Crunch: A wordlist Generator

* Objectives
* Basic Structure
* Pattern Specific Wordlist
* Permutation
* Break wordlist into chunks
* Wordlist Compression
* Handle frequency of characters
* Inversion of output

**Crunch:**

Crunch is a wordlist that generates a utility used to create a wordlist using numbers, letters, and symbols. Mostly Hackers, Penetration testers use this tool to create passwords. The crunch comes pre-installed In Kali Linux.

Basic Structure

Crunch <min><max> -o <filename>

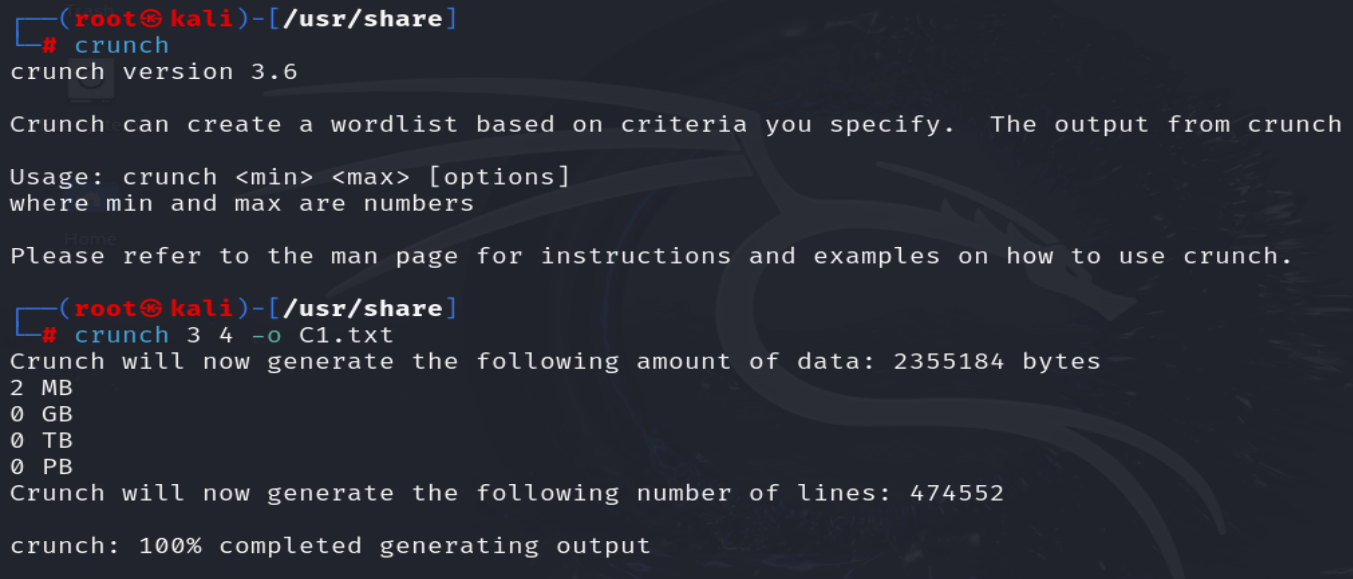
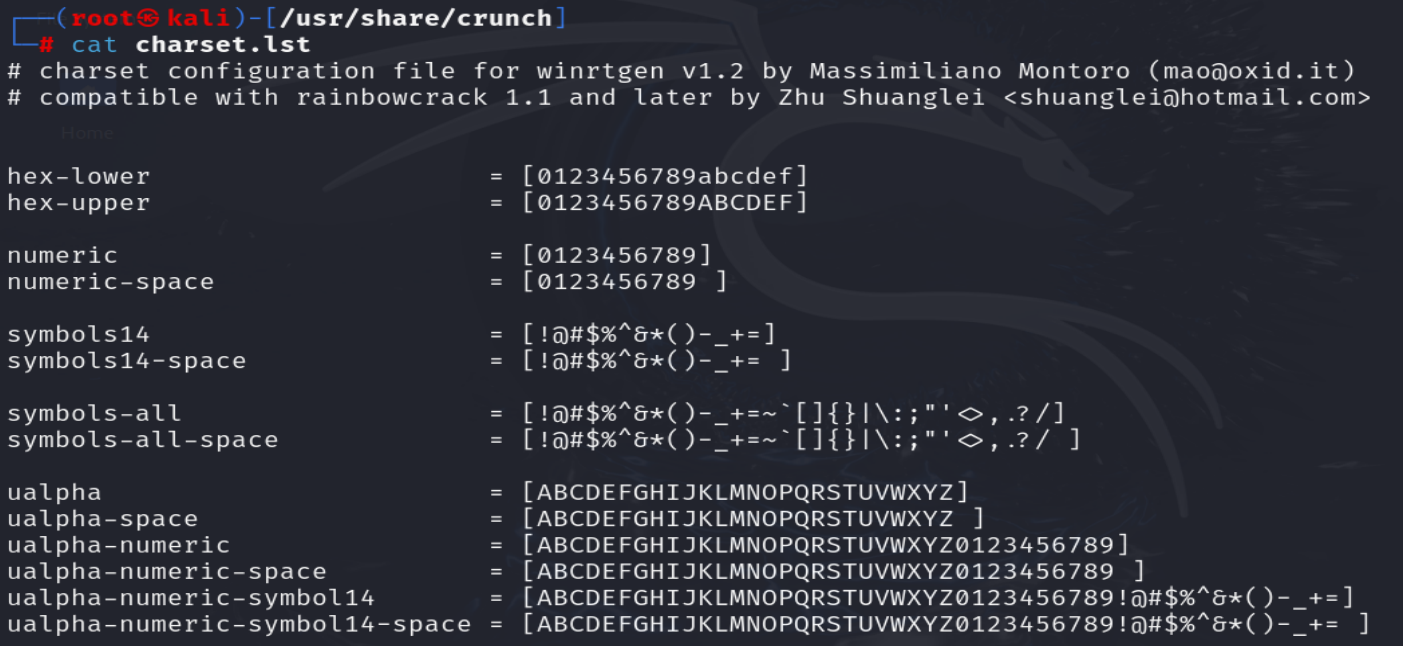


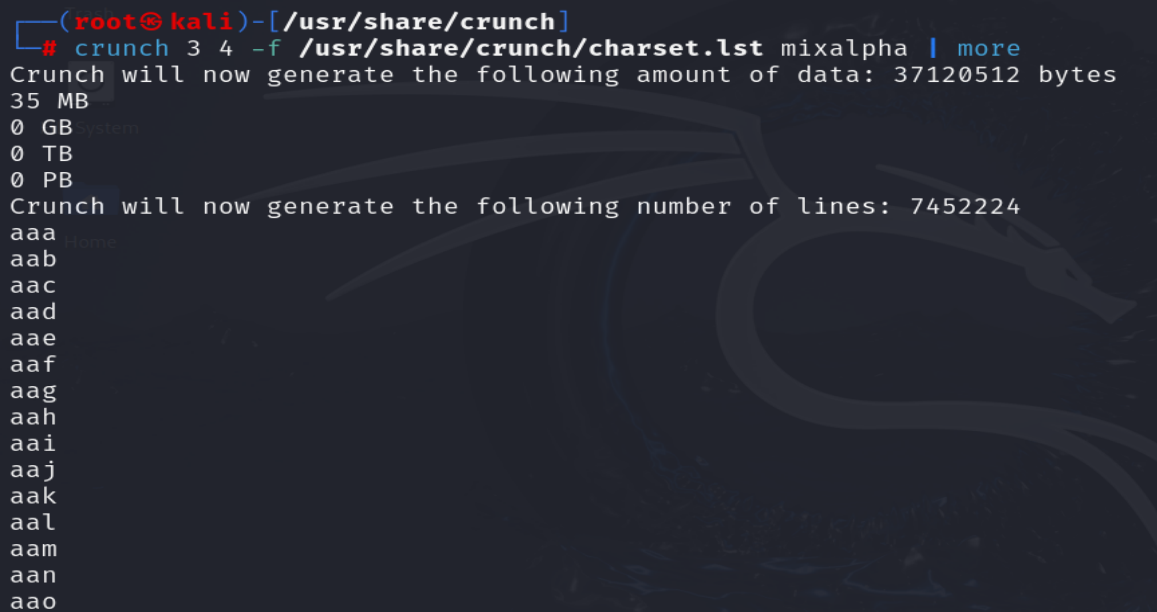
Fig1: Basic structure

* min: It is the minimum password length.
* max: It is the maximum password length.
* charset: Character se to be used.
* -o : Output in a textfile, along with name of the text file.

Charset: In charset we have three options

* Default: Uses small alphabet (Fig1).
* User-defined: User defined which charset wants to use.(abc.. , ABC… ,123…, @#$%^\*)
* Crunch charset: It has its own charset. If we want to use charset of crunch we have to change directory (/usr/share/crunch) and we fined charset.lst in it.



**Crunch charset:** Crunch <min><max> -f /usr/share/charset.list mixalpha | more

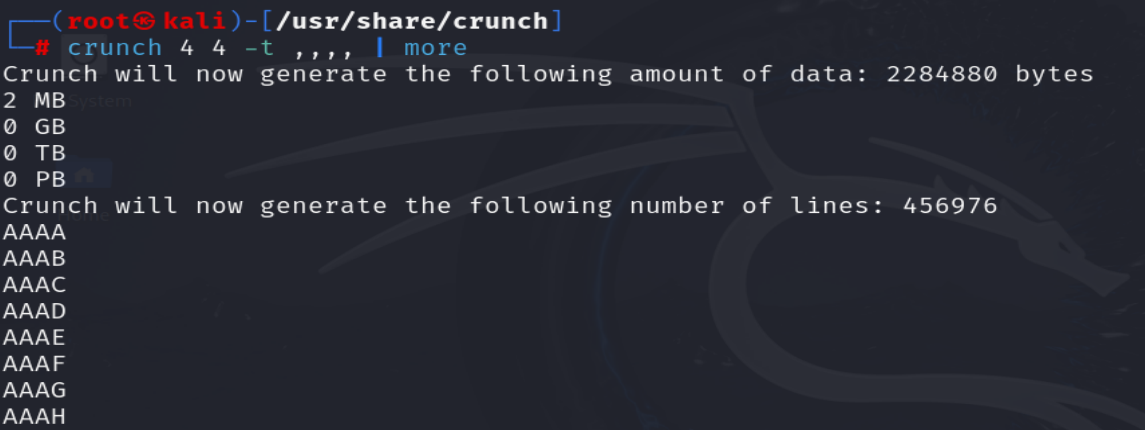
**Pattern specific wordlist**

**-t:** indicatespattern

Scenario 1: Normal condition

* @ - Lowercase
* , - Uppercase
* %- Number
* ^- Symbol

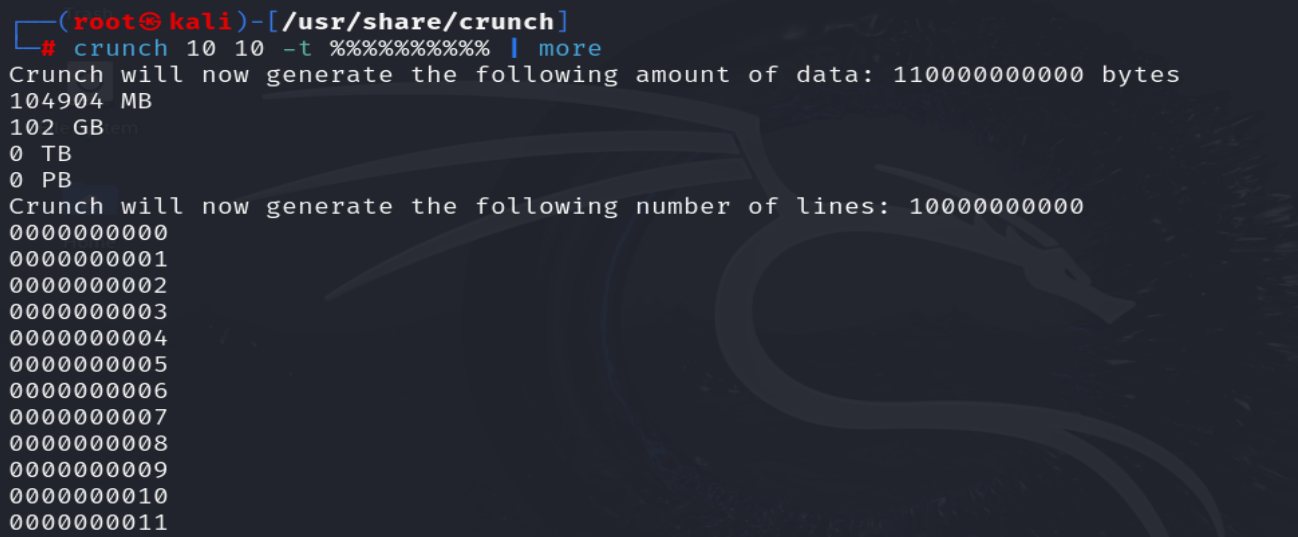
Syntax: Crunch <min><max> path -t //%%(using two upper case letter and 2 Number) below we used only upper case.



Scenario 2: If password is a phone number than

* @ - Lowercase
* , - Uppercase
* %- Number
* ^- Symbol

Syntax: Crunch 10 10 path -t %%%%%%%%%%%(we have to use ten % sign)

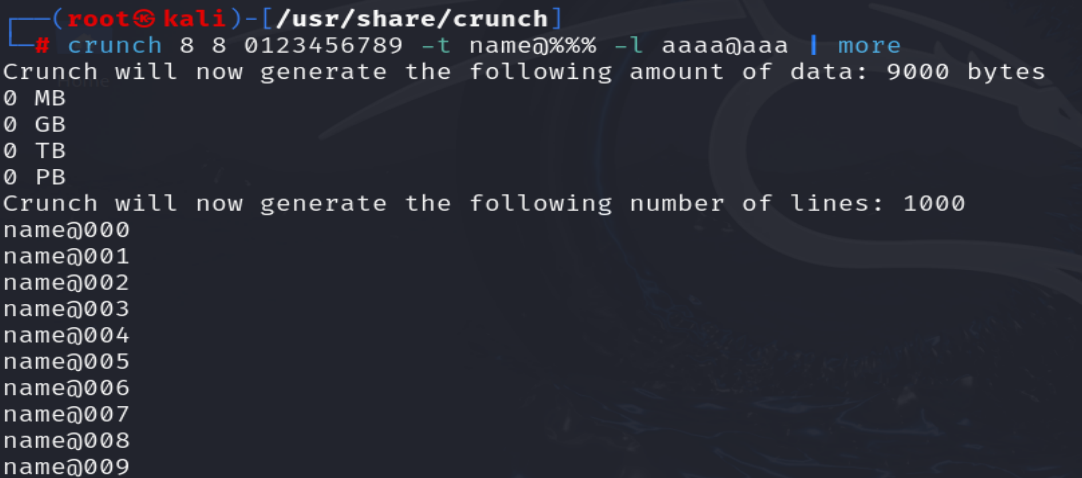


Scenario 3: if password length is 7or8 char including name,@ and number

* @ - Lowercase
* , - Uppercase
* %- Number
* ^- Symbol

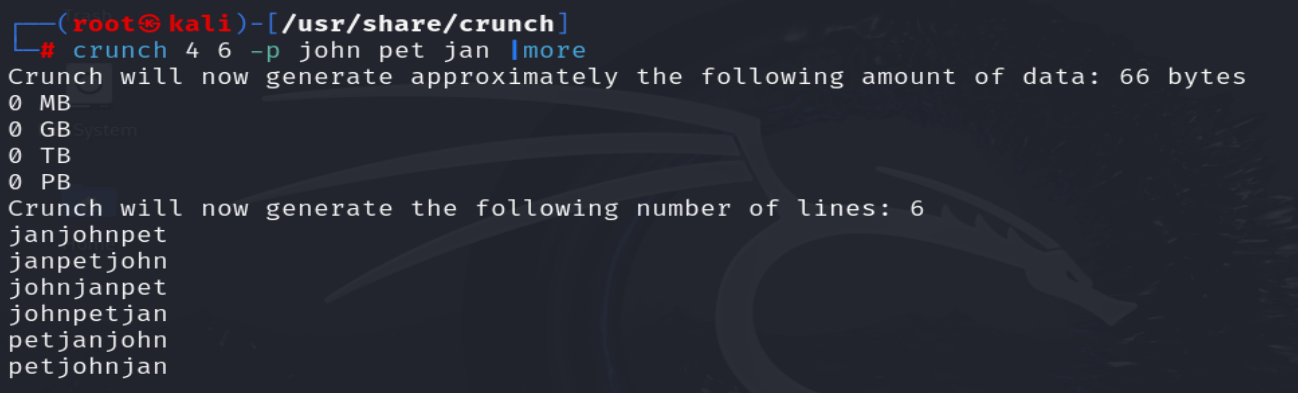
Syntax: Crunch 7 8 charset -t name@%%% -l aaaa@aaa |more

-l is used so that **@** used as it is.

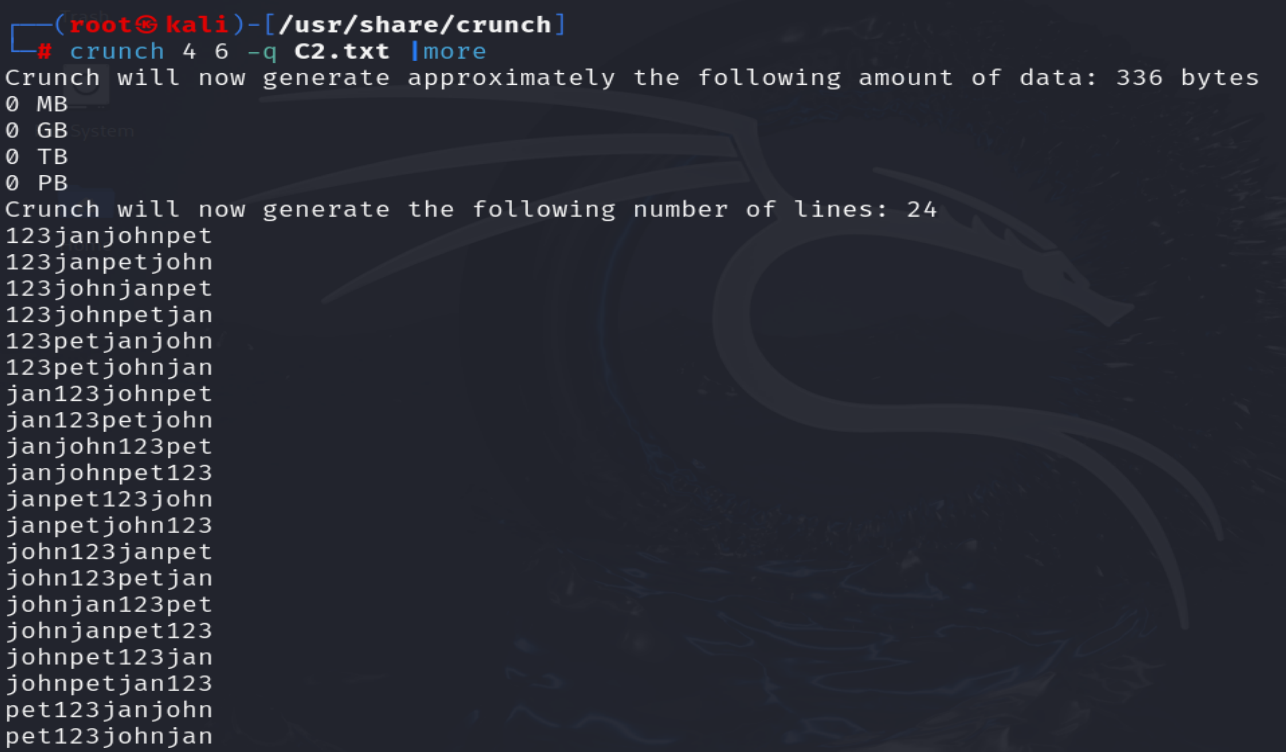


**Permutation:**

In permutation it used only john, pet and jan to create password



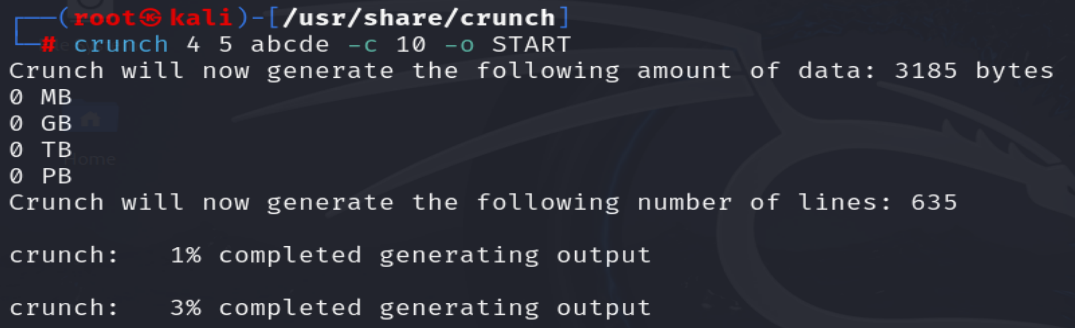
Now saving john, pet, jan and 123 in .txt file.

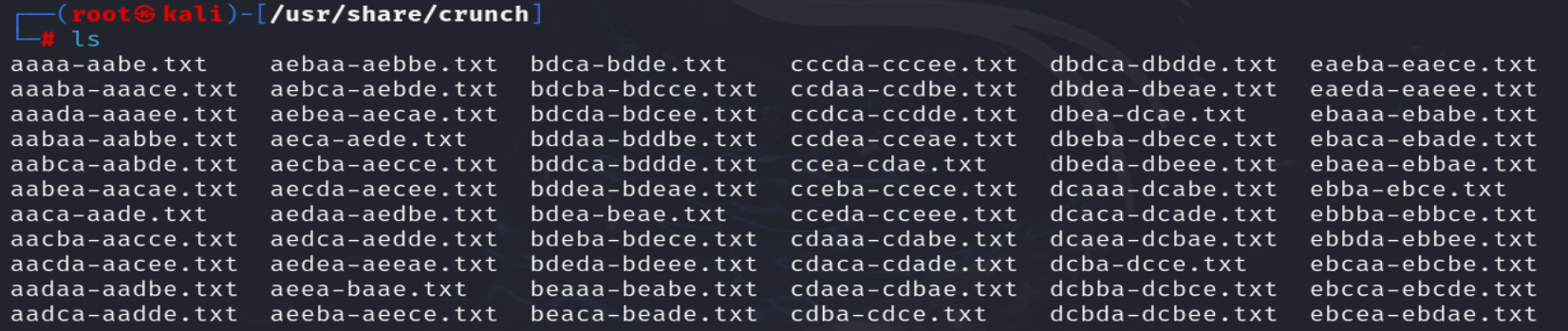


**Break Wordlist into chunks:**

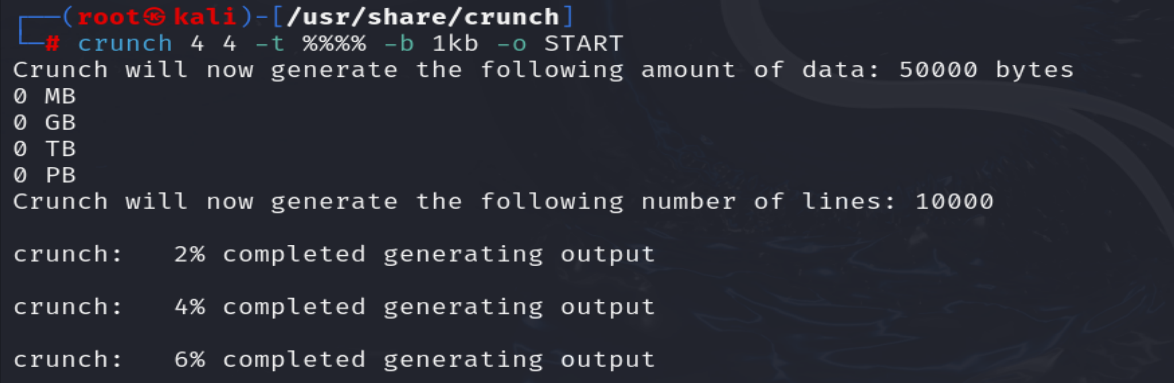
It breaks wordlist into two:

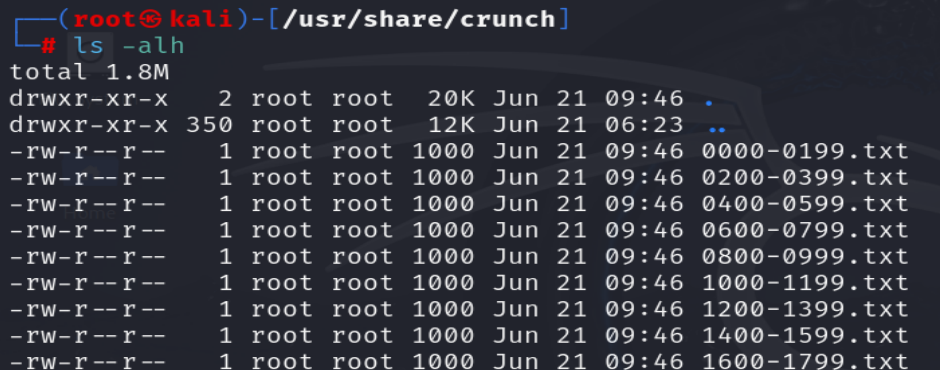
* + - * Lines : After limit it creates new file(e.g. if we use 10 as limit than after 10 lines new file creates and this continues till all password over).





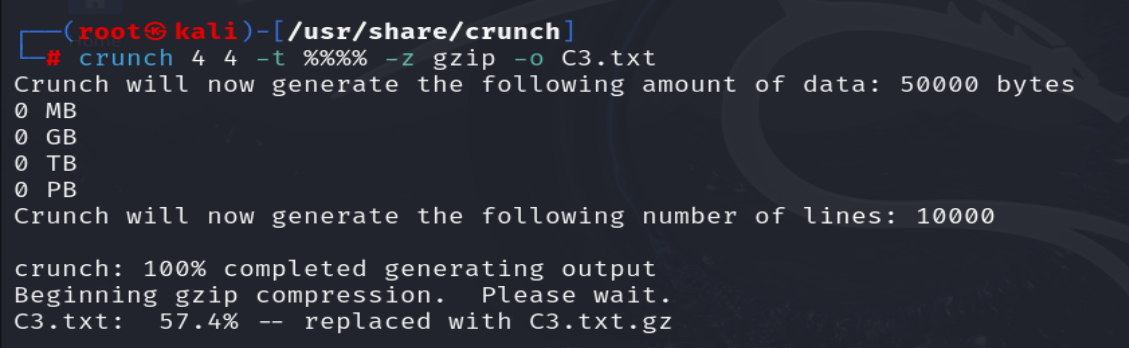
* + - * Storage: After limit it creates new file (e.g. if we use 2MB as limit of a file than after 2MB it creates new file).





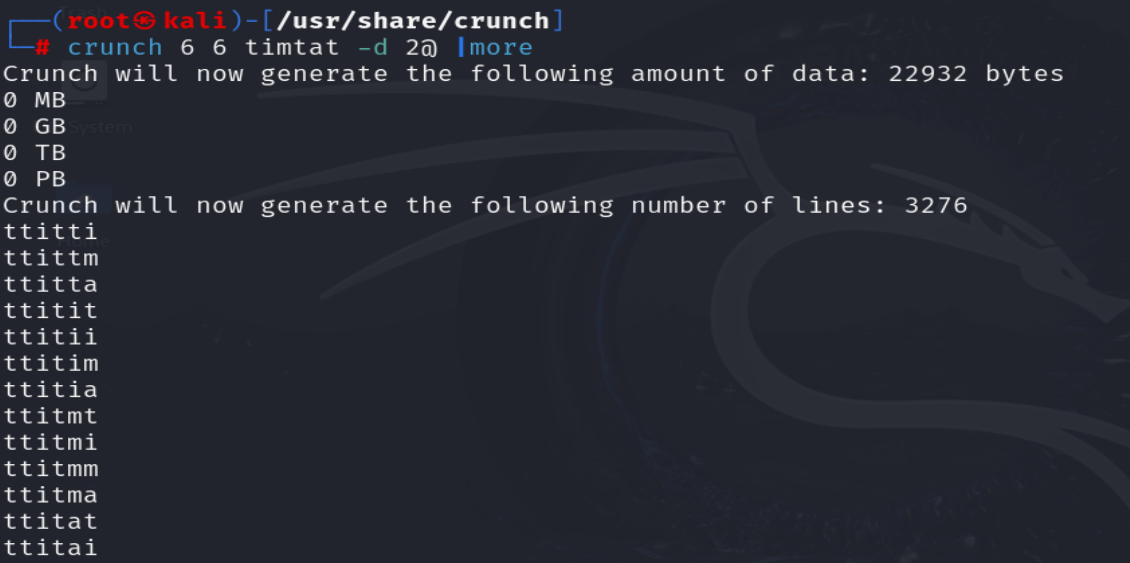
Wordlist Compression:

For compression we have .gzip .bzip2 .lzma .7z etc.



Handle frequency of characters:

Syntax: crunch 6 6 timtat -d 2@ |more



Inversion of output:

Syntax: crunch 8 8 -t pass@%%% -l aaaa@aaa -I |more

