra security in your communications. Deactivate your previous solution with `SYMETRIC = False`, and set `ASYMETRIC = True`.

Collaborate with the other groups to define public keys for all groups as well as yourself. Define your own public key in `MY_PUB_KEY` and update the contents of `PUB_KEYS` following the commented format of a tuple, with the first entry the username of a group with the associated public key. It is important from here to note that changing you username will cause issues with encryption so please keep the same one. You will need to re-enter it at each connexion. Define also your own private key, which you should not share with anyone. Now send messages again. What do you see in the output of the server and why ?

How do you propose to solve this problem from a user stand point. No extra code is needed, simply your knowledge and understanding of this type of encryption.

Propose a final solution to protect all messages sent from the server, be them global or targeted. Implement the necessary changes in your file (normally only a single line to change) and test it out.

4 Authorisation and authentication

As you have seen, every-time you connect to the server a new username is assigned "User-X" with X always increasing. Propose a theoretical solution that would allow you to identify yourself upon connexion as well as the changes that would be needed on the server side.

Finally, what other methods could be implemented in the chat system, not only to identify the users, but regarding messages already sent?

5 Bonus – One way encryption

Encryption is generally two ways, allowing you to protect your data, all the while keeping the ability to recover the data when needed. However, in some cases, recovering the original value is considered a potential risk as the data is simply too sensitive. For this, we use methods such as `Hashing`, which convert a simply text or file into a unique hash value of varying size. This value is considered the *signature* of the data itself, and it is very rare to find two types of data with the same signature.

The command `md5sum` is part of the Linux operating system, and can be used to calculate the hash value of any type of data, from an input string to a file. Try hashing different texts with

the command `echo -n "put_text_here" | md5sum` and analyse the response. Now try hashing something else, such as the python code used previously with `md5sum client.py`.

Analyse the results and what this means for security. What areas can you see this sort of system being useful?