COMP2700 Assignment 2 - Assessment Guidelines

Version 2021-10-20

This assignment has 6 challenges. For each challenge, you are required to submit two components:

- The artefact component (10%): The artefact for each challenge is the flag associated with the challenge: The flags must be submitted through the 'Assignment 2 -- Artefact' quiz link on Wattle. No other forms of submissions are allowed. This will be marked automatically on Wattle. The artefact component accounts for 10% of the total mark for this assignment.

- The report component (90%): For each challenge, except for the first challenge ('Substitution cipher'), you are required to explain in detail your exploration of the challenge, the analysis of its vulnerabilities, and the exploitation steps to obtain the flag that are reproducible. For the first challenge ('Substitution cipher'), no report component is required. That means the marks for the first challenge ('Substitution cipher') is completely determined by the artefact component. The report component must be submitted through Turnitin using the 'Assignment 2 -- Report' link below. The report component accounts for 90% of the total mark for this assignment.

Artefact assessment

The artefact (flag) for each challenge will be marked automatically on Wattle, by an exact match. The flag has the form flag{<some-text>} where <some-text> consists of two or more English words separated by -, for example, flag{hello-world}. Note that you need to include the tag flag{} in your answer, so if the flag you obtained is flag{hello-world} then enter exactly flag{hello-world} in the provided answer box.

Important notes on challenge #3 (CTR MAC): The provided ctrmac_oracle executable contains a bug that may be triggered under specific inputs, that would cause the oracle to print the flag (even when the input is wrong), or to refuse to print the flag when the input is correct. In most cases, we expect this bug to be not triggered. However due to this possibility of erroneous oracle output, if you did not manage to obtain the flag through the oracle, we will use your report to assess the correctness of your solution, and you will be given full mark for the artefact component for this challenge if your solution method is correct. To help you verify your solution without using the oracle, we have included instructions in the appendix of this guideline; the same instructions have also been announced on COMP2700 Teams (both in the Announcement channel and in the FAQ of Assignment 2).

The auto-marking on Wattle will be reviewed manually in case there is a mismatch. Minor typos, such as using '_' instead of '-', or additional spaces, are permissible and will be marked correct manually.

If you have submitted an artefact for a challenge that is not assigned to you, **and** is assigned to another student, even if you get the correct flag for that problem, you will receive 0 mark for the artefact component. An interview will then be arranged for you to explain why you worked on the wrong assignment problem, and you will be asked to demonstrate your exploitation method live (e.g., via Zoom or Teams). Depending on the outcome of the interview, a follow up investigation into academic misconduct may be initiated.

A flag submission for a challenge (except for challenge #1 – Substitution Cipher, where the report component is not required) that is not followed by a report component explaining how the challenge was solved will be given 0 mark.

Report assessment

The report component is required for each challenge, except for challenge #1 (Substitution cipher).

In general, your report should contain two main components for each challenge:

- An explanation of the vulnerabilities you discovered and how you discovered them. This is where you document your analysis of the problem, e.g., analysis of the design of the relevant cryptographic functions and/or source code, analysis of the provided plaintext/ciphertext, and/or the tests you've done to confirm your hypotheses about the vulnerabilities you found.
- Your attack strategy. Having discovered the vulnerabilities, how do you plan to obtain the flag? Describe your overall strategy, and the reasoning behind it, e.g., why do you think that particular strategy would work? Then map it to the concrete steps you need to do and how they translate into your exploitation steps. You need to explain all these components in some detail, so that the assessor can reproduce your exploitation if needed. If you use computer programs to automate parts of your attack, please list the code in the report. However, code alone is not a substitute for clear explanation. Code dumping, without any explanations in English accompanying the code, will not get you any marks. If the assessor deems your exploitation steps to be non-reproducible, an interview may be arranged for you to demonstrate your solutions.

Your report should be written clearly – pay attention to readability of your report, spelling and grammar, clarity of texts (e.g., if you post screenshots, make sure the important details are clearly readable), etc. A badly typeset report may attract a deduction of up to 5% of the possible marks for the report.

For each challenge, there can be one or more key components that lead to the exploitation. Your report will be assessed against the completeness of your analysis with respect to these key components. If a challenge can be solved in more than one way, you only need to explain the key components for your chosen solution.

Restrictions on the exploitation methods: The challenges for this assignment must be solved using analytical means, without using brute force search on the key space (in case the challenge is related to encryption/decryption or computation of message authentication code with a secret key), or in the case of hash function, without using brute-force search on the output of a hash function to find a second pre-image. Your solutions must be reproducible, so make sure you include key steps required to reproduce the flag. If a challenge comes with an `oracle` that prints the flag (when a correct input is provided), you are not allowed to exploit the oracle program itself to obtain the flag; this is an assignment on cryptography, not software security. Any flag obtained through exploiting the oracle program will be considered invalid and will get 0 mark.

Special notes on Challenge #3 (CTR-MAC): As mentioned above, the oracle (ctrmac_oracle) for this question contains a bug that may produce the flag even if your solution method is wrong. As it is logistically difficult to patch this oracle individually (as each student is assigned a unique instance of the oracle), to ensure your solution is the correct one, please make sure you perform a manual check using the instructions provided in the appendix of this guideline. You will be given full mark for the artefact component for this challenge if your solution is correct, regardless of whether or not you managed to get the oracle to print the flag.

Length of the report. Your report (containing explanations for all the challenges) should not exceed 3000 words, excluding figures, tables, code and output generated by programs. This is not a hard limit but keep in mind that reports that are exceedingly long will delay the release of your assignment marks.

To give you some ideas of the level of details we expect to see in the report, an example problem and the report containing a solution for that problem, is provided on Wattle (see the file 'ass2_example.zip' on Assignment 2 page on Wattle).

Late submissions
No late submissions are allowed without a prior approval from the convener of the course.

Interviews

As a deterrence to academic misconduct, the convener may interview a select group of students after the assignment due date. If you are selected for interview, you will be notified at least 5 business days in advanced.

The following are possible grounds for a student to be called for an interview:

- The student submitted an artefact (flag) for a challenge that is not assigned to the student.
- The student's report contains steps that the assessor deems non-reproducible.

- The student uses a method that the assessor suspects to fall outside the allowed exploitation methods.

During the interview, the student will be asked to give a live demonstration of the exploitation steps for selected challenges from Assignment 2 as determined by the interviewer. There are two possible outcomes of the interview: satisfactory or unsatisfactory. A satisfactory outcome will not lead to mark deduction. An unsatisfactory outcome may lead to mark deduction. An unsatisfactory outcome may also lead to a follow-up investigation into academic misconduct. For example, if the student demonstrates a complete lack of knowledge of their written solutions in their report, an investigation into collusion or contract cheating will be initiated. The use of an unapproved method, on the other hand, will not by itself lead to academic misconduct investigation, but will lead to mark deduction.

Note that the interviews are a separate process from, though they may lead to, academic misconduct investigations. In particular, plagiarism and collusion cases will be handled through the standard procedures for academic misconduct investigations. For example, if the convener suspects that a student plagiarised another student's report, a plagiarism case will be initiated, and the student will be informed that they are under an investigation.

Appendix – Checking the validity of your CTR MAC solution

The ctrmac_oracle provided as part of the CTR MAC challenge contains a bug that may produce wrong results, that is, it may print the flag even if your input was wrong, or in some rare cases, it may refuse to print the flag even when you provide it with a correct input.

To ensure that your solution to this challenge is indeed the correct one, you can use the following steps to verify it:

- Generate a MAC for the file fst.bin using a dummy key, e.g., 00112233445566778899aabbccddeeff

  ```
  ./ctrmac.py 00112233445566778899aabbccddeeff fst.bin > mymac.txt
  ```

- Construct your forged file snd.bin and MAC, using fst.bin and mymac.txt if needed (don't use the provided mac1.txt).

- Verify that your forged file snd.bin is different from fst.bin:

  ```
  diff fst.bin snd.bin
  ```

  (if they are different, you should see some lines printed. If the command does not output anything, it means the two files are identical, which means this isn't a valid solution.)

- Verify that the output of the following command matches your forged MAC:

  ```
  ./ctrmac.py 00112233445566778899aabbccddeeff snd.bin
  ```

  If your solution is correct, this should output a string that is identical to your forged MAC.