Protocol Security Lab — 02239 Data Security

Hound-out: September 25, 2024 — Hand-in: October 25, 2024

Group hand-in: form groups of 3 to 4 participants on DTU Learn (self-enrollment)

The lab exercises use the protocol analyzer OFMC 2024 that is found on DTU Learn. The distribution includes executables for Windows, Mac and Linux.[1]

You can test that you have OFMC correctly installed by running it for instance on the example `nspk.AnB` from the lecture found in `examples/cj/6.7...`.

Both of the following two exercises are real-world protocols (with minor modifications/simplifications), both protocols had received a thorough security analysis, and both had attacks that the analysts missed. Have fun finding and fixing them!

For each of the tasks, you find an AnB file on DTU Learn. The first step is to see what attack OFMC gives on them:

`ofmc` *TaskX.AnB*

You need to understand the attack and find a way to fix it. Make a copy of the given file before modifying it. When you want to check that your fix actually works, you can check it with the following command line:

`ofmc` `--numSess 2` `--ctf` *TaskX.AnB* *YourFixX.AnB*

(`--numSess 2`: bounding sessions; `--ctf` *TaskX.AnB*: CTF mode)

Here the first option to bound to two sessions means that OFMC stops if it does not find an attack with any two instances of the protocol roles in parallel. (Note that on a general protocol, OFMC would not terminate but try out an increasing number of sessions until an attack is found or you terminate OFMC with Control-C. With this option it is guaranteed to stop.)

The second option, CTF mode, is given the original (attackable) protocol as an additional input file (besides your fix). In CTF mode, OFMC checks that you did not change the initial knowledge of the roles or the goals (or used the channel notation) – only then it is a valid solution. (By adding additional knowledge to participants or dropping goals or using channels, you could cheat and make a trivially correct solution – that does not give points or at least not full points.)

Task 1 For the first protocol you should discuss the following issues in your report:

1. Try to understand what purpose this protocol tries to achieve. To that end, first look at the initial knowledge and the goals. Can you give a typical real-world scenario from the Internet where the participants have this initial knowledge and want to achieve this goal?

[1] For compiling the sources yourself, you need the Glasgow Haskell Compiler.

2. Describe the attack found by OFMC: what does the attacker do and what goal is broken here?

3. How could this attack be prevented? To answer that it can help to imagine what the protocol designers had in mind with the different messages of the protocol and where the actual attack deviates from their idea: possibly there is just a small modification to fix the problem.

4. If you add the following goal:

   ```
   sk(A,s) guessable secret between A,s
   ```

   then OFMC finds another attack against the original protocol, and probably also against your fix (depdening on how your fix works). Explain why a guessable `sk(A,s)` can be a problem and try to find a fix for that as well.

Task 2

1. Again, start with a description of the purpose of the protocol: what does it try to achieve, how does it work?

2. Analyze the protocol with OFMC and explain the attack: what does the intruder do, what went wrong?

3. How can one fix the protocol by only changing messages in the Action part? Hint: only make changes to the message from `hm` to `B`.

4. Note that the party `hm` is a fixed *honest* (*trustworthy*) server. Let us replace `hm` by `Hm`, i.e., a normal role that can be instantiated by the intruder. Why does the protocol have an attack then? Can there be *any* protocol with the same initial knowledge and the same goals that is secure?