Text, logo

Description automatically generated with medium confidence

Web Juggling Facts

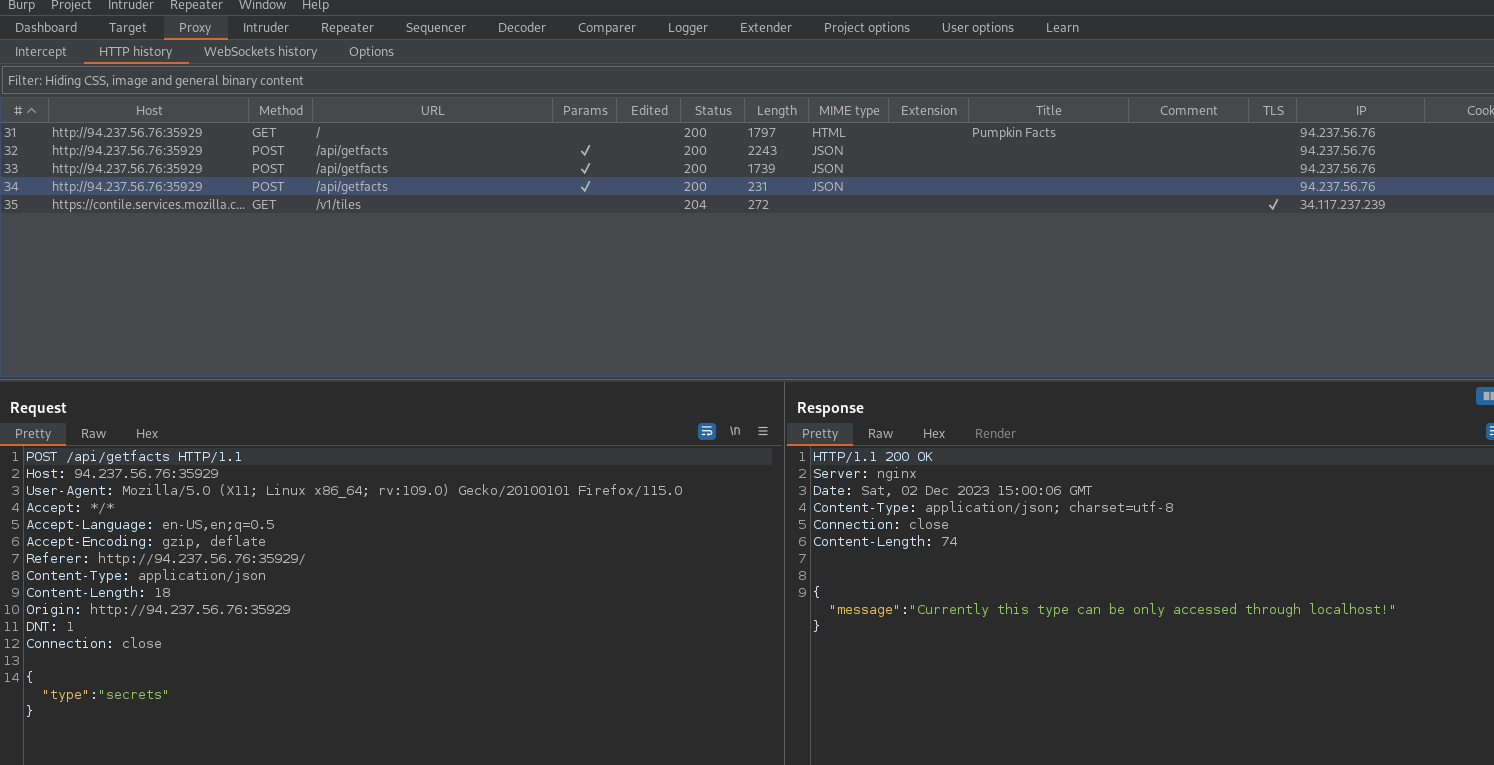
This is a web exploitation challenge where participants are tasked with recovering a secret from a web page. The challenge provides the source code of the web page, and participants need to analyze it to identify any vulnerabilities that can be exploited to reveal the hidden secret. The goal is to understand the code, find weaknesses, and exploit them to uncover the confidential information.

Lets visit the address of the challenge



We are presented with a random pumpkin facts generator

Once we tried its capabilities we review them in burpsuite:

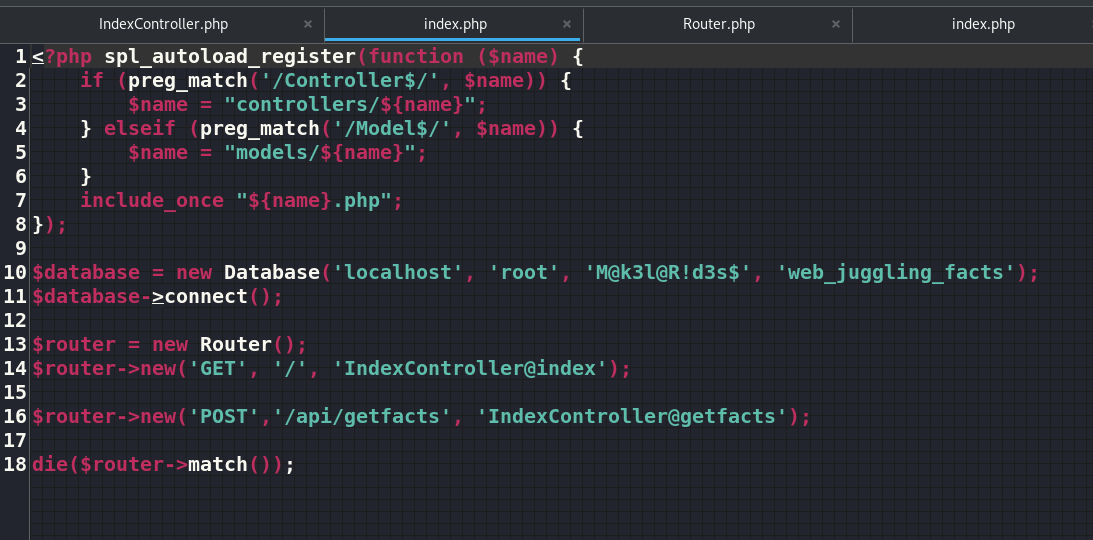


The thing that comes to our attention right away is the response message for the secrets type:

Currently this type can be only accessed through localhost!

lets do some code analysis so we can better understand why we receive such a response.

Index.php



First thing we can see is a cleartext password, and a rounting also a IndexController.php and some of its functions

Lets check them out too:



So the server seems to have a check enabled for ‘REMOTE\_ADDR’ to reply only to a localhost.

We can try some header injection like:

X-Forwarded-For: evil-website.com

X-Forwarded-Host: evil-website.com

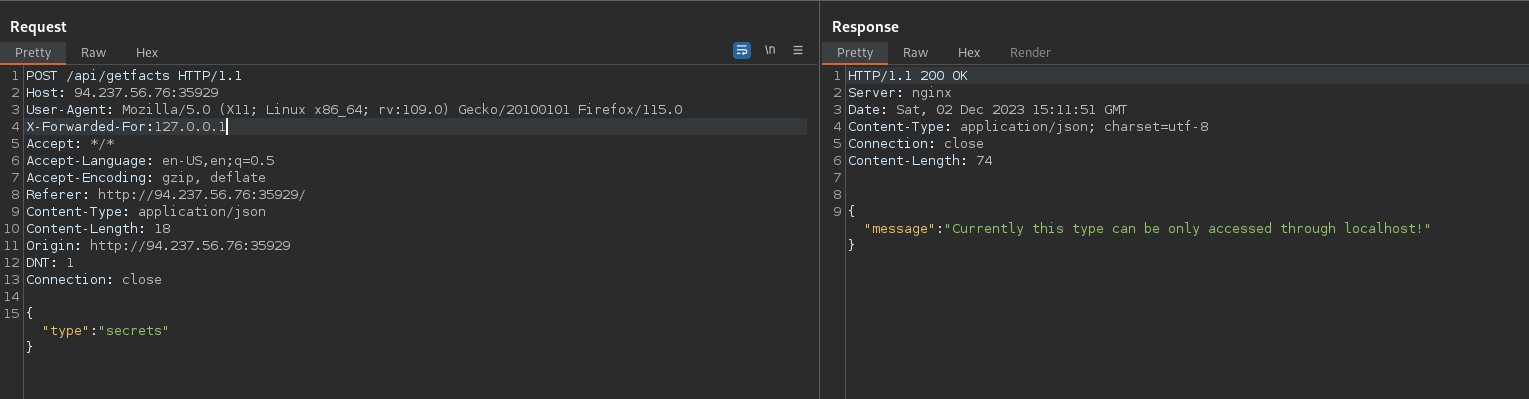
X-Client-IP: evil-website.com

X-Remote-IP: evil-website.com

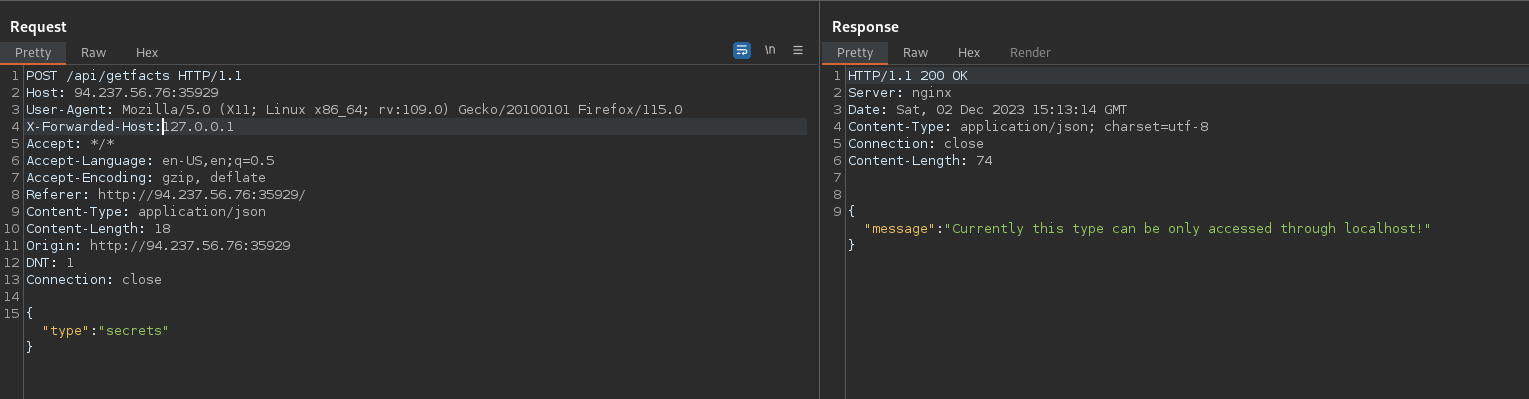
X-Remote-Addr: evil-website.com

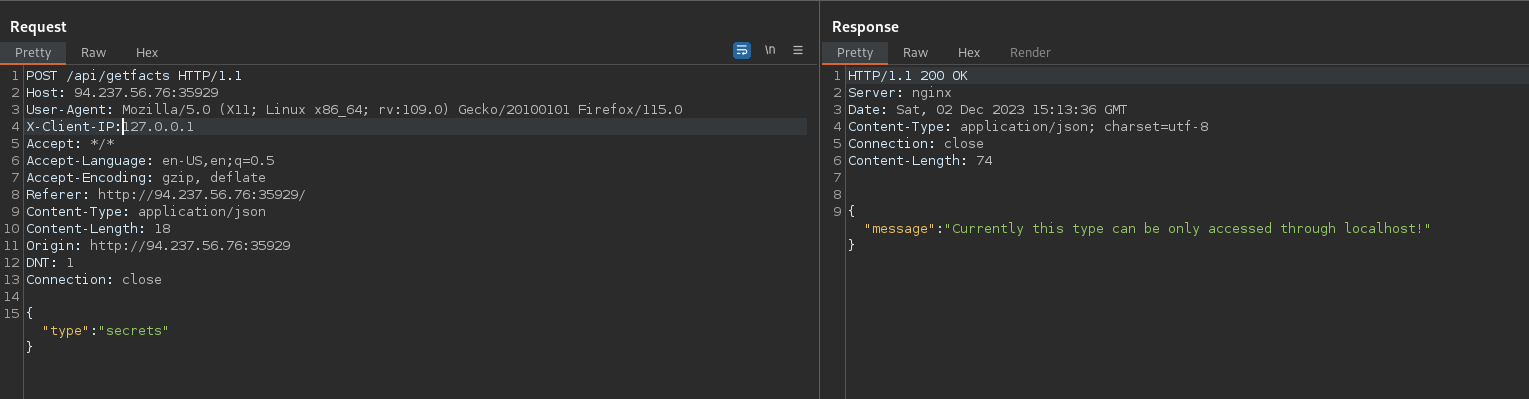
X-Host: evil-website.com

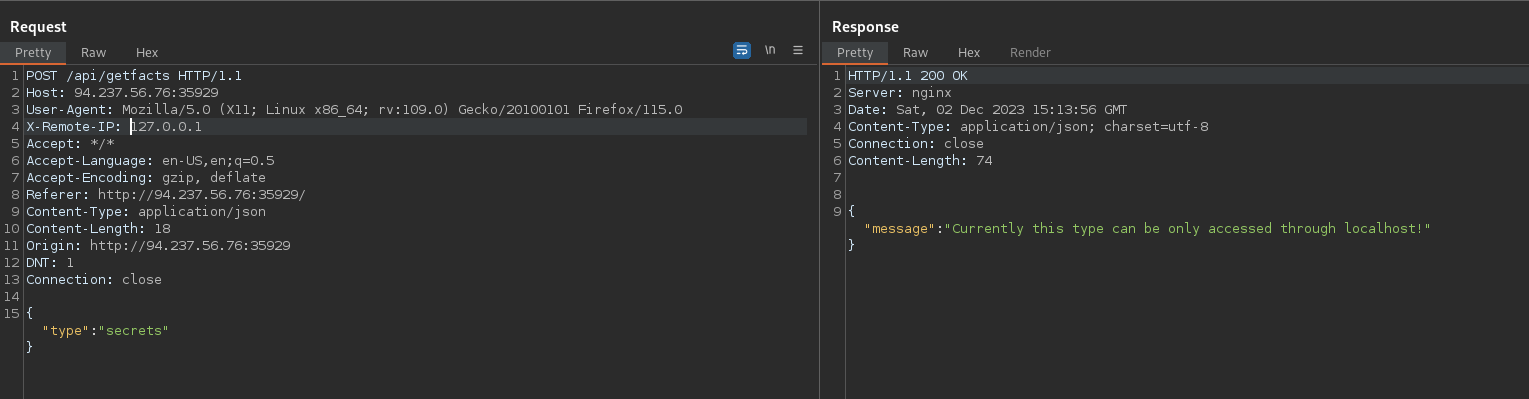
We send the post request for a secrets to a Burp Repeater with a X-Forwarded-For header:

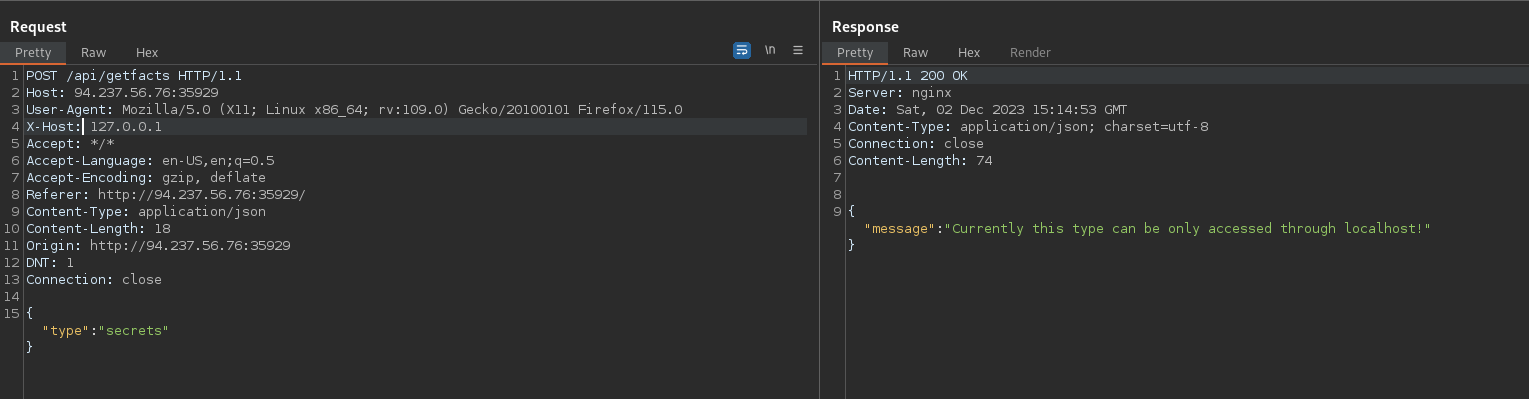
****

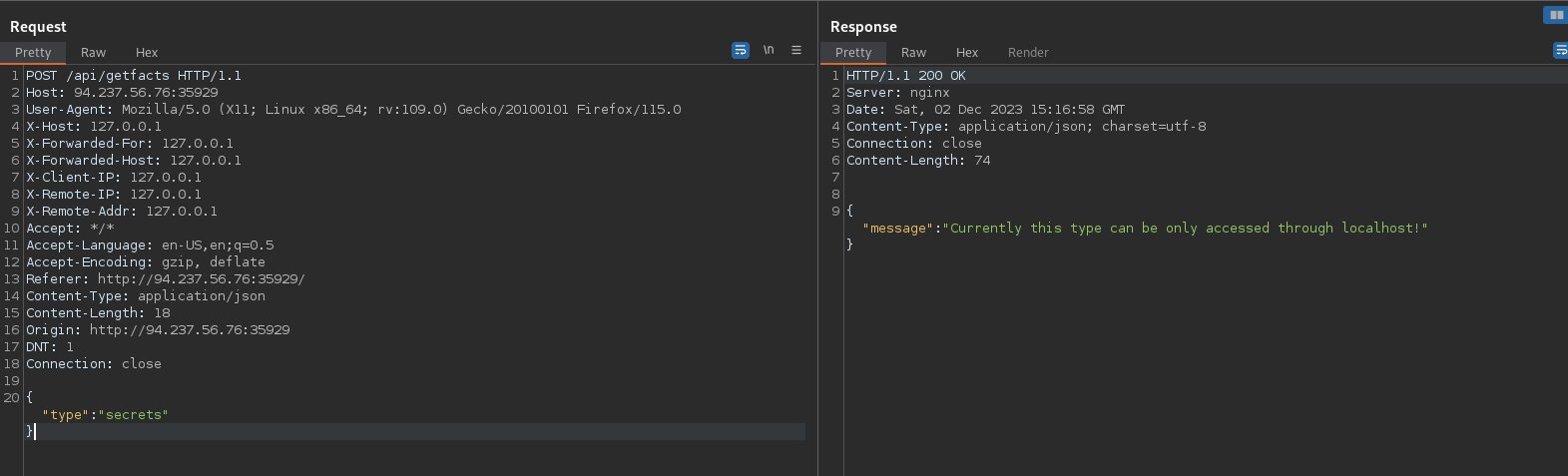
Dosent work, lets try a different one:











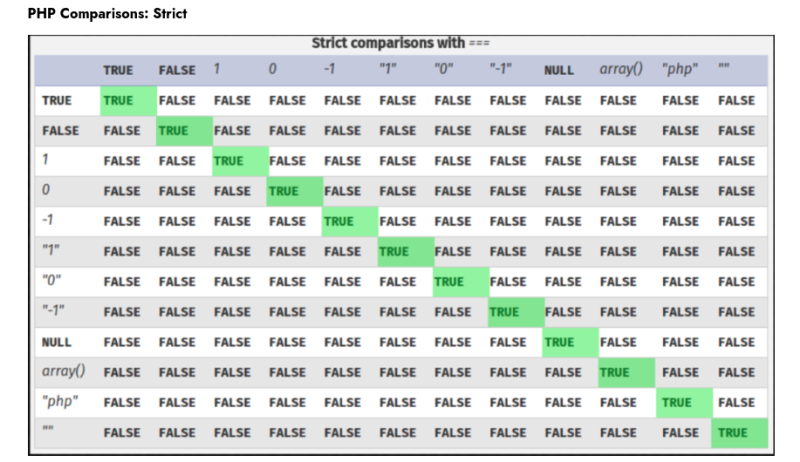
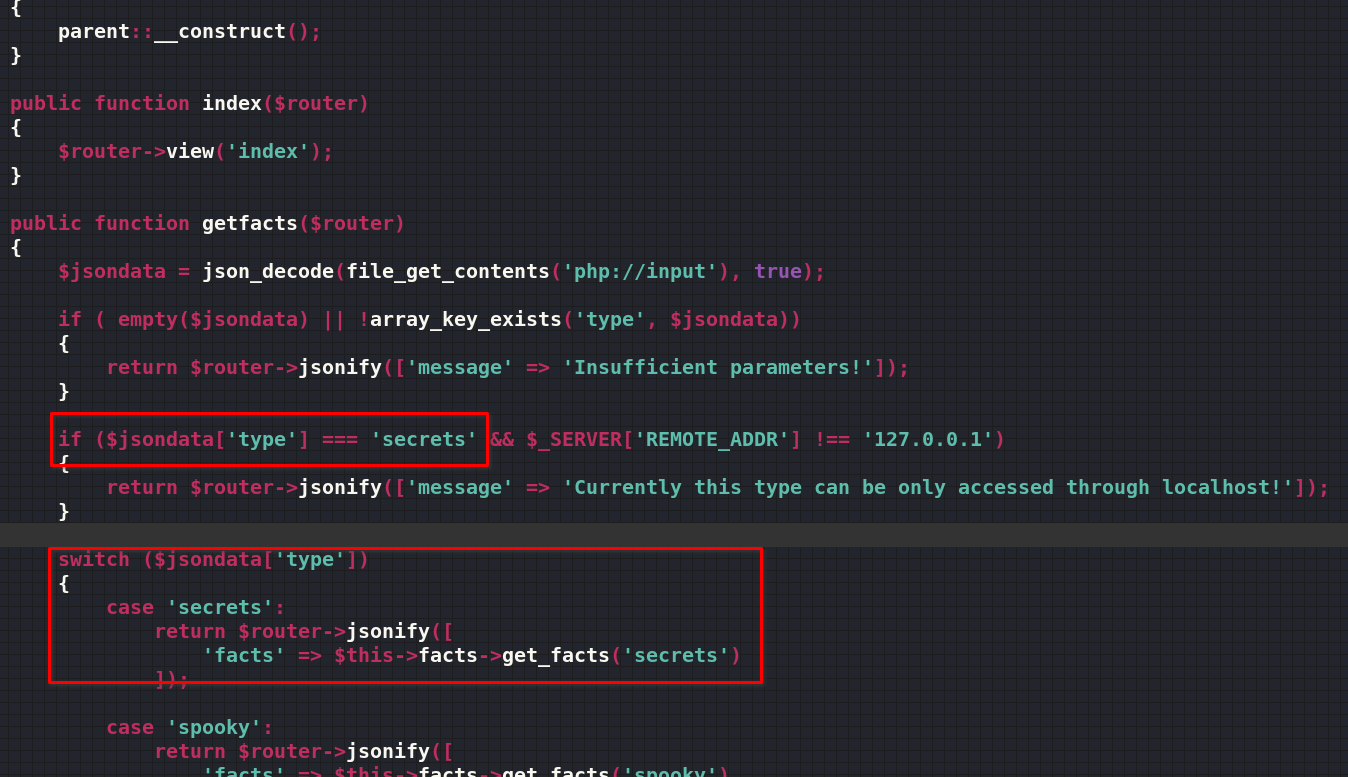
Diddn’t get trough, lets dive in again at the source code, since we are probably missing something

Lets review the source code again:

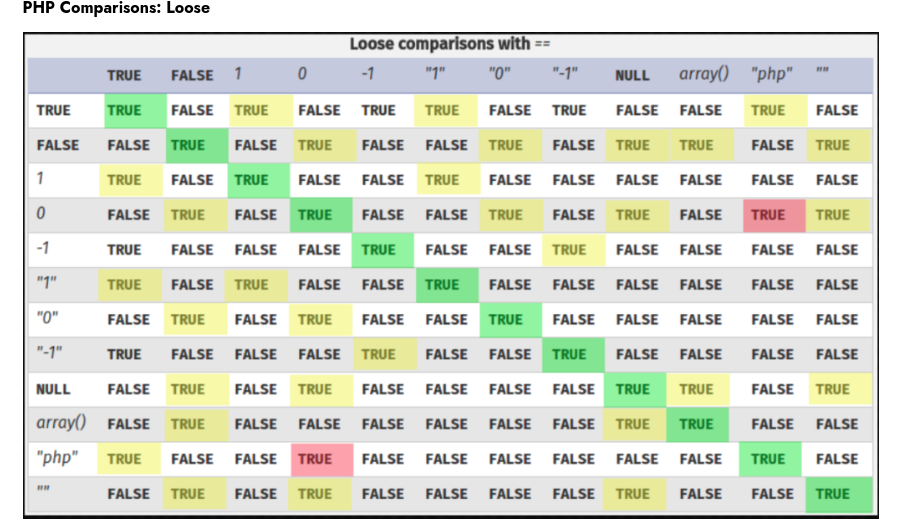
After some research on the Indexcontroller code I found that get\_facts uses stict comparison:

PHP type juggling vulnerability arises when loose comparisons (==) or strict comparisons (===) are used without careful consideration of the data types being compared. In the context of security, the === (strict comparison) operator is generally considered safer than == (loose comparison) because it not only compares values but also ensures that the data types are the same.

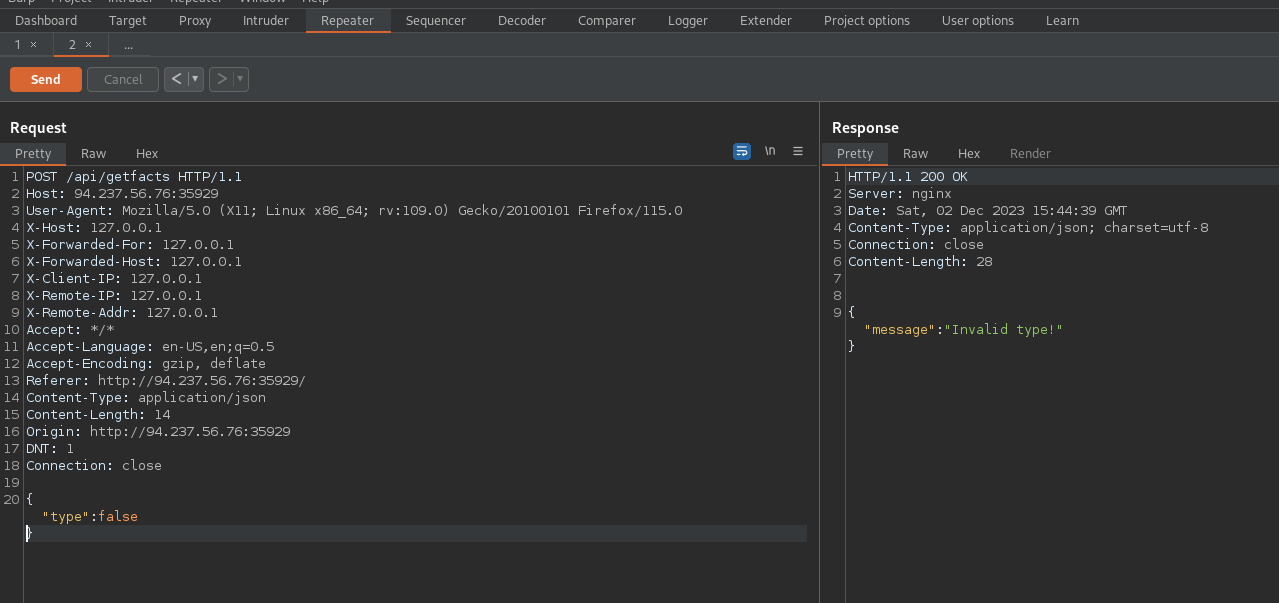
With === both the values and the data types must match for the comparison to be true. If the data types are different, the comparison will return false



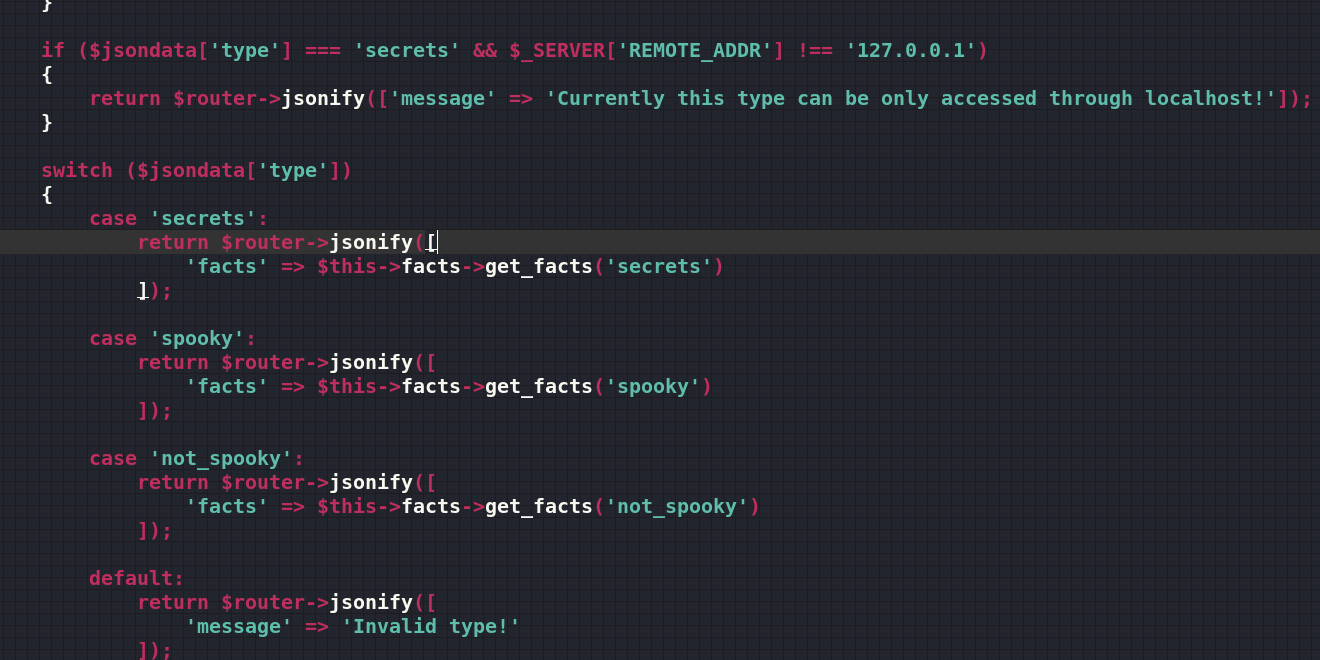
However the switch statement uses loose comparison this means that it performs type coercion during comparisons, similar to the loose equality operator (==). When using switch, PHP checks for equality using loose comparison for each case statement until it finds a match.



in our case we are comparing string to a value of the input

in that case if i type in false, it must invoke the error: Invalid type

So if we type in true, based on the table above the switch will hit the first match, which is: secrets



We got the flag:

