IT Security 2024/2025
Exercise Sheet 10
– Version Detection –

Ben Swierzy

Publication: 12.12.2024
Deadline: 18.12.2024 10:00

Lightning Survey ⚡

Exercise 1 (Flavius, 6×0.5 points)**.** You have been using Flavius for more than two months. Time to take a closer look how it works. Open the web page in your browser and answer the following questions: Write your solution into `task1.txt`.

(1) What is the version of the web server serving the frontend?

(2) What JavaScript framework is used for the frontend?

(3) What is the version of the core component of that framework?

(4) What server serves the API backend?

(5) In what programming language is the API backend written?

(6) What framework is used for the API backend?

Note: In most browsers you can use F12 to open the developer tools and analyze network traffic in the NETWORK *tab.*

Exercise 2 (Builtwith, 4×0.5 points)**.** Check out the scan report from BUILTWITH.COM for STUDIERENDENWERK-BONN.DE. From each of the following categories, select one detected technology and briefly describe a robust method how this information can be obtained in 1-2 sentences. Write your solution into `task2.txt`.

- Widgets
- Frameworks
- JavaScript Libraries and Functions
- SSL Certificates

Exercise 3 (Automata Learning, 5 points)**.** In this task you will practically explore the state machine of `tftpd`, an implementation of the very simple TFTP-protocol (see RFC1350). You find basic wrappers around the packet format in `tftp.py`. Use the Python library `aalpy` to learn the automaton with the L^* algorithm and the Random Walk equivalence oracle. You can use the official MQTT example[1] for inspiration. Your input alphabet should not only consist of the five regular package types but also contain another packet not matching the specification.

Submit your script as `task3.py` and the automaton in Graphviz Dot format as `task3.dot` to the repository[2]. Just calling `print` on the output of `run_Lstar` will use this format. If

[1] `https://github.com/DES-Lab/AALpy/blob/88c9450e4d41ebf3b18cfad551d08096c54bc13f/Examples.py#L293`
[2] If you want to check your automaton visually, you may use the following web tool: `https://dreampuf.github.io/GraphvizOnline`

your automaton has less than three states, you probably have an issue in your implementation. Wireshark might help you in this case.

Hints:

- You can use the following docker image for the implementation:
  `sudo docker run --net=host --rm -it pghalliday/tftp -L -p -c`
- TFTP works via UDP port 69 where the tftpd will listen on your host. After receiving an expected packet, it answers from a different port which is then used for the remaining connection.
- Python's `sendto` and `recvfrom` methods of a `socket.socket` instance are very helpful to obtain and set the ports of incoming and outgoing packets.
- When receiving, it is recommended to use a timeout of 0.05s to obtain a decent performance without packet losses.
- If you experience determinism errors related to WRQ, try to never send two WRQs with the same file name.