Hacking

Willard Rafnsson

IT University of Copenhagen

In this assignment, we will put ourselves in the shoes of the attacker, and hack systems. The goal is to familiarize ourselves with the activities and capabilities of an attacker, so that we know what to expect when engineering security into software. Concretely, we will be crafting *code injection* attacks to solve *capture the flag* challenges.

Capture The Flag

This is hacking as a game. Your objective is to find and report (i.e. capture) the flag. You do this by using any hacking skill and tool that you have at your disposal.

Haaukins Navigate to the Haaukins URL, given in this assignment's section on the course website. Create an account on Haaukins with your ITU initials as Team name[1]. This creates a Kali Linux virtual machine for you on Haaukins, along with a network containing vulnerable computers for you to hack in this assignment. You access and use your virtual machine on Haaukins through the browser. If you need help navigating the interface, then you can check out `https://www.cybertraining.dk/`.

Flag Flags can be placed in various locations – they might be in a file, in the database, stuck into source code, or otherwise – and your goal is to hunt them all down. Each flag looks like `HKN{**********}` or `DDC{**********}`, and often has the word "flag" in some form near it (e.g. `flag`, `Flag`, `FLAG`, `fl4g`, etc.).

Hand-In Once you find a flag, submit the flag in Haaukins under CHALLENGES. The challenge must turn green for it to be considered passed (the TAs will check). (Ignore the "FTP Server Login" challenge for now; that's for later in the course). Then write up a *report* on what you did. List the steps required to obtain the flag. Screenshots & code snippets are welcome. Submit these reports as a PDF in LearnIT.

Conduct Do post if you are stuck, and feel free to give hints in the right direction. But, don't share the solution!

[1] While you do not *need* to use your initials, using your initials will help us identify you in the system, which will help in case you need assistance.

Copy/Paste To/From VM Click INFO on the Haaukins page; there you will find an instructional video on how to do this.

Warning When injecting code into a server, if you are not careful, you might inadvertently make the server unavailable (i.e. compromise *availability*), by e.g. wiping data from a database, crashing software on the machine, or trigger an infinite loop. You can recover from this by logging in on Haaukins, and, instead of connecting to your VM, head to challenges, select the relevant challenge, and hit Reset. Likewise, if you similarly damage your VM, then you can RESET Kali Machine from the same place. That being said, there have been challenges with recovering from these situations in the past. So, avoid "damaging" the services that you are hacking, and do not power off your VM. If you get in a situation that you cannot recover from, please notify course staff; they have admin access, and can fix this. If you cannot wait for them to do so, then you can create a new VM (with e.g. a "0" at the end of your user name). However, only do so if your situation is dire; the server has only a fixed number of VMs it can create (40 for the whole course). Haaukins is a prototype; despite this, we find that it is the best fit for our course. We hope you agree, and we hope for your understanding and patience in case of any flaws you stumble into (e-mail any flaws to me!). Have fun!

Problem 1 : SQL Injection

Visit `http://hackerroom.com/`, an *internal forum*. You are presented with a login interface to a messaging board. Log in as the administrator, and write down the flag.

Hint: You will need to do some sleuthing to find out who the administrator is.

Hint: There is some (simple) client-side input validation in place; read error messages from the login interface carefully to ensure that your login request reaches the server.

Problem 2 : Session Hijacking

Visit `http://todos.hkn/`, a *"todo-list"* site. One of the users of this site has some interesting todo-items. Access the site as that user.

Hint: Create a user account on the site, login, and inspect the session cookie stored in your browser. Does the format look familiar? See if you can decode it.

Hint: The `base64` shell command might be useful; check the `-d` and `-w` flags in its `man`ual-page.

Problem 3 : Insecure Deserialization

There is one more Web server on the network listening on port `80`. Find it (by scanning the network using `nmap`), and visit it in a browser. Access the page as Admin to reach the goal; find the flag, and write up your solution (~ 4 sentences).

Hint: The `--open` flag of `nmap` filters the output to only list those machines that have your specified port open.

Hint: We recommend using the developer tools in your browser to find the serialized object. It is encoded; can you see how? There is a command-line tool by the same name to decode and encode data from/to that encoding. The `%3D` at the end should actually be replaced with `=`. If you want to play with a new tool, then use Burpsuite instead.

Hint: The server runs PHP; lookup how PHP does object serialization/deserialization.

Problem 4 : Reflection

Let's consider the ramifications of what we have learned.

Part 1 In the three CTF challenges above, which *asset(s)* are vulnerable to what kind of *harm* (i.e. which *aspect of security* (CIA) is violated)? (~ 3 sentences)

Part 2 Suppose that, instead of receiving an output that depends on your input, you receive something generic, e.g. "Your e-mail address has been removed from the mailing list", or "Thank you; your order has been placed". How do you check (using the input fields) whether such a Web application is vulnerable to an injection attack? What kind of *harm* could you do, and how? (~ 2 sentences).

Part 3 What can you do, as a security engineer, to prevent injection attacks when designing a system? (~ 1 sentences).

Hint: What must you always do to input from untrusted sources (user input)?