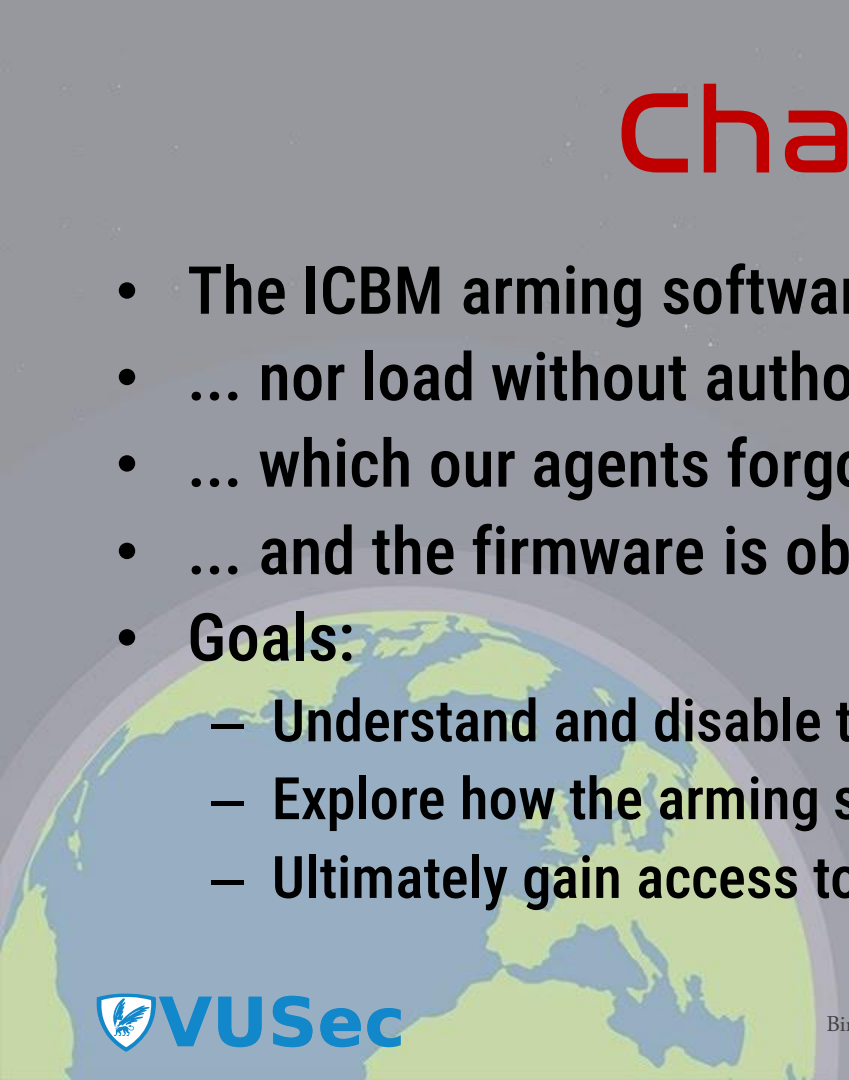
Challenge 1

Challenge 1

- The ICBM arming software won't even load on our PCs!
- ... nor load without authorization
- ... which our agents forgot to procure
- ... and the firmware is obfuscated and resists analysis
- Goals:
  - Understand and disable the anti-* techniques
  - Explore how the arming software checks for authorization
  - Ultimately gain access to the launch code calculator... somehow



Challenge 1

- Anti-debugging
  - So that the binary doesn't object to be run in gdb
- Anti-VM / Platform detection
  - So that the binary doesn't object to be run in a virtual machine or, generally, outside its intended hardware

Challenge 1

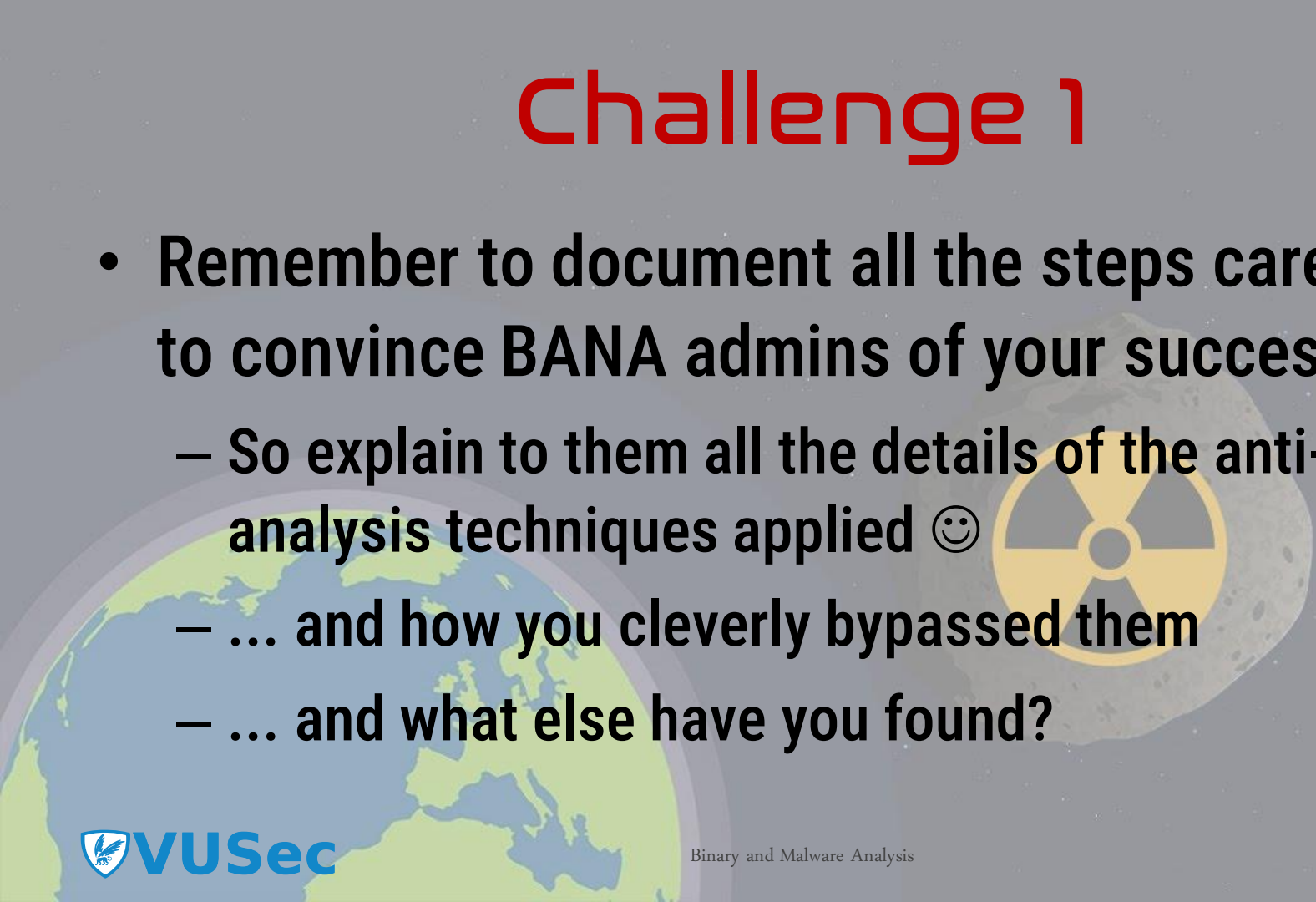
- Get auth & load the launch code calculator
  - The software apparently needs authorization
  - We can reverse-engineer that!
  - Our agents said there might exist a “master auth” that works on all nukes, worth looking into
  - Although such power is surely extra guarded...

Challenge 1

- Misc reports from our agents:
  - Devs sometimes leave debug messages for themselves which are hidden on production HW
  - Finding these might give us some insight

Challenge 1

- Remember to document all the steps carefully to convince BANA admins of your success!
  - So explain to them all the details of the anti-analysis techniques applied ☺
  - … and how you cleverly bypassed them
  - … and what else have you found?



Challenge 1

- Understand and disable the anti-* techniques → max. 6 points
  - Anti-debugging → max. 1 points
    - So that the binary doesn't object to be run in gdb or strace
  - Anti-VM / Platform detection → max. 5 points (2p basic check + 3p supplementary check)
    - So that the binary doesn't object to be run in a VM, e.g., Qemu, Vmware or outside its intended hardware
    - Hint: not all checks run on all authorization attempts.
- Investigate the authorization system → max. 2 points
  - Explain how the binary checks for authorization → max. 1 points
  - Investigate the possibility of a "master authorization" → max. 1 points
    - Figure out what this authorization is and explain how the binary behaves (no patching)
- Successfully start the launch code calculator → max. 1 points
  - Do NOT patch this part out, instead figure out what the binary expects
- Debug code left by developers → max. 1 points
  - Describe under which environmental conditions the binary prints debug messages and what these are. (don't patch this part)

Challenge 1

- Bonus for the first five students who correctly solve the challenge ☺
  - + 50, 40, 30, 20, 10 pts
  - **Doesn't** count towards grading
  - **Does** count towards scoreboard (details to follow)

Challenge 1

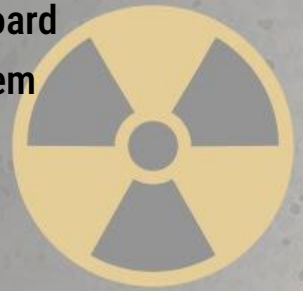
Hint:

- You are allowed and encouraged to run the binary, so you can use tools like gdb and strace in your investigation.
- This assignment is not about learning how to sandbox your environment, like you would normally do when analyzing malware.
- The requirements are not strictly linear! If you're stuck try working on a different item.

Submission Guidelines

You need to deliver a **zip file** containing:

- Patched binary with antianalysis techniques disabled, named ‘armatronic.antianalysis’.
- A text file ‘README’ containing:
  - A nickname of your choice to be displayed in the scoreboard
  - What anti-* tricks you have found & how you disabled them
  - How the binary checks for authorization
  - Your findings regarding the “master authorization”
  - Environmental conditions that trigger debug output
  - How to successfully start the arming code calculator
  - A write-up on how you produced the patched binary



Submission Guidelines

- Submission mechanism is **Canvas**.
- The binary will be sent to you by e-mail shortly.
- **Deadline: Tuesday, 14th April 2020, 23:59 CEST**
- Delay penalties: 1pt/24h delayed

GOOD LUCK!

And don't forget about the Canvas forum!
Experienced 1337h4xx0rs may drop a hint!