



20TH MARCH 2025


ASSIGNMENT

INT 310- CODE VULNERABILITY ANALYSIS/ AMINU IDRIS

SUBMITTED BY:

Ayilara Busari Dare

IDEAS/24/28133



INT310: Web Application Source Code Vulnerability Analysis – Lab 2: Comprehensive Vulnerability Review

Introduction to Code Vulnerability Analysis

Code vulnerability analysis is the process of reviewing source code to identify security flaws that attackers could exploit. It helps prevent cyber threats, ensures compliance with security standards, and strengthens software security.

Common Vulnerabilities in PHP & Python

- Injection Attacks (SQLi, XSS, Command Injection)
- Insecure File Handling (Unrestricted Uploads, Directory Traversal)
- Weak Authentication & Authorization
- Deserialization Issues (Remote Code Execution)

Analysis Methods

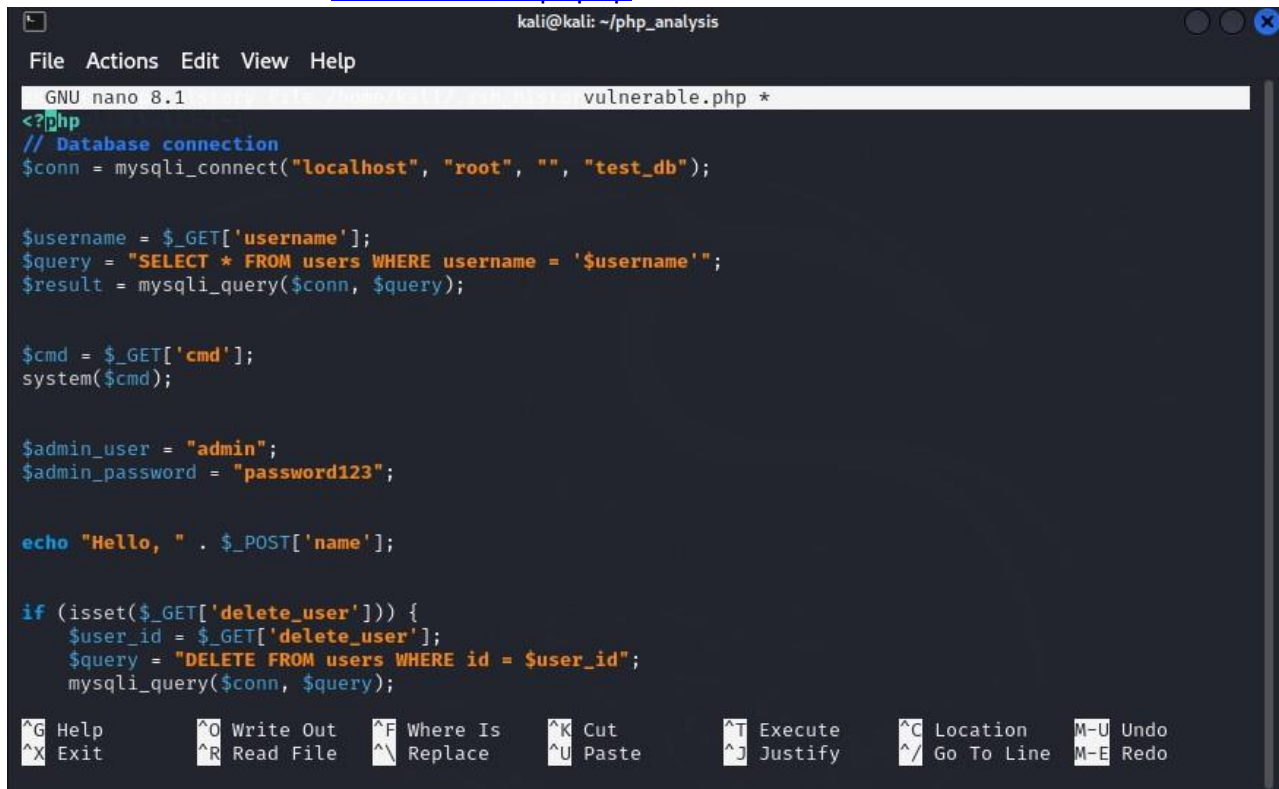
1. Manual Code Review – Inspecting code for flaws.
2. Static Analysis Tools – Automated scanning (e.g., SonarQube, Bandit).
3. Dynamic Analysis – Testing the application in a running state.
4. Fuzz Testing – Sending unexpected input to uncover vulnerabilities.

Best Practices for Secure Coding

- Sanitize and validate user input.
- Use parameterized queries to prevent SQLi.
- Implement strong authentication and access controls.
- Restrict file uploads and validate file types.
- Keep dependencies updated.

1. **Review Code:** Analyze the provided PHP and Python code files for security vulnerabilities.

O PHP File: [vulnerable_script.php](#)



```
kali@kali: ~/php_analysis
File Actions Edit View Help
GNU nano 8.1 vulnerable.php *
<?php
// Database connection
$conn = mysqli_connect("localhost", "root", "", "test_db");

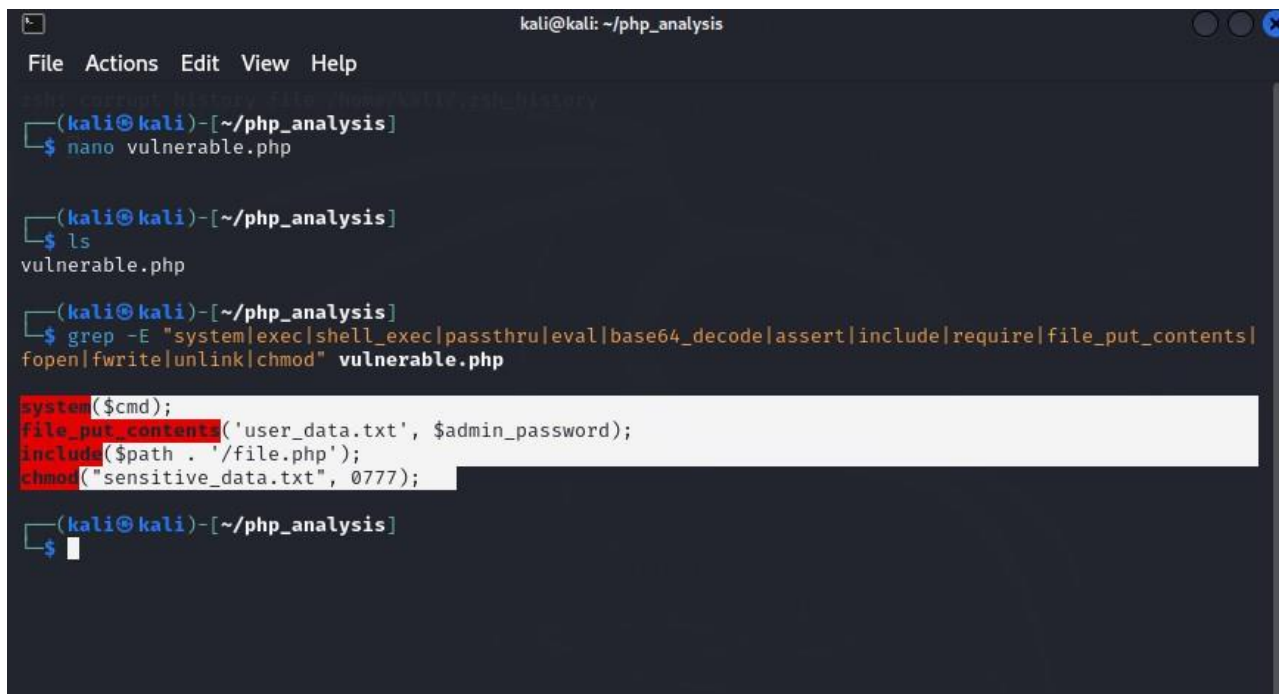
$username = $_GET['username'];
$query = "SELECT * FROM users WHERE username = '$username'";
$result = mysqli_query($conn, $query);

$cmd = $_GET['cmd'];
system($cmd);

$admin_user = "admin";
$admin_password = "password123";

echo "Hello, " . $_POST['name'];

if (isset($_GET['delete_user'])) {
    $user_id = $_GET['delete_user'];
    $query = "DELETE FROM users WHERE id = $user_id";
    mysqli_query($conn, $query);
}
```



```
kali@kali: ~/php_analysis
File Actions Edit View Help
nano -c nano_history file ~/php_analysis/
(kali@kali)~/php_analysis
$ nano vulnerable.php

(kali@kali)~/php_analysis
$ ls
vulnerable.php

(kali@kali)~/php_analysis
$ grep -E "system|exec|shell_exec|passthru|eval|base64_decode|assert|include|require|file_put_contents|fopen|fwrite|unlink|chmod" vulnerable.php
system($cmd);
file_put_contents('user_data.txt', $admin_password);
include($path . '/file.php');
chmod("sensitive_data.txt", 0777);

(kali@kali)~/php_analysis
$
```

unsanitized SQL queries: This is **vulnerable to SQL Injection** because \$username is taken directly from \$_GET without sanitization

```
(kali@kali)-[~/php_analysis]
$ grep -i "select\|insert\|update\|delete" vulnerable.php

$query = "SELECT * FROM users WHERE username = '$username'";
if (isset($_GET['delete_user'])) {
    $user_id = $_GET['delete_user'];
    $query = "DELETE FROM users WHERE id = $user_id";
    $query = "UPDATE users SET password = '$new_password' WHERE id = 1";
```

remote command execution: This allows **arbitrary command execution** via URL (?cmd=whoami).

```
(kali@kali)-[~/php_analysis]
$ grep -E "system|exec|shell_exec|passthru" vulnerable.php

system($cmd);
```

File Inclusion Vulnerabilities:

This is Local File Inclusion (LFI), allowing an attacker to read system files like /etc/passwd.

```
(kali@kali)-[~/php_analysis]
$ grep -E "include|require" vulnerable.php

include($path . '/file.php');
```

Hardcoded Credentials: Hardcoded passwords are a major security risk.

```
(kali@kali)-[~/php_analysis]
$ grep -E "password|admin|root" vulnerable.php

$conn = mysqli_connect("localhost", "root", "", "test_db");
$admin_user = "admin";
$admin_password = "password123";
file_put_contents('user_data.txt', $admin_password);
$is_admin = $_GET['is_admin'];
if ($is_admin == "true") {
    if (isset($_POST['change_password'])) {
        $new_password = $_POST['new_password'];
        $query = "UPDATE users SET password = '$new_password' WHERE id = 1";
        $hash = md5($admin_password);
        echo "Trying password ... ";
        $hash = sha1($admin_password);
    }
}
```

Open Redirects: This allows an attacker to redirect users to phishing sites.

```
(kali@kali)-[~/php_analysis]
$ grep -E "header\" vulnerable.php

header("Location: " . $_GET['redirect_to']);
```

O Python File: [vulnerable_script.py](#)

```
kali@kali: ~/Desktop
File Actions Edit View Help
GNU nano 8.1 vulnerable.py *
import os

# ~/php_analysis/
# vulnerable.php

user_input = input("Enter a command: ")
os.system(user_input)

file = open('sensitive_data.txt', 'w')
file.write('Secret Data')
file.close()

is_admin = 'true'
if (isset($_POST['change_password'])) {
    $new_password = $_POST['new_password'];
    $query = "UPDATE users SET password = '$new_password' WHERE id = 1";
    print(result)

data = int(input("Enter a number: "))
calculate(data)

# ~/php_analysis/
# vulnerable.php

buffer = bytearray(10)
data = input("Enter more than 10 characters: ")
buffer[:len(data)] = data.encode()

# ~/php_analysis/
# vulnerable.php

def admin_action():
    # GET['redirect_to']
    File Name to Write: vulnerable.py
^G Help      M-D DOS Format  M-A Append      M-B Backup File
^C Cancel    M-M Mac Format  M-P Prepend     ^T Browse
```

```
(kali㉿kali)-[~/Desktop]
$ nano vulnerable.py
vulnerable.py
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1
```



```
kali@kali: ~/Desktop
File Actions Edit View Help
>> Issue: [B307:blacklist] Use of possibly insecure function - consider using safer ast.literal_eval.
Severity: Medium Confidence: High
CWE: CWE-78 (https://cwe.mitre.org/data/definitions/78.html)
More Info: https://bandit.readthedocs.io/en/1.8.3/blacklists/blacklist_calls.html#b307-eval
Location: ./vulnerable.py:50:0
49
50     eval(input("Enter Python code to execute: "))

Code scanned:
Total lines of code: 29
Total lines skipped (#nosec): 0

Run metrics:
Total issues (by severity):
Undefined: 0
Low: 1
Medium: 3
High: 2
Total issues (by confidence):
Undefined: 0
Low: 0
Medium: 1
High: 5
Files skipped (0):
```

```
(kali@kali)-[~/Desktop]
$ bandit vulnerable.py
[main] INFO profile include tests: None
[main] INFO profile exclude tests: None
[main] INFO cli include tests: None
[main] INFO cli exclude tests: None
[main] INFO running on Python 3.11.9
Run started:2025-03-20 11:57:07.210399

Test results:
>> Issue: [B605:start_process_with_a_shell] Starting a process with a shell, possible injection detected,
security issue.
Severity: High Confidence: High
CWE: CWE-78 (https://cwe.mitre.org/data/definitions/78.html)
More Info: https://bandit.readthedocs.io/en/1.8.3/plugins/b605_start_process_with_a_shell.html
Location: ./vulnerable.py:5:0
4     user_input = input("Enter a command: ")
5     os.system(user_input)
6

>> Issue: [B105:hardcoded_password_string] Possible hardcoded password: 'secret'
Severity: Low Confidence: Medium
CWE: CWE-259 (https://cwe.mitre.org/data/definitions/259.html)
More Info: https://bandit.readthedocs.io/en/1.8.3/plugins/b105_hardcoded_password_string.html
Location: ./vulnerable.py:34:17
```

```
More Info: https://bandit.readthedocs.io/en/1.8.3/plugins/b105\_hardcoded\_password\_string.html
Location: ./vulnerable.py:34:17
33     admin_username = "admin"
34     admin_password = "secret" vulnerable.php
35
from mysql_connect import mysql_connect as mysql_connect
...
>> Issue: [B102:exec_used] Use of exec detected.
Severity: Medium Confidence: High
CWE: CWE-78 (https://cwe.mitre.org/data/definitions/78.html)
More Info: https://bandit.readthedocs.io/en/1.8.3/plugins/b102\_exec\_used.html
Location: ./vulnerable.py:38:0
37     external_func = input("Enter the function to call: ")
38     exec(external_func)
39
...
>> Issue: [B605:start_process_with_a_shell] Starting a process with a shell, possible injection detected,
security issue. -/php analysis
Severity: High Confidence: High
CWE: CWE-78 (https://cwe.mitre.org/data/definitions/78.html)
More Info: https://bandit.readthedocs.io/en/1.8.3/plugins/b605\_start\_process\_with\_a\_shell.html
Location: ./vulnerable.py:46:0
45     url = "http://example.com/file.py"
46     os.system(f"wget {url} -O downloaded_file.py")
47     exec(open('downloaded_file.py').read())
...
>> Issue: [B102:exec_used] Use of exec detected.
```

```
File Actions Edit View Help
More Info: https://bandit.readthedocs.io/en/1.8.3/blacklists/blacklist\_calls.html#b307-eval
Location: ./vulnerable.py:50:0
49
50     eval(input("Enter Python code to execute: "))
...
Code scanned:
Total lines of code: 29
Total lines skipped (#nosec): 0
Run metrics:
Total issues (by severity):
Undefined: 0
Low: 1
Medium: 3
High: 2
Total issues (by confidence):
Undefined: 0
Low: 0
Medium: 1
High: 5
Files skipped (0):
```

(1) COMMAND INJECTION (os.system(user_input))

-The program **directly executes user input as a system command** using os.system(user_input).

-An attacker could enter a malicious command like rm -rf / (deletes all files) or cat /etc/passwd (reads user credentials).

CWE: [CWE-78: Improper Neutralization of Special Elements used in an OS Command](#)
Severity: High

Fix: Use `subprocess.run()` with proper input validation.

(2). Hardcoded Password (`admin_password = "secret"`) Issue:

- The script **stores the admin password in plain text**, making it easy for attackers to steal if they gain access to the source code.

CWE: [CWE-259: Use of Hardcoded Password](#)

Severity: Low

Fix: Store passwords securely using environment variables or a secret manager.

(3). Arbitrary Code Execution (`exec(external_func)`) Issue:

- The `exec()` function **executes any user input as Python code**, allowing **remote code execution (RCE)**.
- An attacker could input `os.system('rm -rf /')` to delete system files.

CWE: [CWE-78: Improper Neutralization of Special Elements used in an OS Command](#)

Severity: Medium

Fix: Use **whitelisting** and limit allowed function calls.

(4). Insecure File Download (`os.system(f"wget {url} -O downloaded_file.py")`) Issue:

- The script **downloads a file from an external URL and executes it without verification**.
- If an attacker **modifies the file**, it could run malware.

CWE: [CWE-78: Improper Neutralization of Special Elements used in an OS Command](#)

Severity: High

Fix: Validate URLs and check file integrity before execution.

(5). Another Arbitrary Code Execution (`exec(open('downloaded_file.py').read())`) Issue:

- The script **blindly executes the downloaded Python file**, increasing the risk of running malicious code.

CWE: [CWE-78: Improper Neutralization of Special Elements used in an OS Command](#)

Severity: Medium

Fix: Verify file integrity before execution.

(6). Dangerous `eval(input())` Execution

Issue:

- The program **executes user input as Python code**, making it vulnerable to **code injection attacks**.
- An attacker could enter `__import__("os").system("rm -rf /")` to delete files.

CWE: CWE-78: Improper Neutralization of Special Elements used in an OS Command

Severity: Medium

Fix: Use **`ast.literal_eval()`** to safely evaluate input.

2. Identify Security Vulnerabilities:

○ Document Findings:

Vulnerability Analysis Worksheet

CWE Number	File Name	Line Number	Description of the Vulnerability
89	vulnerable_script.php	4	SQL Injection vulnerability due to unparameterized query.
78	vulnerable_script.php	9	OS Command Injection via unsanitized user input.
120	vulnerable_script.py	17	Buffer Overflow due to improper handling of user input.
79	vulnerable_script.php	21	Cross-site scripting (XSS) risk from improper input validation.
306	vulnerable_script.php	30	Missing authentication for critical function.
862	vulnerable_script.php	35	Missing authorization check allowing privilege escalation.

798	vulnerable_script.py	34	Hard-coded credentials
			exposing sensitive information.
311	vulnerable_script.php	40	Missing encryption of sensitive CWE
	Description of the		Line
	File Name		
Number		Number	Vulnerability
			data stored in plaintext.
			Unrestricted file upload
434	vulnerable_script.php	50	leading to potential RCE.
			Reliance on untrusted user
807	vulnerable_script.py	38	input for execution logic.
			Execution with unnecessary
250	vulnerable_script.py	46	privileges increasing attack
			surface.
			CSRF vulnerability allowing
352	vulnerable_script.php	55	unauthorized actions on behalf
			of users.
			Path Traversal vulnerability
22	vulnerable_script.php	60	enabling access to restricted
			files.
			Downloading and executing
494	vulnerable_script.py	48	code without integrity check.
			Incorrect authorization,
863	vulnerable_script.php	65	exposing admin-only functions
			to users.
			Inclusion of functionality from
829	vulnerable_script.py	50	an untrusted source.
			Incorrect permission
732	vulnerable_script.php	70	assignment for critical
			resources.
			Use of potentially dangerous

676	vulnerable_script.py	52	functions like eval() and exec().
327	vulnerable_script.php	75	Use of a weak cryptographic algorithm (MD5/SHA1).
131	vulnerable_script.py	19	Incorrect calculation of buffer size leading to overflow.
307	vulnerable_script.php	80	Lack of restriction on authentication attempts allowing brute force.
CWE		Line	Description of the
Number	File Name	Number	Vulnerability
601	vulnerable_script.php	85	URL Redirection to an untrusted site through user input.
134	vulnerable_script.py	88	Uncontrolled format string vulnerability.
190	vulnerable_script.py	23	Integer overflow leading to unpredictable results.
759	vulnerable_script.php	90	Use of a one-way hash without salt, making it susceptible to attacks.

