BASH برای تست نفوذ وب

FROM A FRIEND

تعریف یک متغییر(دقت کنید نباید فاصله ای بعد و قبل = باشه):

```
age=10
say=“hello world..”
```

- نمایش متغییر:

```
echo $age
echo "salam doste $age saleh"
printf $say
```

- حذف متغییر:

```
unset age
```

بدست آوردن طول متغییر:

```
len=“hello world..”
echo ${#len}
```

استفاده از دستورات لینوکسی در متغییرها:

```
path=$(pwd)
echo $path
```

گرفتن ورودی از کاربر:

- روش اول گرفتن ورودی از کاربر بعد از اجرا کردن برنامه:

```
read age
read -p “chand salete? ” age
```

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ read -p "chand salete? " age
chand salete? 12
```

- روش دوم گرفتن ورودی از کاربر موقع اجرای برنامه:

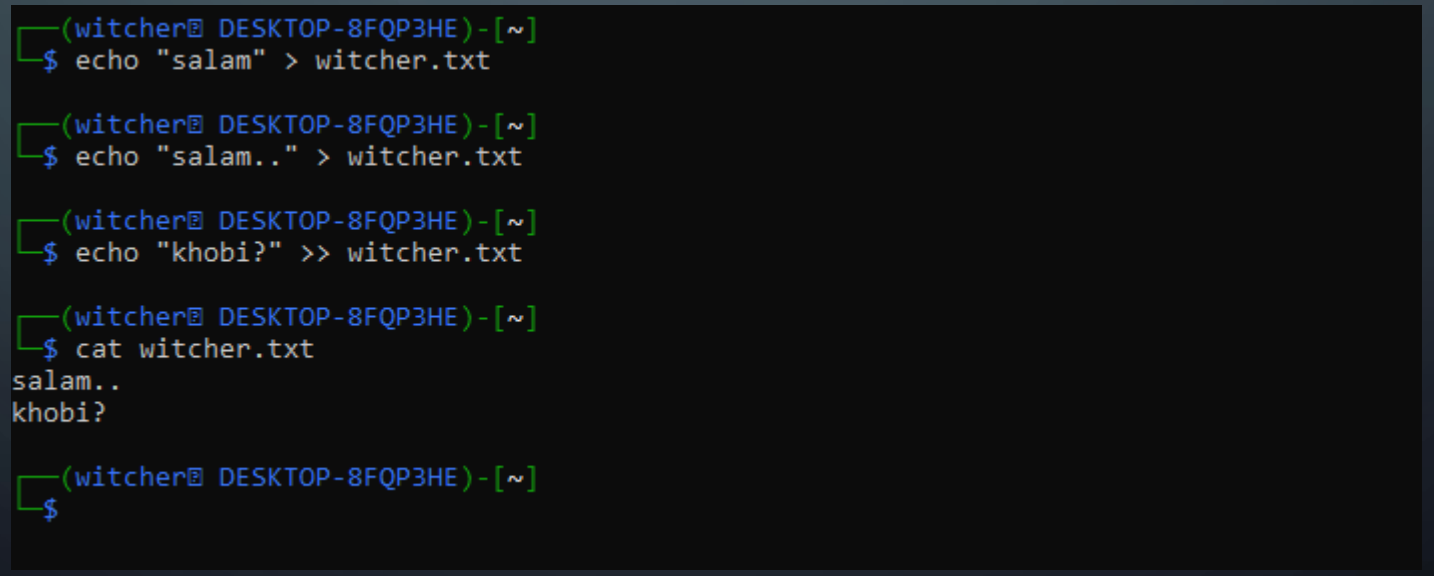
```
fname=$1
lname=$2
```

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ cat name.sh
fname=$1
lname=$2

echo "salam $fname $lname"

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ bash name.sh bita goli
salam bita goli
```

- میتوانیم دستورات داخل یک فایل با پسوند sh بنویسیم و با نوشتن bash و نام فایل اجراش کنیم.

اوپراتورهای > و >>:

- اگر از > استفاده کنیم میتونیم خروجی دستورات داخل یک فایل بریزیم و هربار ازش استفاده شود مقادیر داخل فایل قبلی حذف و مقدار جدید جایگزین میشود.
- برای حذف نشدن مقادیر قبلی باید از >> استفاده کرد اینطور مقادیر قبلی جای خودشان میمانند و مقادیر جدید به فایل اضافه میشود.



اوپراتورهای &> و &>>:

- این اوپراتورها تمام خروجی دستورات داخل یک فایل مینویسند دستور اول به دلیل نصب نبودن این ابزار خطای پیدا نشدن دستور نمایش میدهد و با استفاده از &> این خطا داخل فایل نوشتیم و با &>> خطای دوم هم به فاصل اضافه کردیم.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ cmatrix
Command 'cmatrix' not found, but can be installed with:
sudo apt install cmatrix

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ cmatriii
cmatriii: command not found

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ cmatrix &> witcher.txt

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ cmatiii &>> witcher.txt

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ cat witcher.txt
Command 'cmatrix' not found, but can be installed with:
sudo apt install cmatrix
cmatiii: command not found

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

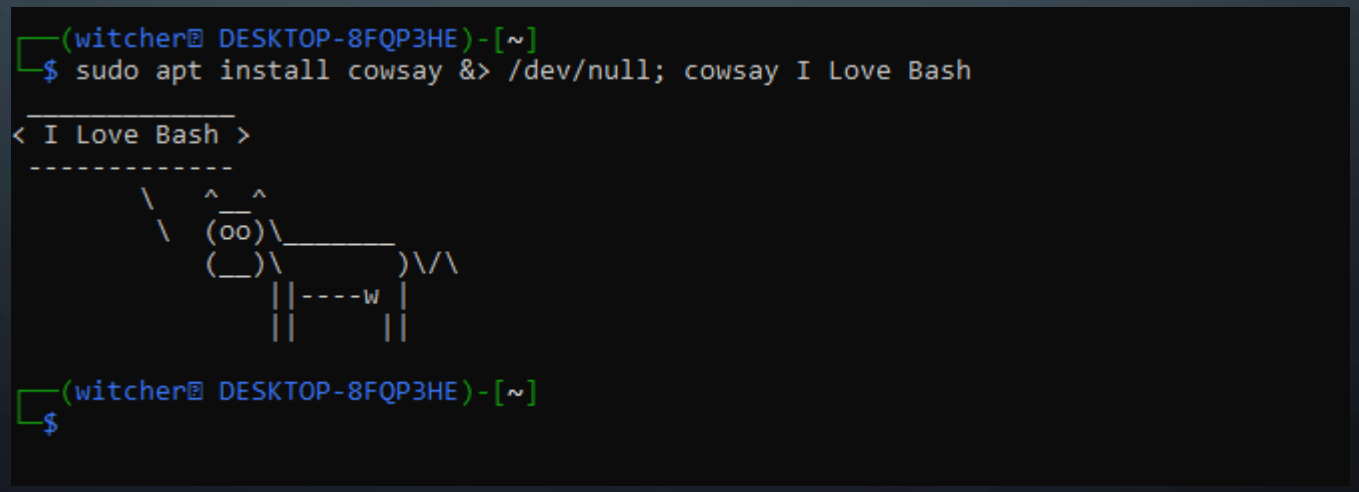
اوپراتور 2>:

- دستور اول echo اشتباه نوشته شده و یک ارور بلند بالا چاپ کرد برای نمایش داده نشدن این ارورها از 2> استفاده میکنیم.
- در خط دوم اگر اروری وجود داشته باشد 2> ارور را به مسیر داده شده یعنی /dev/null همون ناکجا آباد میریزد.
- در خط سوم چون echo درست نوشته شده پس اروری ندارد و دستور اجرا میشود.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ ech hello
Command 'ech' not found, did you mean:
  command 'dch' from deb devscripts
  command 'ecj' from deb ecj
  command 'echo' from deb coreutils
  command 'ecl' from deb ecl
  command 'ecm' from deb gmp-ecm
Try: sudo apt install <deb name>

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ ech hello 2> /dev/null

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ echo hello 2> /dev/null
hello
```

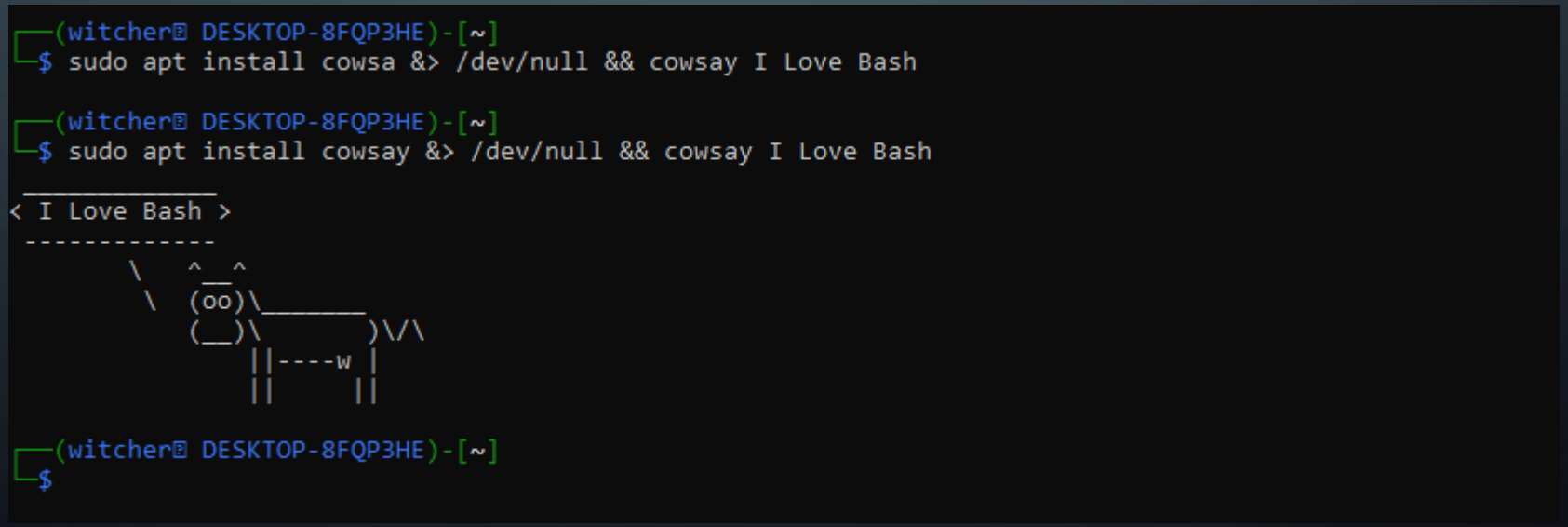
اوپراتور ; و اجرای چند دستور پشت سرهم:

- به ترتیب یکی یکی دستورات اجرا میکند
- اول ابزار cowsay نصب میکنه و برای نشان داده نشدن اضافیات دستور نصب با استفاده از &> به /dev/null ارسالشون میکنیم.
- بعد با ; دستور بعدی خودمون که اجرای این ابزار هست مینویسیم و همینطور که میبینید بعد از نصب ابزار اجرا میشود.



اوپراتور &&:

- تا وقتی دستورات درست باشند ادامه میدهد.
- خط اول اگر دقت کنید میبینید به اشتباه نوشته شده cowsa و برای همین چون اشتباه بود دیگر ادامه نداد و دستورات متوقف شدند.
- خط دوم هم دستور قبل && درست بود هم دستور بعدش برای همین هردو اجرا شدند اول ابزار نصب شد و بعد اجرا شد.



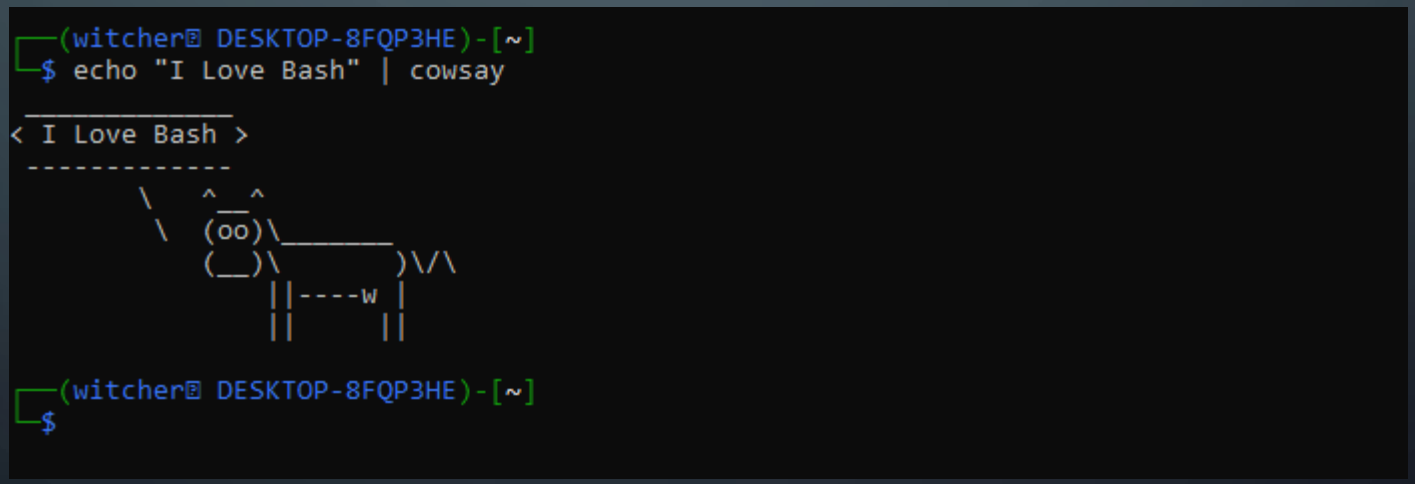
اپراتور ||:

- تا موقعی ادامه میدهد که به دستور درست برسد.
- همینطور که میبینید خط اول، اولین echo درست بود پس نشان داده شد.
- خط دوم اولین و دومین echo به اشتباه ech نوشته شدند پس سومین echo اجرا شد.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ devn="/dev/null" ; echo "salam" 2> $devn || echo "man daram" 2> $devn || echo "bash" 2> $devn || echo "yad migiram" 2> $devn
salam

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ devn="/dev/null" ; ech "salam" 2> $devn || ech "man daram" 2> $devn || echo "bash" 2> $devn || echo "yad migiram" 2> $devn
bash

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

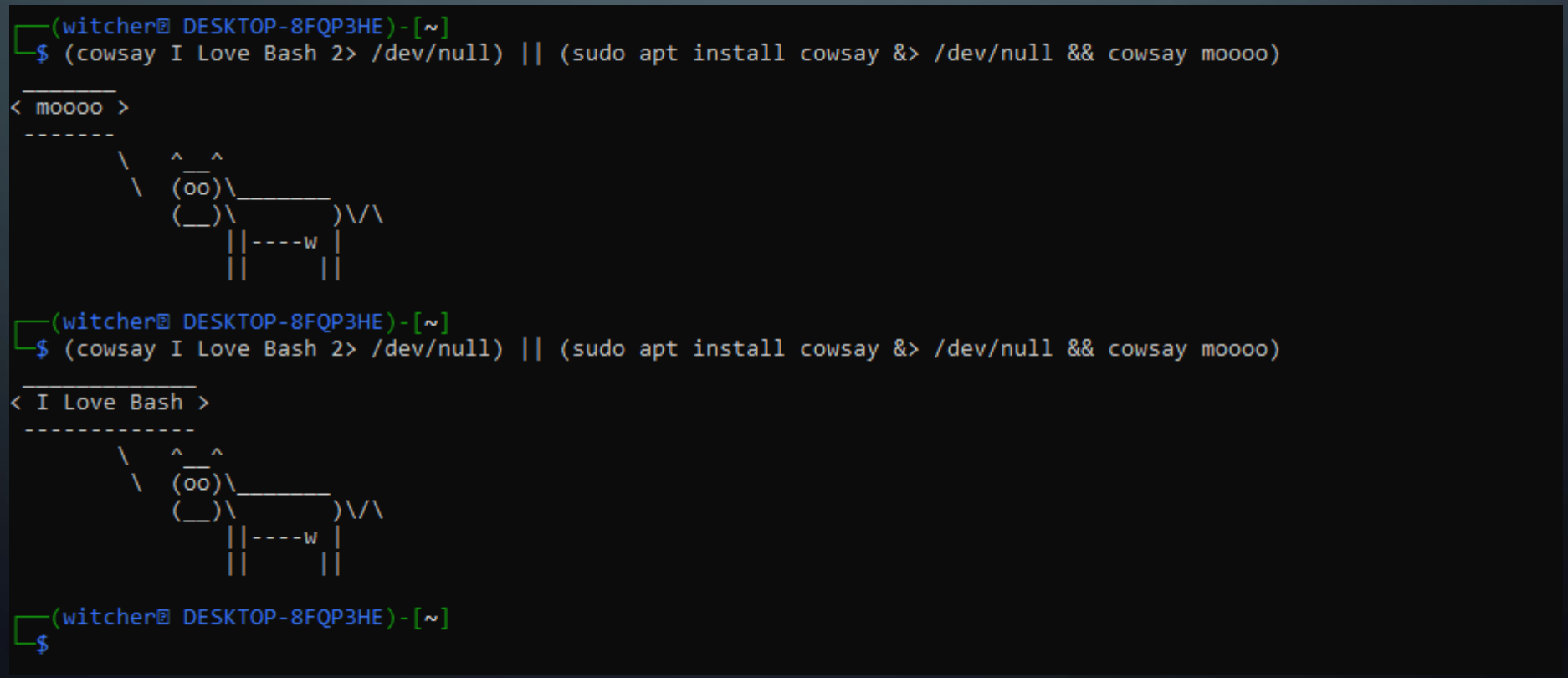
اوپراتور |

- خروجی دستورهدایت میکند به دستور جدید
- با استفاده از | خروجی دستور قبل که نمایش یک پیام توسط echo بود به دستور بعد یعنی cowsay داده شد و به این شکل پیام نمایش داده شد.



گروه‌ها ():

- حالا ما میتوانیم این دستورات کنترلی که یادگرفتیم پشت سرهم بچینیم و ازشون استفاده کنیم.
- خط اول اگر cowsay نصب باشد اجرا میشود و دیگر ادامه نمیدهد اما اگر نصب نباشد دستور بعد از || اجرا میشود و اول نصب و بعد از اینکه نصب شد cowsay اجرا میشود.
- خط دوم چون الان cowsay نصب هست پس دستورات بعد از || اجرا نمیشوند.



آرایه‌ها در BASH

- خط اول یک آرایه تعریف شده که دارای 3 مقدار میباشد.
- خط دوم با حالت * هست به معنی تمامی مقادیر نشان داده شود.
- خط سوم عدد 0 به معنی اولین خانه از آرایه نشان داده شود.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ cat witcher.sh
IP_ADDRESSES=(192.168.1.1 192.168.1.2 192.168.1.3)

echo "${IP_ADDRESSES[*]}"
echo "${IP_ADDRESSES[0]}"

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ bash witcher.sh
192.168.1.1 192.168.1.2 192.168.1.3
192.168.1.1

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

- برای اضافه کردن یک مقدار به آرایه برای مثال:

IP_ADDRESSES[3]=1.1.1.1

دستورات شرطی (فاصله‌ها خیلی مهم هستند حواستون باشه!):

```
age=5
if [ $age -ge 1 ] && [ $age -le 10 ]; then
   echo "kocholo"
elif [ $age -gt 10 ]; then
   echo "bozorg shodia"
fi
```

شرط اول میگه اگر age بزرگتر مساوی 1 و کوچکتر مساوی 10 بود پیام kocholo نمایش بده.

شرط دوم هم میگه در غیر اینصورت اگر(یعنی اگر شرط اول برقرار نبود) age بزرگتر از 10 بود پیام bozorg shodia نشون بده.

دستورات	توضیحات
if [age -eq 10] if [age = 10]	اگر age برابر با 10 بود
if [age != 10] if [age -ne 10]	اگر age مخالف 10 بود
if [age -gt 10]	اگر age بزرگتر از 10 بود
if [age -lt 10]	اگر age کوچکتر از 10 بود
if [age -ge 10]	اگر age بزرگتر مساوی 10 بود
if [age -le 10]	اگر age کوچکتر مساوی 10 بود
if [age -gt 10] && [age != 20]	اگر age بزرگتر از 10 و مخالف 20 بود
if [age -eq 10] || [age -eq 20]	اگر age برابر با 10 یا 20 بود

چیزهایی که تا اینجا یادگرفتیم یک تمرین کوچک انجام بدیم:

- اول شما بنویسید بعد کد زیر نگاه کنید:
- برنامه ای بنویسید که سن کاربر بگیرد و بررسی کند اگر کوچکتر از 10 و بزرگتر مساوی 1 بود پیام salam kocholo درغیر این صورت اگر بزرگتر مساوی 10 بود پیام salam bozorg shodia نشان بدهد.

```
read -p "chand salete? " age
if [ $age -ge 1 ] && [ $age -lt 10 ]; then
   echo "kocholo"
elif [ $age -ge 10 ]; then
   echo "bozorg shodia"
fi
```

حلقه‌های for :

- روش اول: از 1 تا 5 ادامه میدهد و یکی یکی اعداد نمایش داده میشود.

```
for i in {1..5}; do
    echo $i
Done
```

- روش دوم: تا وقتی i کوچکتر مساوی 5 هست ادامه میدهد و یکی یکی اعداد نمایش داده میشود.

```
for (( i = 1; i <= 5; i++ )); do
    echo $i
done
```

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ cat witcher.sh
for (( i = 1; i <= 5; i++ )); do
    echo $i
done


┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ bash witcher.sh
1
2
3
4
5

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

حلقه‌های while :

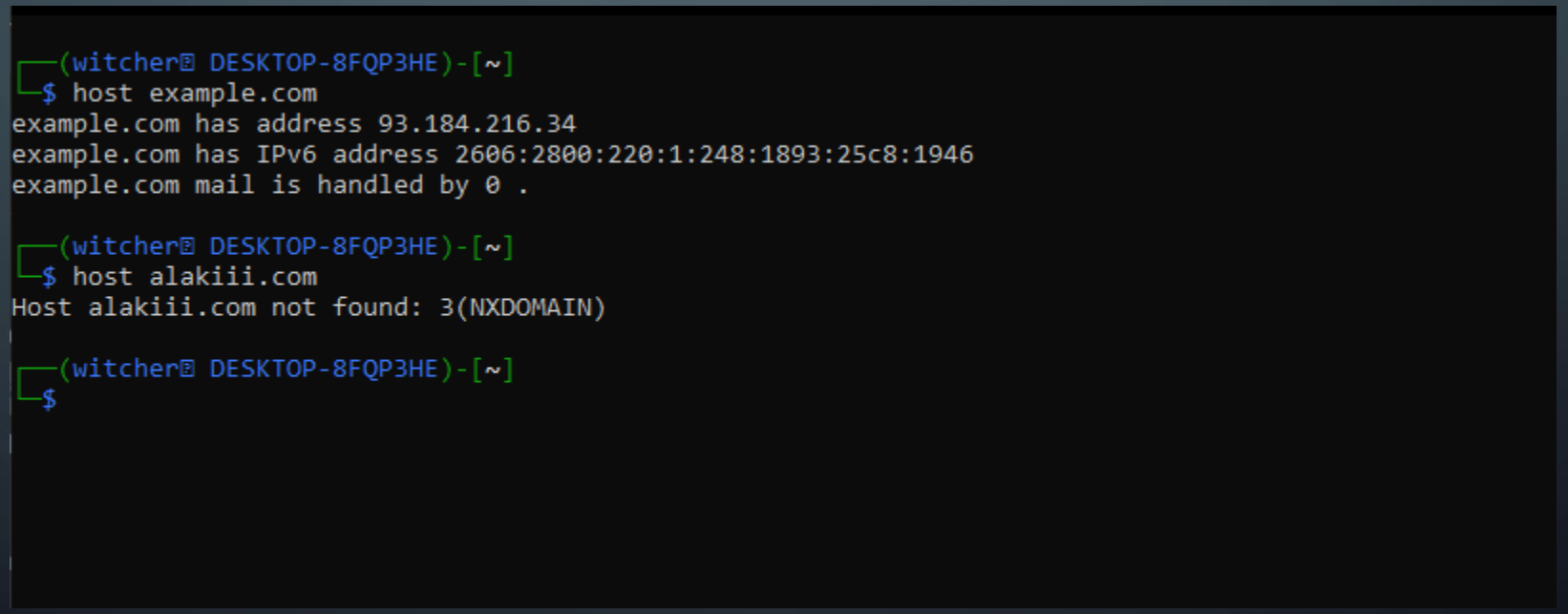
- تا وقتی که counter کوچکتر مساوی 10 باشد ادامه میدهد و بعد از نشان دادن counter هربار به اضافه 1 میکند.

```
counter=1
while [ $counter -le 10 ]; do
   echo $counter
   ((counter++))
done
```

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ bash while.sh
1
2
3
4
5
6
7
8
9
10
```

ابزار HOST

- این ابزار اطلاعات DNS مربوط به یک دامنه را بررسی میکند.
- چشمی تفاوتش را میتوانید ببینید دامنه دوم نتیجه‌ای ندارد.



حالا کد زیر تست کنید تا نتیجه نیاز به سنجیدن چشمی نباشد

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ host example.com
example.com has address 93.184.216.34
example.com has IPv6 address 2606:2800:220:1:248:1893:25c8:1946
example.com mail is handled by 0 .

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ echo $?
0

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ host alakiii.com
Host alakiii.com not found: 3(NXDOMAIN)

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ echo $?
1

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

استفاده از دستورات شرطی برای مشاهده ساب‌دامنه‌های دارای نتیجه:

اگر دامنه نتیجه‌ای داشته باشد کلمه یافتم مثل مورد یک نمایش داده میشود.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ if host example.com; then echo "[+] - yaftammm"; fi
example.com has address 93.184.216.34
example.com has IPv6 address 2606:2800:220:1:248:1893:25c8:1946
example.com mail is handled by 0 .
[+] - yaftammm

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ if host alakiii.com; then echo "[+] - yaftammm"; fi
Host alakiii.com not found: 3(NXDOMAIN)

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

حالا از &> /DEV/NULL استفاده میکنیم تا اطلاعات اضافی حذف بشوند.
میتوانید & هم ننویسید برای مثال: > /DEV/NULL

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ if host example.com &> /dev/null; then echo "[+] - yaftammm"; fi
[+] - yaftammm

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ if host alakiii.com &> /dev/null; then echo "[+] - yaftammm"; fi

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ if host alakiii.com &> /dev/null; then echo "[+] - yaftammm";else echo "[-] - nayaftam :("; fi
[-] - nayaftam :(

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

استفاده از حلقه ها برای نمایش خط به خط یک فایل:

- همینطور که میبینید با read به این صورت میتوانید یک خط از فایل نمایش بدید.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ cat subs.txt
www
test
admin
blog
hello
developer

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ read age < subs.txt; echo $age
www
```

- پس برای دریافت تمام خط فایل‌ها نیاز هست حلقه بزنیم روی فایل تا خط به خط گرفته و به متغییر subs داده شود و توسط $subs به example.com چسبانده میشود.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ while read subs; do echo "$subs.example.com"; done < subs.txt
www.example.com
test.example.com
admin.example.com
blog.example.com
hello.example.com
developer.example.com

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

ترکیب کردن چیزایی که تا الان یادگرفتیم

- حالا میتوانیم یکی یکی ساب‌دامنه‌ها را با این ابزار بررسی کنیم و آنهایی که نتیجه دارند ببینیم.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ while read subs; do if host "$subs.example.com" &> /dev/null; then echo "$subs.example.com"; fi; done < subs.txt
www.example.com

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

- برای دامنه هم یک ورودی با متغییر domain تعریف میکنیم تا هربار دستی واردش نکنیم.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ read -p "domain: " domain; while read subs; do if host "$subs.$domain" &> /dev/null; then echo "$subs.$domain"; fi; done < subs.txt
domain: example.com
www.example.com

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

نوشتن برنامه قبلی داخل یک فایل

- به این صورت با دادن دامنه و فایل subs.txt تمام خطوط دریافت میکند و به دامنه میچسباند و موجود بودن ساب‌دامنه بررسی میشود.

```
┌──(witcher㉿DESKTOP-8FQP3HE)-[~]
└─$ cat witcher.sh
domain=$1

while read subs; do
   if host $subs.$domain > /dev/null; then
      echo $subs.$domain
   fi
done

┌──(witcher㉿DESKTOP-8FQP3HE)-[~]
└─$ bash witcher.sh harvard.edu < subs.txt
www.harvard.edu
test.harvard.edu

┌──(witcher㉿DESKTOP-8FQP3HE)-[~]
└─$ cat subs.txt | bash witcher.sh harvard.edu
www.harvard.edu
test.harvard.edu
```

- مثل خط اخر میشود به جای > از | استفاده کرد.

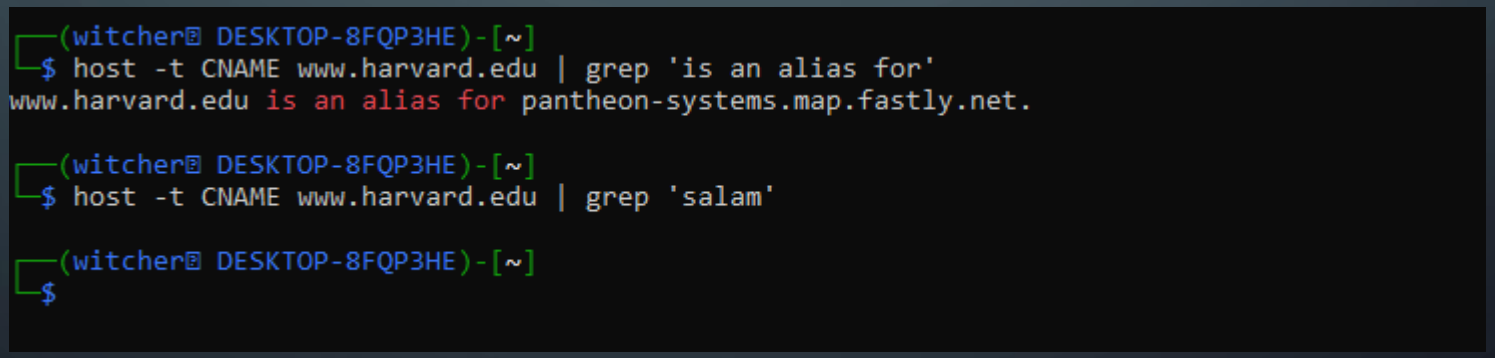
اجرای راحت‌تر فایل BASH

- خط اول که مشخص هست خطا داده چون هنوز دسترسی مجاز برای اجرا شدن را ندارد.
- خط دوم دسترسی را با chmod +x به فایل bash میدهیم.
- حالا در خط سوم خیلی راحت فایل bash را اجرا میکنیم.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ ./witcher.sh example.com < subs.txt
-bash: ./witcher.sh: Permission denied

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ chmod +x witcher.sh

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ ./witcher.sh example.com < subs.txt
www.example.com

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

دستور grep:

- از این دستور برای جست‌وجوی کلمات و جملات در متن استفاده میکنیم.
- در خط اول ما به دنبال یک جمله بودیم و پیدا شد برای همین با رنگ قرمز نشان داد و خط‌های دیگر نشان داده نشد.
- خط دوم ما به دنبال کلمه salam بودیم و چون چنین کلمه ای وجود نداشت چیزی نشان داده نشد.



زبان AWK:

- یک زبان به شدت قوی برای کار با متن‌ها که میشود همراه bash ازش استفاده کرد.
- متنی که خط اول نشان داده شده 6 کلمه یا بهتره بگیم بخش هست پس ما برای دسترسی به هر بخش با $ مشخص میکنیم برای مثال ما اخرین بخش نیاز داشتیم و $6 نوشتیم.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ host -t CNAME www.harvard.edu | grep 'is an alias for'
www.harvard.edu is an alias for pantheon-systems.map.fastly.net.

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ host -t CNAME www.harvard.edu | grep 'is an alias for' | awk '{print $6}'
pantheon-systems.map.fastly.net.
```

استفاده از NF:

- خط اول دستور host استفاده کردیم تا یکسری اطلاعات از www.harvard.edu نشان داده شود.
- در خط دوم با استفاده از NF میتوانیم بفهمیم هر خط چند بخش دارد.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ host www.harvard.edu
www.harvard.edu is an alias for pantheon-systems.map.fastly.net.
pantheon-systems.map.fastly.net has address 151.101.2.133
pantheon-systems.map.fastly.net has address 151.101.66.133
pantheon-systems.map.fastly.net has address 151.101.194.133
pantheon-systems.map.fastly.net has address 151.101.130.133
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42::645
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42:200::645
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42:400::645
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42:600::645

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ host www.harvard.edu | awk '{print NF}'
6
4
4
4
4
5
5
5
5
```

استفاده از $NF:

- خط اول $NF آخرین بخش از هر خط نشان میدهد دیگر نیاز نیست دستی با $ و یک عدد مشخص کنیم.
- خط دوم شرط گزاشتیم اگر NF == 6 بود نمایش داده شود یعنی اگر اون خط 6 بخش بود اخرین بخشش نمایش داده میشود.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ host www.harvard.edu | awk '{print $NF}'
pantheon-systems.map.fastly.net.
151.101.2.133
151.101.66.133
151.101.130.133
151.101.194.133
2a04:4e42::645
2a04:4e42:200::645
2a04:4e42:400::645
2a04:4e42:600::645

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ host www.harvard.edu | awk 'NF == 6 {print $NF}'
pantheon-systems.map.fastly.net.

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

استفاده از NR:

- خط دوم با استفاده از END میتوانیم آخرین خط را نمایش دهیم.
- خط سوم با استفاده از NR مشخص میکنیم کدام خطها نیاز داریم برای مثال گفتیم اگر NR کوچکتر مساوی 3 نمایش داده شود برای همین 3 خط اول فقط نمایش داده میشود.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ host www.harvard.edu
www.harvard.edu is an alias for pantheon-systems.map.fastly.net.
pantheon-systems.map.fastly.net has address 151.101.2.133
pantheon-systems.map.fastly.net has address 151.101.66.133
pantheon-systems.map.fastly.net has address 151.101.194.133
pantheon-systems.map.fastly.net has address 151.101.130.133
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42:400::645
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42:200::645
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42::645
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42:600::645

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ host www.harvard.edu | awk 'END {print}'
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42:600::645

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ host www.harvard.edu | awk 'NR <= 3 {print}'
www.harvard.edu is an alias for pantheon-systems.map.fastly.net.
pantheon-systems.map.fastly.net has address 151.101.2.133
pantheon-systems.map.fastly.net has address 151.101.66.133
```

نکته تکمیلی AWK:

- با F– مشخص میکنیم برچه اساسی جداسازی انجام بگیره در مثال‌های قبلی بر اساس فاصله بود اما اینبار گفتیم بر اساس نقطه باشه و با $1 میخایم که براساس نقطه اولین بخش نمایش بدهد.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ host www.harvard.edu
www.harvard.edu is an alias for pantheon-systems.map.fastly.net.
pantheon-systems.map.fastly.net has address 151.101.66.133
pantheon-systems.map.fastly.net has address 151.101.194.133
pantheon-systems.map.fastly.net has address 151.101.2.133
pantheon-systems.map.fastly.net has address 151.101.130.133
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42::645
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42:200::645
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42:400::645
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42:600::645

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ host www.harvard.edu | awk -F. '{print $1}'
www
pantheon-systems
pantheon-systems
pantheon-systems
pantheon-systems
pantheon-systems
pantheon-systems
pantheon-systems
pantheon-systems

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

ویرایشگر متن sed:

- s به معنی جایگزین کردن میباشد و g به معنی این هست برای تمام متن این اتفاق بیوفتد.
- قسمت اول قرمز رنگ / / به این شکل مشخص کردیم هرجا در متن فاصله دید با ---- جایگزین کند.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ host www.harvard.edu | sed 's/ / ---- /g'
www.harvard.edu ---- is ---- an ---- alias ---- for ---- pantheon-systems.map.fastly.net.
pantheon-systems.map.fastly.net ---- has ---- address ---- 151.101.130.133
pantheon-systems.map.fastly.net ---- has ---- address ---- 151.101.2.133
pantheon-systems.map.fastly.net ---- has ---- address ---- 151.101.66.133
pantheon-systems.map.fastly.net ---- has ---- address ---- 151.101.194.133
pantheon-systems.map.fastly.net ---- has ---- IPv6 ---- address ---- 2a04:4e42:400::645
pantheon-systems.map.fastly.net ---- has ---- IPv6 ---- address ---- 2a04:4e42:600::645
pantheon-systems.map.fastly.net ---- has ---- IPv6 ---- address ---- 2a04:4e42:200::645
pantheon-systems.map.fastly.net ---- has ---- IPv6 ---- address ---- 2a04:4e42::645
```

- بدون g به صورت زیر میشود درواقع فقط اولین فاصله‌ها از هر خط جایگزین شدند.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ host www.harvard.edu | sed 's/ / ---- /'
www.harvard.edu ---- is an alias for pantheon-systems.map.fastly.net.
pantheon-systems.map.fastly.net ---- has address 151.101.66.133
pantheon-systems.map.fastly.net ---- has address 151.101.130.133
pantheon-systems.map.fastly.net ---- has address 151.101.2.133
pantheon-systems.map.fastly.net ---- has address 151.101.194.133
pantheon-systems.map.fastly.net ---- has IPv6 address 2a04:4e42:200::645
pantheon-systems.map.fastly.net ---- has IPv6 address 2a04:4e42::645
pantheon-systems.map.fastly.net ---- has IPv6 address 2a04:4e42:600::645
pantheon-systems.map.fastly.net ---- has IPv6 address 2a04:4e42:400::645
```

:XARGS

- کاربرد xargs گرفتن خروجی دستورات قبل و هدایتشان به دستورات بعدی که به صورت پیشفرض نمیتوانند خروجی دستور قبل را دریافت کنند.
- همه ابزارها نمیتوانند خروجی دستور قبل را با | دریافت کنند همانطور که میبینید ابزار host به ما این قابلیت را نمیدهد.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ echo www.harvard.edu | host
Usage: host [-aCdilrTvVw] [-c class] [-N ndots] [-t type] [-W time]
            [-R number] [-m flag] [-p port] hostname [server]
       -a is equivalent to -v -t ANY
       -A is like -a but omits RRSIG, NSEC, NSEC3
       -c specifies query class for non-IN data
```

- با استفاده از xargs میتوانیم بدون مشکل کار را پیش ببریم.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ echo www.harvard.edu | xargs host
www.harvard.edu is an alias for pantheon-systems.map.fastly.net.
pantheon-systems.map.fastly.net has address 151.101.2.133
pantheon-systems.map.fastly.net has address 151.101.66.133
pantheon-systems.map.fastly.net has address 151.101.130.133
pantheon-systems.map.fastly.net has address 151.101.194.133
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42::645
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42:200::645
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42:400::645
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42:600::645

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

استفاده بیشتر از یکبار از XARGS:

- با استفاده از xargs و سوییچ I- مشخص کردیم هرجا {} قرار دادیم خروجی دستور قبل به آنجا هدایت شود.

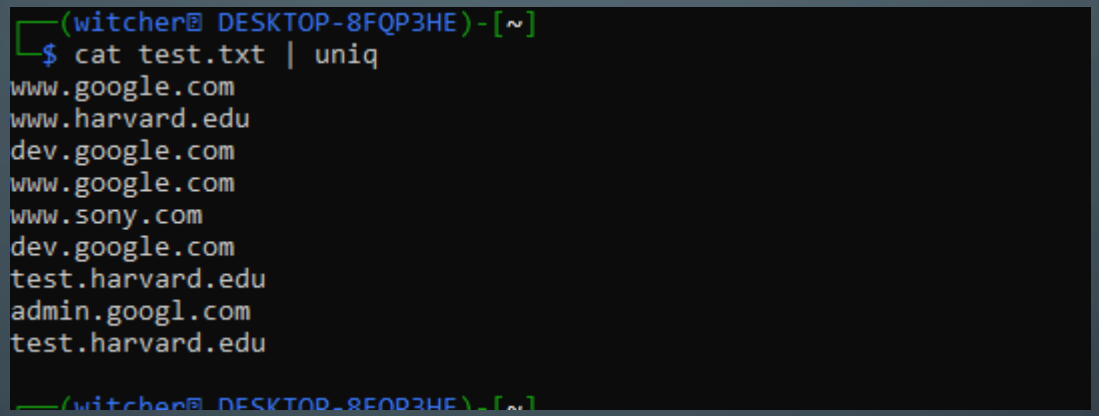
```
└─$ cat subs.txt | xargs -I {} host {}.harvard.edu
www.harvard.edu is an alias for pantheon-systems.map.fastly.net.
pantheon-systems.map.fastly.net has address 151.101.130.133
pantheon-systems.map.fastly.net has address 151.101.66.133
pantheon-systems.map.fastly.net has address 151.101.194.133
pantheon-systems.map.fastly.net has address 151.101.2.133
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42::645
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42:400::645
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42:600::645
pantheon-systems.map.fastly.net has IPv6 address 2a04:4e42:200::645
Host admin.harvard.edu not found: 3(NXDOMAIN)
Host blog.harvard.edu not found: 3(NXDOMAIN)
Host hello.harvard.edu not found: 3(NXDOMAIN)
Host developer.harvard.edu not found: 3(NXDOMAIN)

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

- با استفاده از sh -c میتوانیم شرط یا متغییر و... تعریف کنیم و خروجی دستور قبل را بهشان هدایت کنیم.

```
└─$ cat witcher.sh
domain=$1

xargs -I {} sh -c "if host {}.$domain > /dev/null; then echo {}.$domain; fi"

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ cat subs.txt | bash witcher.sh harvard.edu
www.harvard.edu
test.harvard.edu

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

دستور uniq:

- دستور uniq موارد تکراری را حذف میکند. اما نه موارد تکراری در کل لیست بلکه فقط مقدار فعلی را با مقدار جدید میسنجد و اگر تکراری باشد حذف میکند. به مثال نگاه کنید:



- در تصویر uniq تمام موارد تکراری حذف نکرد. چون سنجش به این شکل بود خط یک با خط دوم برابر است؟ خیر. پس هیچ کدام حذف نمیکند حالا دوباره آیا خط دوم با خط سوم برابر است؟ خیر. پس حذف نکرد و به همین شکل تا خط آخر ادامه داد.

دستور sort:

- با استفاده از دستور sort میتوانیم لیستمان را مرتب کنیم، موارد اضافی یک لیست را حذف کنیم، تعداد تکرار هر خط در لیست را بفهمیم و...

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ cat test.txt | sort
admin.googl.com
dev.google.com
dev.google.com
test.harvard.edu
test.harvard.edu
www.google.com
www.google.com
www.harvard.edu
www.sony.com
```

- برای نمایش داده نشدن موارد تکراری کافیست از سوییچ **u-** استفاده کنید

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ cat test.txt | sort -u
admin.googl.com
dev.google.com
test.harvard.edu
www.google.com
www.harvard.edu
www.sony.com

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

حذف تمام موارد تکراری یک لیست با استفاده از uniq و sort:

- برای حذف تمام موارد تکراری با uniq باید ابتدا با sort مرتب کنیم . برای مثال خط دوم با خط سوم سنجیده شد و uniq متوجه شد این دو باهم برابر هستند پس مورد تکراری نمایش داده نشد.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ cat test.txt | sort | uniq
admin.googl.com
dev.google.com
test.harvard.edu
www.google.com
www.harvard.edu
www.sony.com

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

- با سوییچ **u-** میتواینم مشاهده کنیم کدوم خطوط اصلا تکرار نشده اند برای مثال وقتی سابدامنه جدید پیدا کنید اگر با بقیه موارد تکرار نداشته باشد میفهمید یک سابدامنه جدید است.

- با سوییچ **c-** میتوانیم تعداد تکرار هر مورد از لیست را مشاهده کنیم.

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ cat test.txt | sort | uniq -u
admin.googl.com
www.harvard.edu
www.sony.com

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```

```
┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$ cat test.txt | sort | uniq -c | sort
      1 admin.googl.com
      1 www.harvard.edu
      1 www.sony.com
      2 dev.google.com
      2 test.harvard.edu
      2 www.google.com

┌──(witcher㉿ DESKTOP-8FQP3HE)-[~]
└─$
```