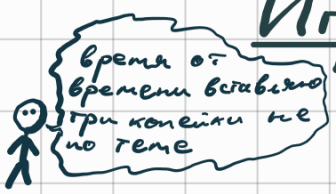


Игры про основы работы с командной строкой!

overthewire.org

пример подключения к уровню:

```
ssh bandit0@bandit.labs.overthewire.org -p 2220
```

username — hostname — port num

Dashed filename (или первый затык)

О чём речь: файлы с тире как первой символ имени

пример: `-filename.jpg`

В чём проблема? Команды используют тире для указания опций и аргументов.

Поэтому при обращении к файлу с именем, начинающимся с тире, следует использовать абс. путь.

пример: `cp ./-filename.jpg`

Spaces in filename

Обычно файлы в линукс не имеют пробелов, потому что синтаксис команд linux предполагает разделение аргументов в команде пробелами:

```
command [options] argument1 argument2
```

Если имя файла содержит пробелы, слова будут восприняты как отдельные аргументы

Решение:

имя файла в кавычки:

```
"file name withn spaces"
```

или отделяет каждый пробел обратной косой чертой:

```
file\ name\ with\ spaces
```

Hidden files

Файлы в линукс, начинающиеся с точки,

считаются спрятанными (системные файлы и можно создать самому).

пример имени: .hidden.txt

Их не показывает ls.

Решение:

ls -a

(от слова all)

Human readable files in Linux.

Файлы, которые читаемы человеком (т.е. содержат печатаемые символы в какой-либо из кодировок).

Как тут команда file?

file filename - определяет тип файла filename (ещё есть много опций, см. man)

напечатанный тип содержит одно из ключевых слов:

- text: файл содержит только печатные символы и немного контрольных

- executable: файл содержит рез-т компиляции программы
- data: что-то другое (обычно что-то бинарное и непонятное)

How to: find

Мощная команда с кучей возможностей и огромным man'ом.

minimum:

```
find dir -name filename
```

найти файл с именем filename, начиная с директории dir

`-size n[cwbkMG]`

- bytes
- two-byte words
- 512-byte blocks
- kibibytes
- mebibytes
- gibibytes

`+` - greater than

`-` - less than

Пример: `find / -size -1024c`

ищет файлы от 0 до 1023 байт (округление вверх)

-user имя

Файлом владеет пользователь с именем имя

How to: grep

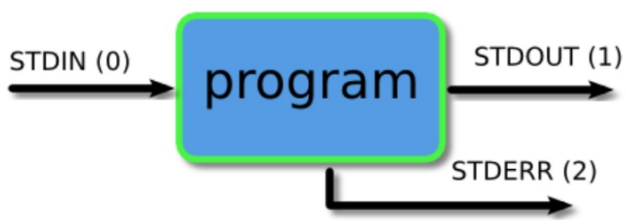
grep [OPTION] Patterns [FILE]

Утилита печатает строки, которые соответствуют заданному шаблону

Пример: grep "meow" -f text.txt - выведет все строки со словом meow в файле text.txt

Piping and redirecting

- STDIN (0) - Standard input (data fed into the program)
- STDOUT (1) - Standard output (data printed by the program, defaults to the terminal)
- STDERR (2) - Standard error (for error messages, also defaults to the terminal)

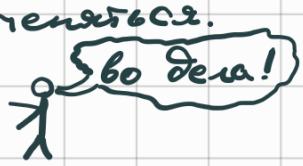


Redirection (пересылка данных в файлы и из них)

- Оператор "больше" перенаправляет поток с stdout в указанный файл (и сохраняет этот файл) вместо печати на экран

```
Terminal
1.  user@bash: ls
2.  barry.txt bob example.png firstfile foo1 video.mpeg
3.  user@bash: ls > myoutput
4.  user@bash: ls
5.  barry.txt bob example.png firstfile foo1 myoutput video.mpeg
6.  user@bash: cat myoutput
7.  barry.txt
8.  bob
9.  example.png
10. firstfile
11. foo1
12. myoutput
13. video.mpeg
14. user@bash:
```

форматирование данных после меняется.



Если файл ~~7~~, то будет создан.
Если файл 7, то будет перезаписан!

- Оператор >> дописывает перенаправляемую инф-ю в конец файла

```
Terminal
1.  user@bash: cat myoutput
2.  7 barry.txt
3.  user@bash: ls >> myoutput
4.  user@bash: cat myoutput
5.  7 barry.txt
6.  barry.txt
7.  bob
8.  example.png
9.  firstfile
10. foo1
11. myoutput
12. video.mpeg
13. user@bash:
```

- Оператор "меньше" (<) позволяет посылать данные из файла в утилиту

```
Terminal
1.  user@bash: wc -l < barry.txt > myoutput
2.  user@bash: cat myoutput
3.  7
4.  user@bash:
```

выполняется справа - налево

`wc file` - выводит кол-во слов, строк и байт в файле

`-c` - размер объекта в байтах

`-m` - кол-во символов в объекте

`-l` - кол-во строк в объекте

`-w` - кол-во слов

- Идентификация потока осуществляется вставлением его номера перед оператором

```
Terminal
1. user@bash: ls -l video.mpg blah.foo
2. ls: cannot access blah.foo: No such file or directory
3. -rwxr--r-- 1 ryan users 6 May 16 09:14 video.mpg
4. user@bash: ls -l video.mpg blah.foo 2> errors.txt
5. -rwxr--r-- 1 ryan users 6 May 16 09:14 video.mpg
6. user@bash: cat errors.txt
7. ls: cannot access blah.foo: No such file or directory
8. user@bash:
```

← пересылает stderr в файл

- Чтобы указать, что мы хотим переслать инф-ю в какой-то поток, пишем & и номер потока

```
Terminal
1. user@bash: ls -l video.mpg blah.foo > myoutput 2>&1
2. user@bash: cat myoutput
3. ls: cannot access blah.foo: No such file or directory
4. -rwxr--r-- 1 ryan users 6 May 16 09:14 video.mpg
5. user@bash:
```

(без & было бы перенаправление в файл с именем 1)

Piping (пересылка данных из одной программы в другую)

- Оператор палка "|" берёт вывод программы слева и отдаёт его на вход программы справа.

```
Terminal
1. user@bash: ls
2. barry.txt bob example.png firstfile foo1 myoutput video.mpeg
3. user@bash: ls | head -3
4. barry.txt
5. bob
6. example.png
7. user@bash:
```

```
Terminal
1. user@bash: ls | head -3 | tail -1
2. example.png
3. user@bash:
```

совет:

запускать программы последовательно и убеждаться, что вывод соответствует ожидаемому, а затем составлять pipe (уменьшит баги дебага)

How to: uniq

выводит или пропускает повторяющиеся строки.

Удаляет все кроме одной повторяющиеся строки из ВВОДА (или стандартного ввода) и печатает на ВЫВОД (или стандартный вывод).

без аргументов

-c, --count
выводить число повторов в начале каждой строки

-d, --repeated
выводить только повторяющиеся строки

-D, --all-repeated[*=delimit-method*]
печатать все повторяющиеся строки delimit-method={none(по умолчанию),prepend,separate} Разделение делается по пустым строкам.

-f, --skip-fields=*N*
не сравнивать первые N полей

-i, --ignore-case
игнорировать регистр при сравнении

-s, --skip-chars=*N*
не сравнивать первые N знаков

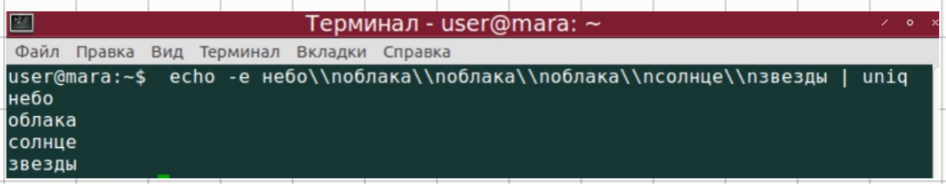
-u, --unique
выводить только неповторяющиеся строки

-w, --check-chars=*N*
сравнивать только первые N символов в строке

Замечание: uniq не считает повторяющиеся строки одинаковыми, если они не прилегают друг к другу. Сначала нужно отсортировать

```
$ uniq опции [файл_источник [файл_для_записи]]
```

пример



How to: sort

Сортирует строки текстовых файлов

```
sort [параметр] [Файл(ы)]
```

-b, --ignore-leading-blanks
игнорировать пробелы в начале сортируемых полей или начале ключей

-d, --dictionary-order
воспринимать в составе ключей лишь буквы (латинского алфавита), цифры и пробелы, игнорируя все прочие символы

-f, --ignore-case
во время сортировки преобразует строчные (маленькие) в соответствующие прописные (большие) буквы, т.е. выполняется сортировка нечувствительная к регистру символов

-g, --general-numeric-sort
выполнять сравнение в соответствии с общим числовым значением (используют совместно с параметром -b). Это численная сортировка, при которой дополнительно распознаётся экспоненциальное представление чисел (например, 9.1019e7)

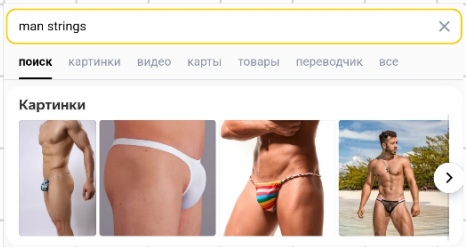
-i, --ignore-nonprinting
в ключах рассматриваются только печатаемые (ASCII) символы, а остальные игнорируются

-M, --month-sort
выполнять сравнение по трёх-символьным сокращениям англоязычных названий месяцев, т.е. (незнакомое) < `JAN' < ... < `DEC'

-n, --numeric-sort
числовая сортировка, т.е. сравнение ведётся по числовому значению (используют совместно с параметром -b)

-r, --reverse
сортировка выполняется в обратном порядке (по убыванию)

How to : strings



Находит печатаемые (printable) строки в файле и выводит в stdout.

strings [-a][-t *format*][**-n** *number*][*file*...]

-a
Scan files in their entirety. If **-a** is not specified, it is implementation-defined what portion of each file is scanned for strings.

-n *number*
Specify the minimum string length, where the *number* argument is a positive decimal integer. The default shall be 4.

-t *format*
Write each string preceded by its byte offset from the start of the file. The format shall be dependent on the single character used as the *format* option-argument:

d
The offset shall be written in decimal.

o
The offset shall be written in octal.

x
The offset shall be written in hexadecimal.

Base64 encoding

Группа алгоритмов шифрования "бинарный код в текст". Схема (очень примитивная) на вход

получается бинарные данные, а выдаёт printable символы, ограниченные набором в 64 уникальных символа. Принцип: 6 бит исходных данных ↓ один из 64 уникальных символов

Таблицы соответствия есть в интернете, но как правило символы есть набор {A-Z, a-z, 0-9} и ещё 2 значения (кто на что горазд, напр, в RFC 4648 это '+' и '/')

Encoding of the source string ⟨Man⟩ in Base64

Source ASCII text	Character	M							a							n							
	Octets	77 (0x4d)							97 (0x61)							110 (0x6e)							
Bits		0	1	0	0	1	1	0	1	0	1	1	0	0	0	0	1	0	1	1	0	1	1	1	0
Base64 encoded	Sextets	19							22					5				46							
	Character	T							W					F				u							
	Octets	84 (0x54)							87 (0x57)					70 (0x46)				117 (0x75)							

How to: base64

Base64 encode or decode FILE, or standart input, to standart output

base64 [OPTION] [FILE]

-w, --wrap=*COLS*
Wrap encoded lines after COLS character (default 76). Use 0 to disable line wrapping.

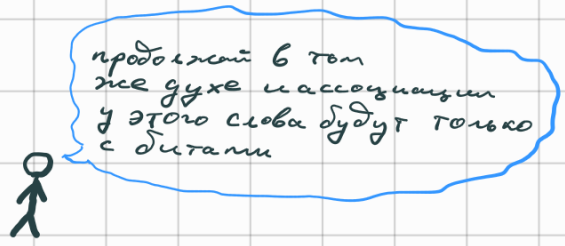
-d, --decode
Decode data.

-i, --ignore-garbage
When decoding, ignore non-alphabet characters.

Padding и рассуждения по этому поводу.

Т.к. Base64 шестибитовый алгоритм шифрования (3×8=24 бита)

то 3 октета незашифрованного текста или иных данных → 4 символа Base64-зашифрованного текста (4 секстета) 4×6=24 бита



до после

ещё раз: 3 байта → 4 байта

Если размер входных данных (в байтах) не кратен 3, на выходе зашифрованные данные должны иметь набивку (padding), чтобы их длина была кратна 4 (символ набивки - это '=')

примеры

Source ASCII text	Character	M							a															
	Octets	77 (0x4d)							97 (0x61)															
Bits		0	1	0	0	1	1	0	1	0	1	1	0	0	0	0	1	0	0						
Base64 encoded	Sextets	19							22						4			Padding						
	Character	T							W						E			=						
	Octets	84 (0x54)							87 (0x57)						69 (0x45)			61 (0x3D)						

Source ASCII text	Character	M																						
	Octets	77 (0x4d)																						
Bits		0	1	0	0	1	1	0	1	0	0	0	0											
Base64 encoded	Sextets	19							16						Padding			Padding						
	Character	T							Q						=			=						
	Octets	84 (0x54)							81 (0x51)						61 (0x3D)			61 (0x3D)						

Input		Output		Padding
Text	Length	Text	Length	
light work.	11	bGlnaHQgd29yay4=	16	1
light work	10	bGlnaHQgd29yaw==	16	2
light wor	9	bGlnaHQgd29y	12	0
light wo	8	bGlnaHQgd28=	12	1
light w	7	bGlnaHQgdw==	12	2

Base64 можно детектить по = в конце текста и его размеру (кратен 4)

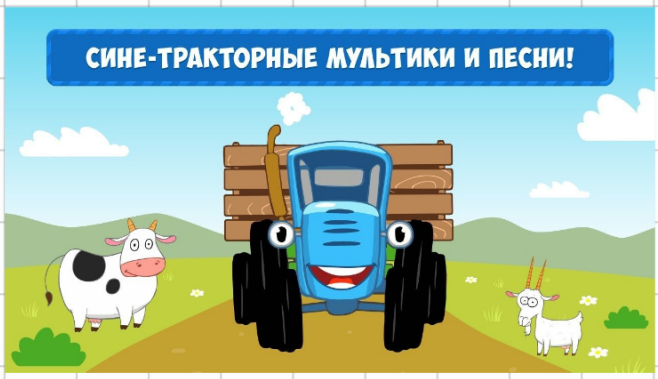
ROT 13 encryption

(rotate by 13 places)

В латинице 26 букв. Алгоритм сдвигает каждую букву латиницы по циклу на 13 мест (символы не из алфавита остаются нетронутыми). Повторное применение алгоритма совершает декодирование.

How to: tr

tr - выполняет символьное преобразование путём подстановки или удаления символов.



tr [ОПЦИЯ] строка_1 [строка_2]

-c, --complement
замещает первый набор символов СТРОКИ_1 его дополнением (всеми символами, отсутствующими в СТРОКЕ_1)

-d, --delete
удаляет все символы, которые перечислены в наборе СТРОКА_1 без преобразования

-s, --squeeze-repeats
заменяет последовательность повторяющихся символов в наборе СТРОКА_1 на один такой символ (т.е. удаляет все повторяющиеся символы, кроме первого)

-t, --truncate-set1
ограничивает (делает обрезание) набор СТРОКА_1, если он длиннее набора СТРОКА_2

Выполняет преобразование, подстановку (замену), сокращение и/или удаление символов, поступающих со стандартного ввода, записывая результат на стандартное устройство вывода. Она часто применяется для удаления управляющих символов из файла или преобразования регистра символов. Как правило, команде tr передаются две строки (набора) символов: первый набор СТРОКА_1 содержит искомые символы, а второй СТРОКА_2 - те, на которые их следует заменить. При запуске команды устанавливается соответствие между символами обоих наборов, а затем начинается преобразование.

Пример.

```
echo "fooman@example.com" | tr 'A-Za-z'
                           'N-ZA-Mn-za-m'
```

How to: xxd

xxd – создаёт представление файла в виде шестнадцатеричных кодов или выполняет обратное преобразование.

bin to hex

xxd [ключи] [входной файл [выходной]]

xxd -r [ключи] [входной файл [выходной]]

hex to bin

ключи:

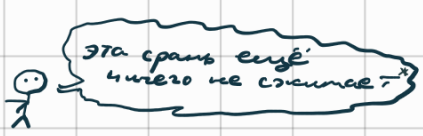
-b – вместо шестнадцатеричного кода показывает двоичный

-c num – задает кол-во октетов, к-е выводится на строке

-S seek – начинает с указанного смещения в байтах

-a – скипает нулевые строки, заменяя *.

How to : tar



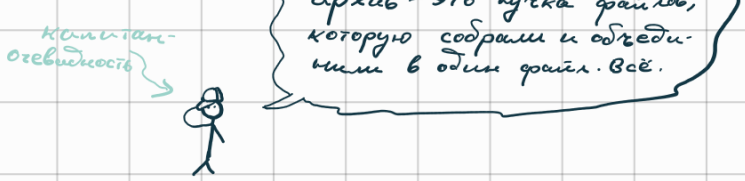
Если не указать флаг сжатия каким-либо форматом

Создаёт архив, добавляет или удаляет из него файлы.

архивирует:

```
tar опции архив.tar файлы_для_архивации
```

разархивирует:




```
tar опции архив.tar
```

Опции:

Функция	Длинный формат	Описание
-A	--concatenate	Присоединить существующий архив к другому
-c	--create	Создать новый архив

← один из этих должен быть обязательно

-d	--diff --delete	Проверить различие между архивами Удалить из существующего архива файл
-r	--append	Присоединить файлы к концу архива
-t	--list	Сформировать список содержимого архива
-u	--update	Обновить архив более новыми файлами с тем же именем
-x	--extract	Извлечь файлы из архива

Параметр	Длиннный формат	Описание
-C *dir*	--directory=DIR	Сменить директорию перед выполнением операции на *dir*
-f *file*	--file	Вывести результат в файл (или на устройство) *file*
-j	--bzip2	Перенаправить вывод в команду *bzip2*
-p	--same-permissions	Сохранить все права доступа к файлу
-v	--verbose	Выводить подробную информацию процесса
	--totals	Выводить итоговую информацию завершенного процесса
-z	--gzip	Перенаправить вывод в команду *gzip*

← необязательные

Примеры

```
$ tar --totals -cvf archive.tar file1 file2 file3
```

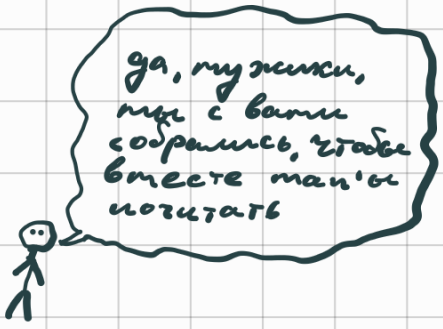
← создать архив с именем archive.tar из файлов file{1-3}

```
$ tar -tf archive.tar
```

← посмотреть содержимое архива archive.tar

```
$ tar -cjvf archive.tar.bz2 file1 file2 file3
```

← создать архив, а затем сжать его в формате bz2



Сигнатура: ustar на 257 байте

What is: gzip

Алгоритм сжатия без потерь, к-й является комбинацией LZ77 и кодирования Хаффмана (это называется DEFLATE алгоритм)

Формат файла .gzip:

- 10 байт заголовка, содержащего магическое число 1f 8b, метод сжатия (08 для DEFLATE), 1 байт флагов, 4 байта timestamp, флаги сжатия, ID операционной сис-мы.
- доп-ых не так важно
- payload
- CRC-32

How to: gzip

gzip опции файл

опции:

-c	выводить архив в stdout
-d	распаковать
-l	показать инф-ю об архиве
-0	мин. уровень сжатия

-9 max уровень сжатия

пример:

```
$ gzip -c файл > архив.gz
```

What is: bzip2

Сжатие алгоритмом Барроуза-Уилера (сжимает лучше, чем gzip, но медленнее)

расширение: .bzz

сигнатура: Bzh

How to: bzip2

Аналогично gzip

пример

```
$ bzip2 file
```

создаст сжатый файл file.bzz

где смотреть в xxd?	Пояснение
в колонке посредине	магическое число - это hex представление характерных чисел данного формата бинарника
в колонке справа	сигнатура - это ASCII расшифровка характерных чисел данного формата бинарника