Challenge Documentation Standard v1

Hi/Welcome/Salutations,

You will need to complete this documentation for any and every challenge you develop. Deliver it along with your game files.

# Deliverable

For file download challenges:

* The file to be provided to players
* This document

For challenges that run on infrastructure:

* Source code
* Dockerfile
* Docker-compose file
* This document
  + If applicable: PoC source code to beat the challenge
* You are **not** expected to return the sample challenges or sample docker-compose file that is in the repository

**Make sure to use relative pathing in your config files.** Your challenge is expected to be autonomous, an admin should be able to run docker-compose up and the game be operational, on the port expected, and without need for modification.

If you require clarification or have queries regarding challenge infrastructure, please get in touch with C\_Sto on the challenge development Slack.

# Challenge Metadata

|  |  |
| --- | --- |
| **Company Name** | Hivint |
| **Developer Name(s)** | Sam Reid |
| **Contact Email / Phone** | sam@hivint.com |

|  |  |
| --- | --- |
| **Challenge Category** | Web / Exploitation / Crypto / Network Forensics / Miscellaneous |
| **Challenge Tier** | 1 / 2 / 3 / 4 |
| **File Download?** | Yes / No (If yes, provide the file name) |

Challenge Details

All fields mandatory

# Player facing

|  |  |
| --- | --- |
| Challenge Name | Kawaii Challenge Response Protocol |
| Challenge Synopsis Details given to the player before attempting the challenge. Max 150 words. | Waifu Robotics have gone rouge, capturing and keeping waifus in their highly secured vault. Their ‘KCRP’ is a challenge/response protocol for authenticating with a pre-shared secret. It’s thought to be the best CR protocol known to otakus.  You’ve already had some success against the protocol though, obtaining the source code of the authentication server, but it doesn’t contain the secret key:  <LINK TO DOWNLOAD>  Looks like you’re going to need to break KCRP to unlock the vault and save your beloved Yukari. |
| Challenge Hint(s) Provide up to three (3) hints that can be periodically given to players struggling to complete the challenge. | 1. XXXXXXXXXXXX 2. XXXXXXXXXXXX 3. XXXXXXXXXXXX |

# Administrator facing

|  |  |
| --- | --- |
| Challenge Flag What is the flag that the player retrieves upon winning? | WACTF{YUKARI\_MY\_LOVE\_6270} |
| Challenge Vulnerability What is/are the vulnerability/ies exploited in this challenge. Link to articles/writeups where applicable. | Buffer overflow in parsing of the “response” variable allows player to modify the “secret \_key” variable to a known value. Player can then return the correct response when challenged to get the flag. |
| Challenge Proof-of-Concept Provide a step-by-step PoC for your challenge. Include screenshots if easier. If players are expected to build a script to beat the challenge, provide the winning source code **as well as** written PoC. | 1. Identify vulnerable line in source code as 139 2. Ask server for challenge:    1. Payload: “0” 3. Craft response to overflow “response” buffer and set “secret\_key” variable to known value:    1. Hex encode payload    2. Sample payload: “13030303030303030304141414141414141” 4. Observe failure to authenticate and then request a new challenge from the server:    1. Payload: “0” 5. Hex decode challenge and XOR output with the chosen secret key (as set in step 3) 6. Respond    1. Hex encode after XOR in step 5    2. Sample payload: “16C6A346B7537617871” 7. Observe successful authentication and the flag returned. |