# Rigel (Launched 2003)

- *Username:* `rigel`
- ▶ *Click to reveal password:*
- *Points:* 10 for code, 5 for writeup

*Relevant lectures: 5 - Mitigating Memory Safety Vulnerabilities*

> **STORY**
> The revelations from Antares are clear. Gobians considered the orbiter a serious threat, and you must too. Luckily, you now know where the final answer to this question, the blueprint, lies... Rigel is a true display of Gobian technological ingenuity. Launched right before the fall of the Union, it is armed with all of the most powerful hardening techniques at the time. Luckily, CSA allies have managed to disable the non-executable pages on the remote system and provided you with the shellcode to extract the blueprints from the satellite. Your final job is to defeat the remaining ASLR and stack canary countermeasures, hack into Rigel, and get the blueprints to fully understand Caltopia's true intentions.

This part of the project enables both stack canaries and ASLR.

Consider that enabling ASLR means you may end up with a nondeterministic solution. `./exploit` accounts for this by running multiple times (which could take some time). If you see some `Segmentation Fault`s when running your script, that's expected!

---

## Tips

- It might help to read Section 8 of "ASLR Smack & Laugh Reference" by Tilo Müller.
- You may find it useful to know how to examine the addresses of individual assembly instructions. This can be done by running `disas <function_name>` in gdb, where `<function_name>` is the name of a function like `main` or `abs`

- It might also help to note that a no-op instruction in assembly can be represented by the single-byte instruction `0x90`

---

## Deliverables

- A script `interact`
- A writeup.

---

---