COMP8053 – Embedded Software Security

Assignment 2 - 50% of final grade.

Answer **all** questions. Submit your answers in a single pdf file through the submission facility on Canvas in "Assignments -> Assignment 1 Submission". The assignment is due at 23:59 on December 18th (end of week 12). **No penalty late submission until 23:59 December 25th 2020, after which a -20% penalty will apply for submissions until 23:59 January 1st 2021. Submissions after this date will receive a score of 0.**

For questions 1+2 you **must provide a description of the process taken to solve the problem**. This description should explain what actions you took and why you did them. What is important is not showing the correct answer, but showing that you understand why it is the correct answer. i.e. rather than merely showing the correct command you must use, explain what that command does, what information it provides you and how you can use that information to proceed further. Write as if your target audience is a computer scientist but with no security background (i.e. they know how to operate a computer and what the stack/heap are, but not what a buffer overflow attack is).

What you must do (for Q1+2):

You must document your approach clearly in the following way:

1) Provide a large paragraph, or two, which gives a high-level description of the approach you intend to take in order to achieve your attack. It should be clear and concise. If you started with one approach, but swapped midway to another after realising something, describe both the initial idea and your final one here too.

2) Show step-by-step how you performed your attack. You should include screenshots of your input/output (e.g. using the Windows "snipping tool") and provide short comments explaining **why** you did each action. For example:

   Sample Command (should be screenshotted, along with any relevant output included too): x\24x $esp
   Sample explanations for why:

   I used it to show the stack. **(BAD – this is what you did, not why you did it)**

   I used it to show the contents of 24 addresses on the stack, in order to identify the exact location of the buffer and calculate how much overflow was necessary to write over the saved base pointer. **(GOOD! – here the purpose of using a command to show the stack is explained!)**

3) **You must include a detailed diagram of the current state of the stack at each major step, starting from the initial state before your attack begins.** The diagrams should show all of the relevant elements on the stack (similar to those provided in the lecture notes).

4) You must show how you would change the code to **fix all the vulnerabilities** in the programs provided for Q2+3. Provide a brief description of why your changes fix the issues.

For questions 1+2, you will need to download the relevant C program file and copy it into the Protostar virtual machine. This can be done on Linux machines by using the "scp" (secure copy) command. For the Windows machines in the lab, PuTTY includes its own version of "scp" in the file "pscp.exe" located in the PuTTY installation directory. To use it, open a command prompt (start menu -> type in "cmd.exe") and navigate to the PuTTY installation directory. From there, type the following command:

```
pscp <filename> user@<ip address>:/home/user/
```

Where <filename> is the name of the C program file you are copying over, and <ip address> is the IP Address of the Protostar VM you are running (get it by typing "ip addr" inside the Protostar VM). This will create a copy of the file inside the /home/user/ directory on Protostar (the default starting directory).

Next you must compile the C program, with the following command:

```
gcc <filename> -o <outfile name>
```

Where <filename> is the name of the C program file, and <outfile name> is your name for the compiled binary. You can then run the compiled binary by typing:

```
./<outfile name>
```

Contact me if you have any difficulties in copying over and compiling the C programs! It is not intended that getting the files onto the VM are part of the challenge!

Question 1 – [30 Marks]

For this question, you will use the program “Question1.c” available in “Units -> Marked Assignment 2”. You must cause the program to run the “winner” function.

Question 2 – [20 Marks]

For this question, you will use the program “Question2.c” available in “Units -> Marked Assignment 2”. You must overcome the cruel marking scheme and make the program output “Grade of 100 assigned.” (Note: making it output “Grade of 100 assigned” does not guarantee you will get 100% on this assignment).

Question 3 – [10 Marks]

- ASLR is a defensive mechanism for software security. Explain how ASLR functions, and evaluate its effectiveness as a method to mitigate both buffer overflow attacks and format string attacks.

Question 4 – [20 Marks]

1) The ARM TrustZone allows secure execution with a “Secure World” and execution of regular/unsecure cose with the “Normal World”. Fully explain how the ARM Trustzone provides this securely isolated “Secure World”. Be sure to discuss it from both a software perspective and a hardware perspective, indicating what functionality from both sides is required to provide the “Secure World”.

2) Illustrate in detail how a secure booting process can be implemented through the use of a TPM.

Question 5 – [20 Marks]

1) Explain the purpose of both *Encryption* and *Hashing* when two parties wish to engage in a secure communication. Your explanation should make it clear why using only one of them is ineffective.

2) Explain clearly two reasons why public key cryptography can be necessary instead of symmetric cryptography, despite symmetric encryption keys being smaller and faster to use compared to public keys with the same level of security.

3) Alice wishes to communicate with Bob using symmetric key X and the AES encryption cipher. Currently Alice has the private RSA key Pr_a, and the public RSA key Pu_a. Bob has the private RSA key Pr_b, and the public RSA key Pu_b. Explain clearly all messages they need to send to each other in order for Alice to send the plaintext message M to Bob, including a digital signature with it.

Academic Integrity

This is an individual assignment. The work you submit must be your own. In no way, shape or form should you submit work as if it were your own when some or all of it is not. Any online source that is used must be cited. If you are unsure on whether something should be cited, general rule of thumb is to err on the side of caution and include the citation. You can also ask me via email. You must not directly copy text even from a cited source. All answers must be in your own words.

Collusion: Given how much freedom there is in the assignment, everybody's work will be different. It will be obvious if there is collusion. All parties to collusion will be penalized.

Deliberate plagiarism: You must not plagiarise the programs, results, writings or other efforts of another student or any other third-party. Plagiarism will meet with severe penalties, which can include exclusion from the Institute.

Your report will be checked for signs of collusion, plagiarism, falsification and fabrication. You may be called to discuss your submission and implementation with me and this will inform the grading, any penalties and any disciplinary actions.