|  |  |
| --- | --- |
| Logo  Description automatically generated | Challenge Writeup  Sci-Fi NameGen  by Kouretas Panagiotis |

# Main Concept

# Welcome to the Sci-Fi Character Name Generator! Choose your character role and generate your unique Sci-Fi character name.

# Exploitation

# Sci-Fi NameGen

First of all, we deploy the challenge from CTFLib.

Then we are redirected at <http://localhost:8081/> which is the home page of the challenge as seen in Figure 5.1. To generate our character name, we need to fill up our first and last name and to choose one of the given character roles.

A screenshot of a computer

Description automatically generated

Figure 5 1 - Challenge's Home Page / Sci-Fi name generator page.

Then if we click the “Generate Sci-Fi Name” button, we are moved to another page where we can see our generated Sci-Fi character’s name at <http://localhost:8081/generate.php?input=PD94bWwgdmVyc2lvbj0nMS4wJyBlbmNvZGluZz0nVVRGLTgnPz48aW5wdXQ%2BPGZpcnN0TmFtZT5QYW5hZ2lvdGlzPC9maXJzdE5hbWU%2BPGxhc3ROYW1lPktvdXJldGFzPC9sYXN0TmFtZT48cm9sZT5HYWxhY3RpYyBFeHBsb3Jlcjwvcm9sZT48L2lucHV0Pg%3D%3D> as demonstrated in Figure 5.2.

A screen shot of a computer

Description automatically generated

Figure 5 2 - Page for dIsplaying the generated name.

The first thing we notice is that the URL of the challenge contains an input parameter. This parameter is an encoded value and most probably base64 encoded. Let’s decode this value using Hackvertor which is one of the Burp Suite’s Proxy tool extensions. The decoded value is the following as depicted in the figure below:

<?xml version='1.0' encoding='UTF8'?><input><firstName>Panagiotis</firstName><lastName>Kouretas</lastName><role>Galactic Explorer</role></input>

A screenshot of a computer

Description automatically generated

Figure 5 3 - Base64 + URL decoding value using Burp's Hackvertor.

If we try to modify, decode and send back the updated XML data in the input we can see that it's done successfully as depicted in Figure 5.4. So, it seems we have control over the XML data.

A screen shot of a computer

Description automatically generated

Figure 5 4 - Successfully modifying XML data in the input parameter.

Since we have XML data, let’s try an XXE (XML external entity injection) attack. Upon further investigation, I found the following payload from “PayloadAllTheThings” which is an online GitHub repository.

<?xml version='1.0' encoding='UTF-8'?><!DOCTYPE input [<!ENTITY xxe SYSTEM 'file:///etc/passwd'>]><input><firstName>&xxe;</firstName><lastName>Kouretas</lastName></input>

This payload attempts to read the contents of the “/etc/passwd” file on a Unix-like system, which is a common target for attackers as it contains a list of users on the system. Let’s Base64+Url encode this and pass it in as our input in the URL.

A screen shot of a computer

Description automatically generated

Figure 5 5 - XXE payload successfully returning "/etc/passwd" file.

It looks like we have a Local File Inclusion. Although there doesn’t seem to be anything interesting in that file, so kept looking around the page searching for something that could help us take it one step further.

Inspecting the source code of the page, I found a comment that was indicating that there is a special functionality by accessing the website from a local address.

Comment: “<!-- Special sci-fi Character Name functionality: To test the special sci-fi title functionality, make sure you're accessing this page from the web server. -->”.

Based on this let’s send off the following updated payload that accesses the website from the localhost and see what will happen.

<?xml version='1.0' encoding='UTF-8'?><!DOCTYPE input [<!ENTITY xxe SYSTEM 'http://localhost/generate.php'>]><input><firstName>&xxe;</firstName><lastName>Kouretas</lastName></input>

A screen shot of a computer

Description automatically generated

Figure 5 6 - Hidden flag successfully retrieved.

YEAH, it seems that the special sci-fi title functionality is returning the hidden FLAG.

CTFLIB{XXE\_Galaxy\_Br3ach}