PROBLEM STATEMENT

The most common form of authentication is the combination of a username and a password or passphrase. If both match values stored within a locally stored table, the user is authenticated for a connection. Password strength is a measure of the difficulty involved in guessing or breaking the password through cryptographic techniques or library-based automated testing of alternate values.

Work with an Emulator that uses information about the user to create a password list that can be used against him.

SOLUTION

A weak password might be very short or only use alphanumeric characters, making decryption simple. A weak password can also be one that is easily guessed by someone profiling the user, such as a birthday, nickname, address, name of a pet or relative, or a common word such as God, love, money or password.

TECHNOLOGY USED

CUPP:

Cupp stands for **Common User Passwords Profiler** and this tool can be used in many circumstances like license penetration tests or forensic crime investigations, CUPP is a cross-platform and written in Python and it's functioning is simple but with very powerful results. This application is a social engineer's best friend when it comes to creating targeted password dictionaries which are tailored to an individual.

Cupp takes vectors from the profiling done for an individual, such as their nickname, pets name, child's birthdate, etc. It works on the principle that a password is, more often, a combination of things known to an individual. These known things are often personal details that are very close to a person's heart.

In cases when a person might use special notations in place of alphabets (e.g.: leet can be written as 133t) Cupp has you covered.

Aircrack-ng:

Aircrack-ng is a complete suite of tools to assess WiFi network security.

It focuses on different areas of WiFi security:

- Monitoring: Packet capture and export of data to text files for further processing by third party tools
- Attacking: Replay attacks, deauthentication, fake access points and others via packet injection
- Testing: Checking WiFi cards and driver capabilities (capture and injection)
- Cracking: WEP and WPA PSK (WPA 1 and 2)

All tools are command line which allows for heavy scripting. A lot of GUIs have taken advantage of this feature. It works primarily on Linux but also Windows, macOS, FreeBSD, OpenBSD, NetBSD, as well as Solaris and even eComStation 2.

Which can be used along with CUPP to crack the victims password with the help of personalized dictionary of generated passwords and usernames.

WORKING PROCESS

Optional Arguments (CUPP):

-i Interactive questions for user password profiling

-w FILENAME Use this option to profile an existing dictionary,

-l Download huge wordlists from a repository

-a Parse default usernames and passwords directly from Alecto DB.

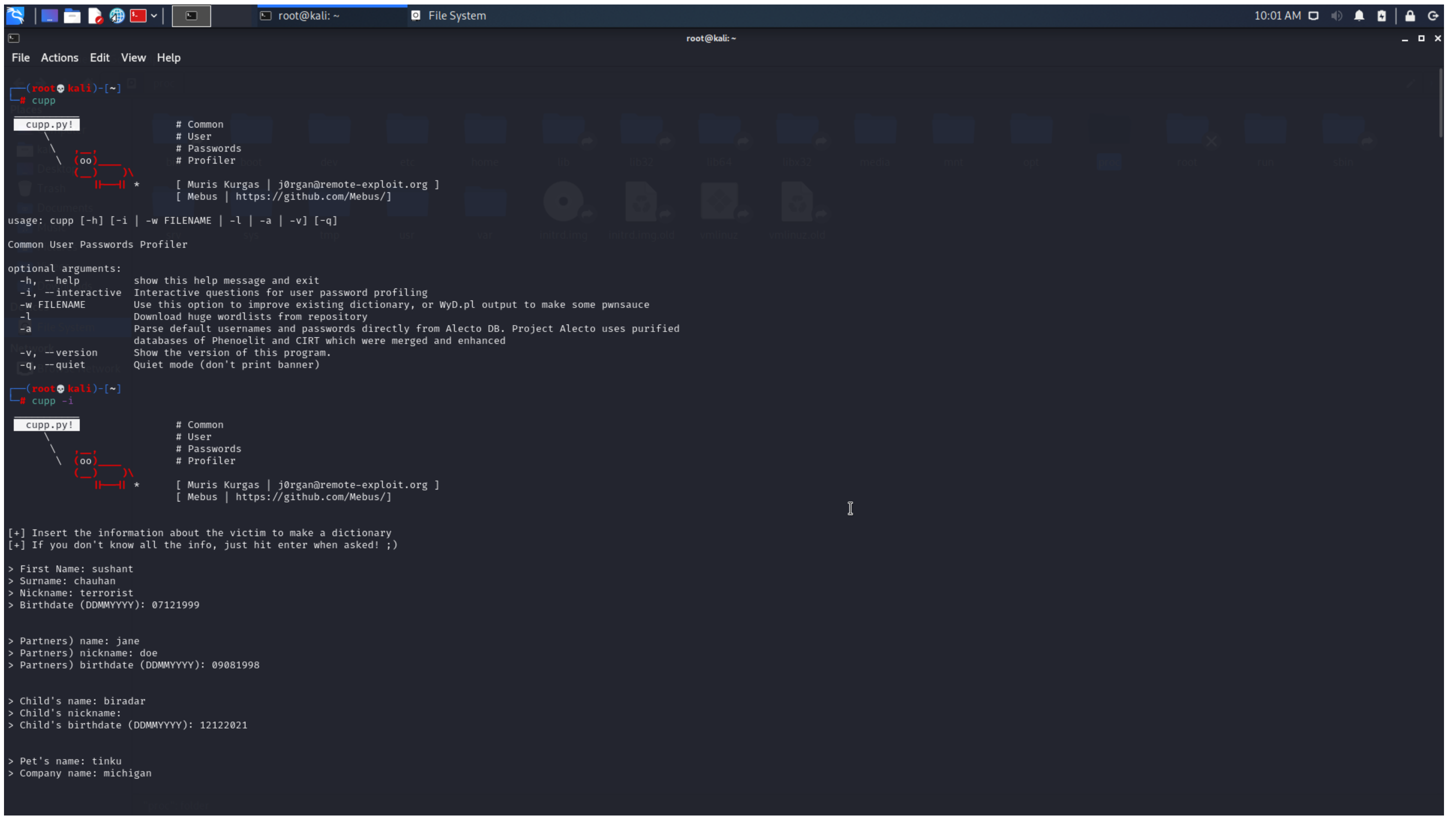
Project Alecto uses purified databases of Phenoelit and CIRT which merged and enhanced.

-v Version of the program

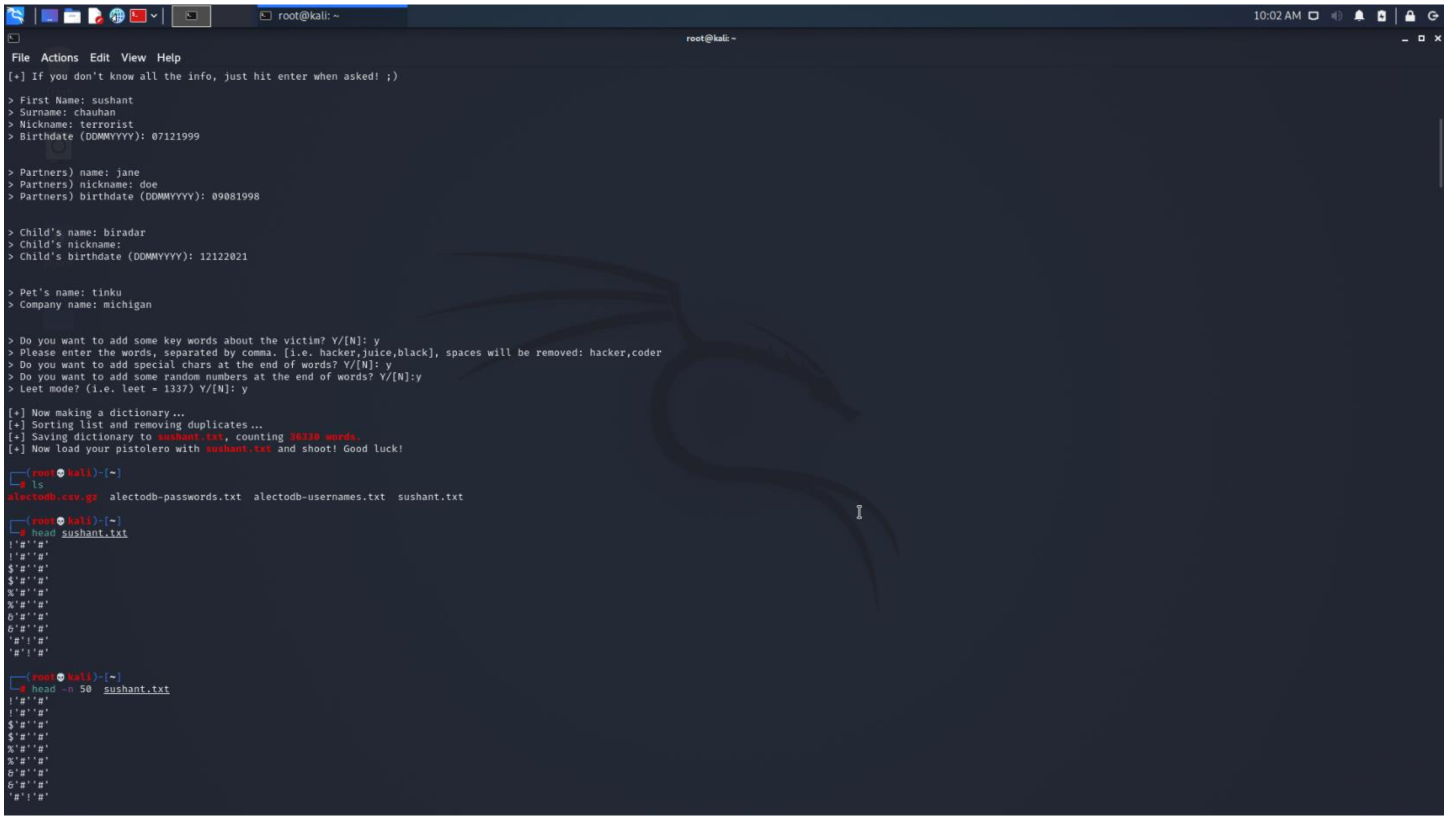
Generating Custom Dictionary:

We will be using the interactive option to generate the custom dictionary. You will see that we have the option to input options such **as pet's name, child's name, partners nickname,** etc. All these things are highly personal and very common to find these things in a password, one way or another.

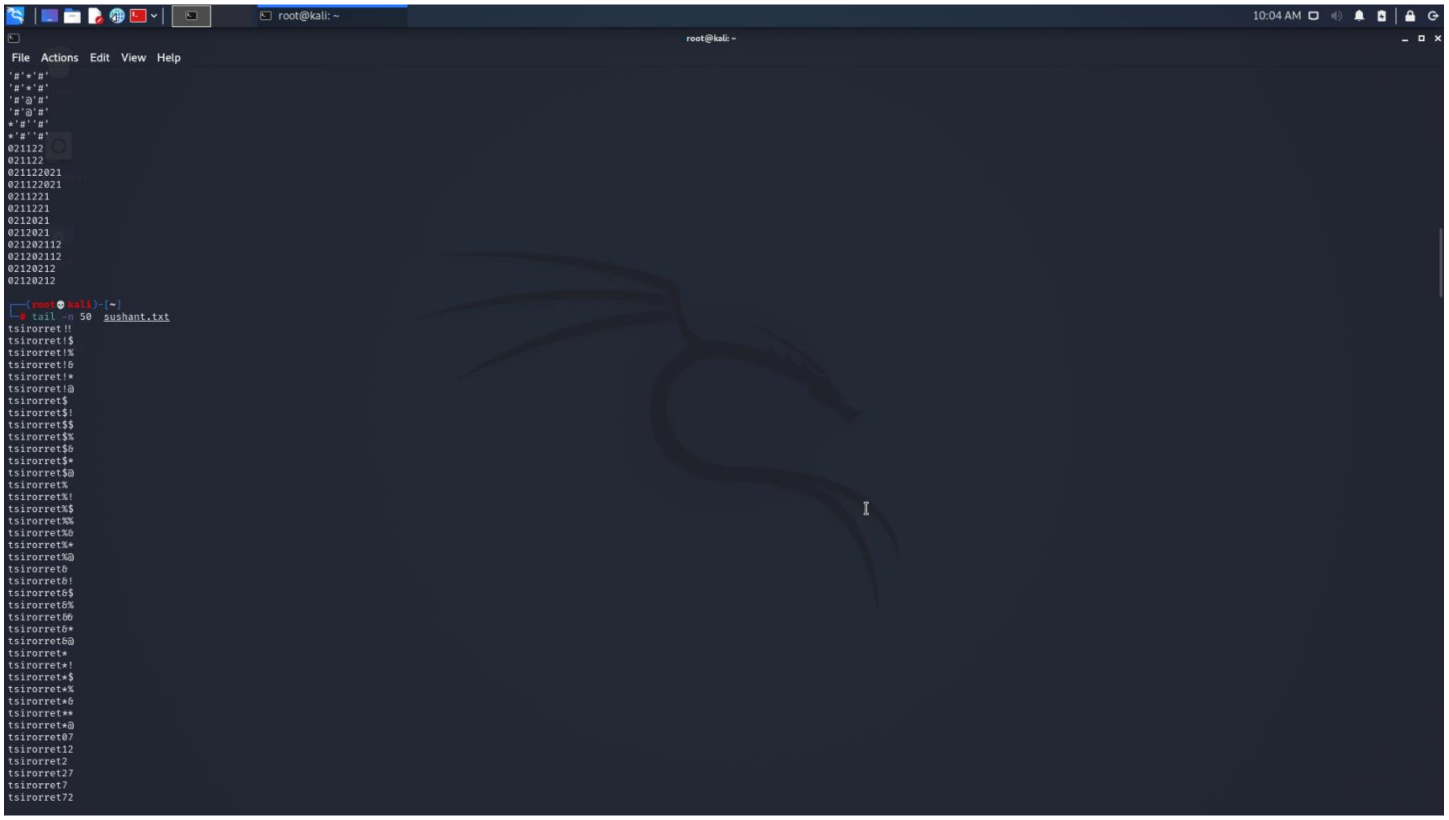
There's also an option to add any specific keywords, special characters, and random numbers. Apart from all this, there's the option to activate Leet mode, this will make the generated dictionary extremely effective.



```
┌──(root💀kali)-[~]
└─# cupp

 ___________
   cupp.py!                 # Common
      \                     # User
       \   ,__,             # Passwords
        \  (oo)____         # Profiler
           (__)    )\
              ||--|| *      [ Muris Kurgas | j0rgan@remote-exploit.org ]
                            [ Mebus | https://github.com/Mebus/]

usage: cupp [-h] [-i | -w FILENAME | -l | -a | -v] [-q]

Common User Passwords Profiler

optional arguments:
  -h, --help         show this help message and exit
  -i, --interactive  Interactive questions for user password profiling
  -w FILENAME        Use this option to improve existing dictionary, or WyD.pl output to make some pwnsauce
  -l                 Download huge wordlists from repository
  -a                 Parse default usernames and passwords directly from Alecto DB. Project Alecto uses purified
                     databases of Phenoelit and CIRT which were merged and enhanced
  -v, --version      Show the version of this program.
  -q, --quiet        Quiet mode (don't print banner)

┌──(root💀kali)-[~]
└─# cupp -i

 ___________
   cupp.py!                 # Common
      \                     # User
       \   ,__,             # Passwords
        \  (oo)____         # Profiler
           (__)    )\
              ||--|| *      [ Muris Kurgas | j0rgan@remote-exploit.org ]
                            [ Mebus | https://github.com/Mebus/]


[+] Insert the information about the victim to make a dictionary
[+] If you don't know all the info, just hit enter when asked! ;)

> First Name: sushant
> Surname: chauhan
> Nickname: terrorist
> Birthdate (DDMMYYYY): 07121999


> Partners) name: jane
> Partners) nickname: doe
> Partners) birthdate (DDMMYYYY): 09081998


> Child's name: biradar
> Child's nickname:
> Child's birthdate (DDMMYYYY): 12122021


> Pet's name: tinku
> Company name: michigan
```



```
[+] If you don't know all the info, just hit enter when asked! ;)

> First Name: sushant
> Surname: chauhan
> Nickname: terrorist
> Birthdate (DDMMYYYY): 07121999


> Partners) name: jane
> Partners) nickname: doe
> Partners) birthdate (DDMMYYYY): 09081998


> Child's name: biradar
> Child's nickname:
> Child's birthdate (DDMMYYYY): 12122021


> Pet's name: tinku
> Company name: michigan


> Do you want to add some key words about the victim? Y/[N]: y
> Please enter the words, separated by comma. [i.e. hacker,juice,black], spaces will be removed: hacker,coder
> Do you want to add special chars at the end of words? Y/[N]: y
> Do you want to add some random numbers at the end of words? Y/[N]:y
> Leet mode? (i.e. leet = 1337) Y/[N]: y

[+] Now making a dictionary ...
[+] Sorting list and removing duplicates ...
[+] Saving dictionary to sushant.txt, counting 36330 words.
[+] Now load your pistolero with sushant.txt and shoot! Good luck!

┌──(root💀kali)-[~]
└─# ls
alectodb.csv.gz  alectodb-passwords.txt  alectodb-usernames.txt  sushant.txt

┌──(root💀kali)-[~]
└─# head sushant.txt
!'#''#'
!'#''#'
$'#''#'
$'#''#'
%'#''#'
%'#''#'
&'#''#'
&'#''#'
'#'!'#'
'#'!'#'

┌──(root💀kali)-[~]
└─# head -n 50  sushant.txt
!'#''#'
!'#''#'
$'#''#'
$'#''#'
%'#''#'
%'#''#'
&'#''#'
&'#''#'
'#'!'#'
```

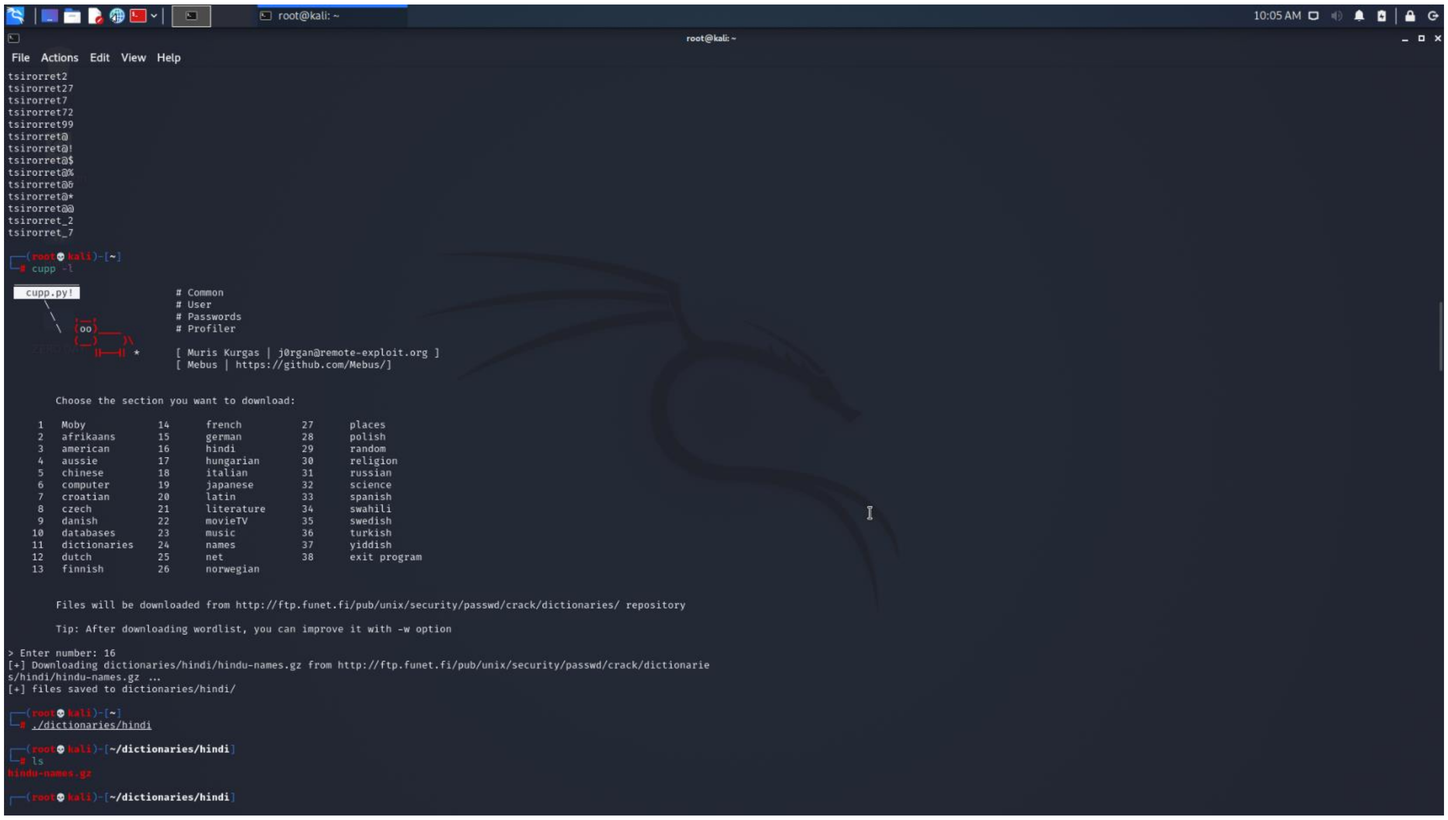


```
'#'*'#'
'#'*'#'
'#'@'#'
'#'@'#'
*'#''#'
*'#''#'
021122
021122
021122021
021122021
0211221
0211221
0212021
0212021
021202112
021202112
02120212
02120212

┌──(root💀kali)-[~]
└─# tail -n 50  sushant.txt
tsirorret!!
tsirorret!$
tsirorret!%
tsirorret!&
tsirorret!*
tsirorret!@
tsirorret$
tsirorret$!
tsirorret$$
tsirorret$%
tsirorret$&
tsirorret$*
tsirorret$@
tsirorret%
tsirorret%!
tsirorret%$
tsirorret%%
tsirorret%&
tsirorret%*
tsirorret%@
tsirorret&
tsirorret&!
tsirorret&$
tsirorret&%
tsirorret&&
tsirorret&*
tsirorret&@
tsirorret*
tsirorret*!
tsirorret*$
tsirorret*%
tsirorret*&
tsirorret**
tsirorret*@
tsirorret07
tsirorret12
tsirorret2
tsirorret27
tsirorret7
tsirorret72
```

Downloading Dictionaries from Cupp Repository:

Cupp has its own repositories of dictionaries which are pre-classified. These dictionaries can be downloaded and used. The downloaded files are compressed and have to be uncompressed to be viewed.

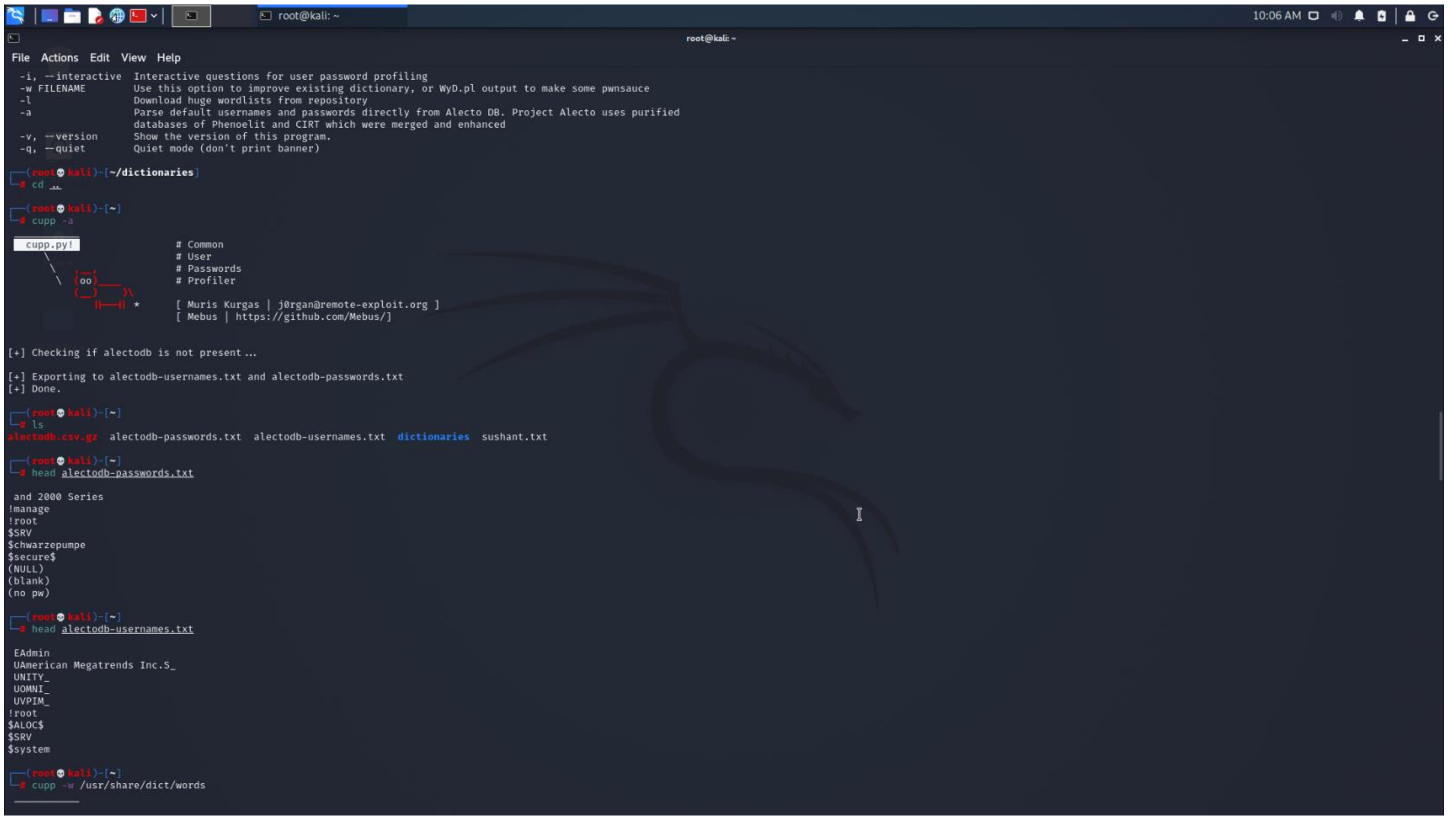
Enter the number to choose a name to select the dictionary you want to download; we have **pressed 16** and downloaded to view a dictionary of Hindi names.



```
tsirorret2
tsirorret27
tsirorret7
tsirorret72
tsirorret99
tsirorret@
tsirorret@!
tsirorret@$
tsirorret@%
tsirorret@&
tsirorret@*
tsirorret@@
tsirorret_2
tsirorret_7

┌──(root💀kali)-[~]
└─# cupp -l
 ___________
   cupp.py!                 # Common
      \                     # User
       \   ,__,             # Passwords
        \  (oo)____         # Profiler
           (__)    )\
              ||--|| *      [ Muris Kurgas | j0rgan@remote-exploit.org ]
                            [ Mebus | https://github.com/Mebus/]


        Choose the section you want to download:

     1   Moby            14      french          27      places
     2   afrikaans       15      german          28      polish
     3   american        16      hindi           29      random
     4   aussie          17      hungarian       30      religion
     5   chinese         18      italian         31      russian
     6   computer        19      japanese        32      science
     7   croatian        20      latin           33      spanish
     8   czech           21      literature      34      swahili
     9   danish          22      movieTV         35      swedish
    10   databases       23      music           36      turkish
    11   dictionaries    24      names           37      yiddish
    12   dutch           25      net             38      exit program
    13   finnish         26      norwegian


        Files will be downloaded from http://ftp.funet.fi/pub/unix/security/passwd/crack/dictionaries/ repository

        Tip: After downloading wordlist, you can improve it with -w option

> Enter number: 16
[+] Downloading dictionaries/hindi/hindu-names.gz from http://ftp.funet.fi/pub/unix/security/passwd/crack/dictionarie
s/hindi/hindu-names.gz ...
[+] files saved to dictionaries/hindi/

┌──(root💀kali)-[~]
└─# ./dictionaries/hindi

┌──(root💀kali)-[~/dictionaries/hindi]
└─# ls
hindu-names.gz

┌──(root💀kali)-[~/dictionaries/hindi]
```

Downloading Default Usernames and Passwords:

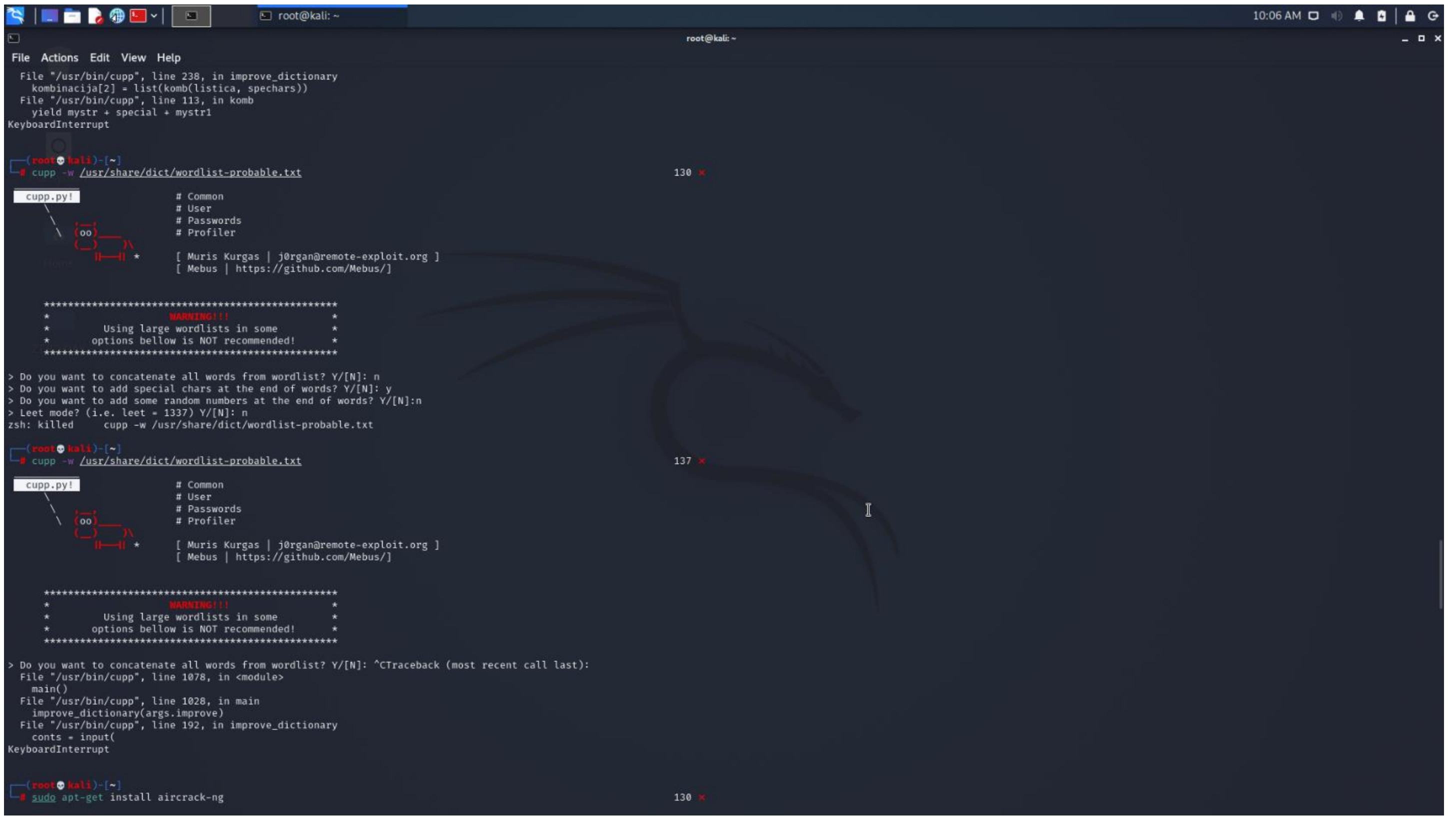
Cupp can download premade dictionaries holding the most common usernames and passwords from the project Alecto database for usage.



```
  -i, --interactive  Interactive questions for user password profiling
  -w FILENAME        Use this option to improve existing dictionary, or WyD.pl output to make some pwnsauce
  -l                 Download huge wordlists from repository
  -a                 Parse default usernames and passwords directly from Alecto DB. Project Alecto uses purified
                     databases of Phenoelit and CIRT which were merged and enhanced
  -v, --version      Show the version of this program.
  -q, --quiet        Quiet mode (don't print banner)

┌──(root㉿kali)-[~/dictionaries]
└─# cd ..

┌──(root㉿kali)-[~]
└─# cupp -a

  cupp.py!                 # Common
      \                    # User
       \   ,__,            # Passwords
        \  (oo)____        # Profiler
           (__)    )\
              ||--|| *     [ Muris Kurgas | j0rgan@remote-exploit.org ]
                           [ Mebus | https://github.com/Mebus/]


[+] Checking if alectodb is not present ...

[+] Exporting to alectodb-usernames.txt and alectodb-passwords.txt
[+] Done.

┌──(root㉿kali)-[~]
└─# ls
alectodb.csv.gz  alectodb-passwords.txt  alectodb-usernames.txt  dictionaries  sushant.txt

┌──(root㉿kali)-[~]
└─# head alectodb-passwords.txt

 and 2000 Series
!manage
!root
$SRV
$chwarzepumpe
$secure$
(NULL)
(blank)
(no pw)

┌──(root㉿kali)-[~]
└─# head alectodb-usernames.txt

 EAdmin
 UAmerican Megatrends Inc.S_
 UNITY_
 UOMNI_
 UVPIM_
!root
$ALOC$
$SRV
$system

┌──(root㉿kali)-[~]
└─# cupp -w /usr/share/dict/words
```

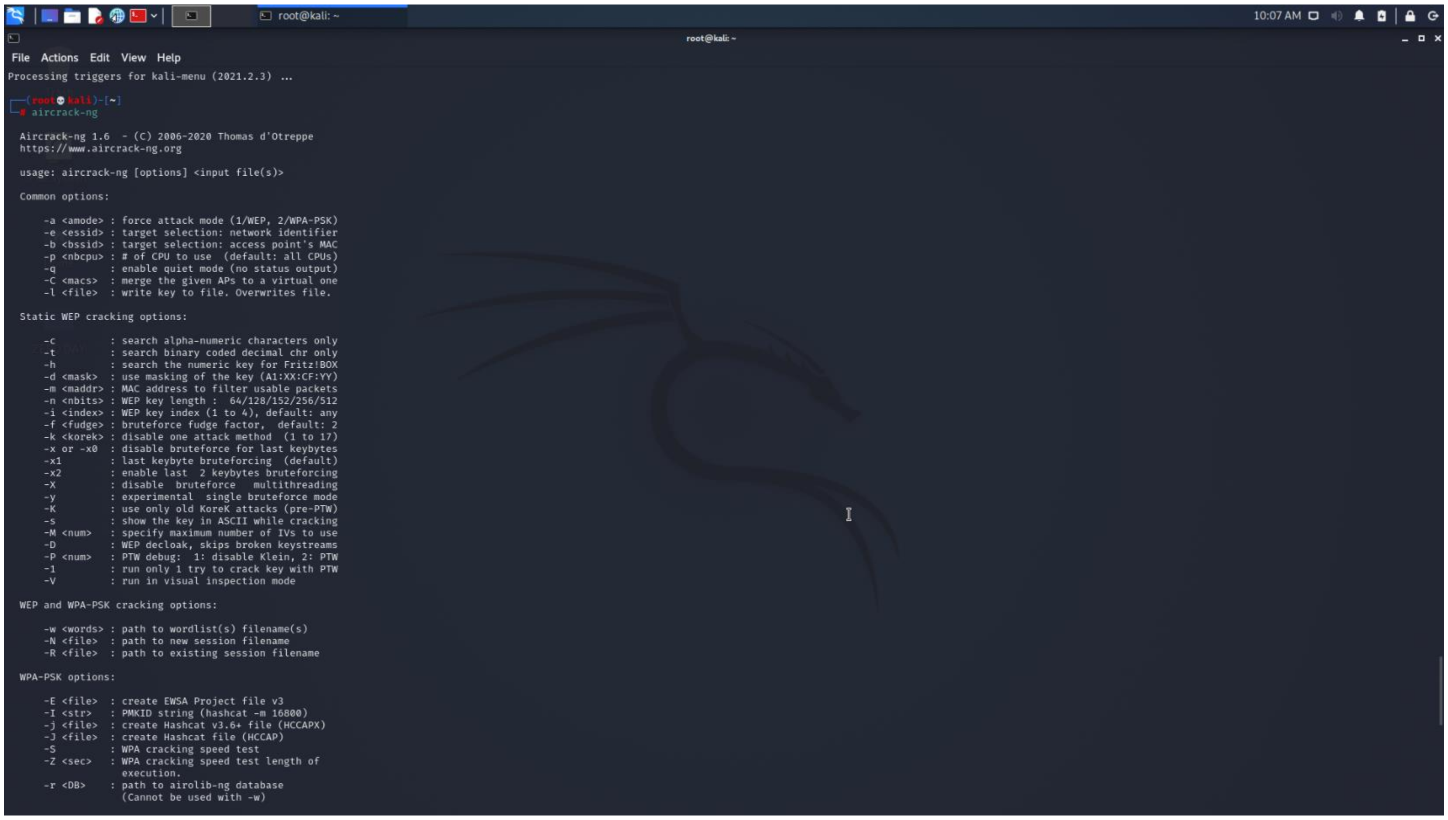
Adding to Custom Dictionary:

Cupp gives us the option to add more words to our created dictionary. We can customize the kind of words we would like to add by using the provided options.



```
  File "/usr/bin/cupp", line 238, in improve_dictionary
    kombinacija[2] = list(komb(listica, spechars))
  File "/usr/bin/cupp", line 113, in komb
    yield mystr + special + mystr1
KeyboardInterrupt


┌──(root💀kali)-[~]
└─# cupp -w /usr/share/dict/wordlist-probable.txt                130 ✘

 cupp.py!                 # Common
      \                   # User
       \   ,__,           # Passwords
        \  (oo)____       # Profiler
           (__)    )\
              ||--|| *    [ Muris Kurgas | j0rgan@remote-exploit.org ]
                          [ Mebus | https://github.com/Mebus/]


      *************************************************
      *                  WARNING!!!                   *
      *         Using large wordlists in some         *
      *       options bellow is NOT recommended!      *
      *************************************************

> Do you want to concatenate all words from wordlist? Y/[N]: n
> Do you want to add special chars at the end of words? Y/[N]: y
> Do you want to add some random numbers at the end of words? Y/[N]:n
> Leet mode? (i.e. leet = 1337) Y/[N]: n
zsh: killed     cupp -w /usr/share/dict/wordlist-probable.txt

┌──(root💀kali)-[~]
└─# cupp -w /usr/share/dict/wordlist-probable.txt                137 ✘

 cupp.py!                 # Common
      \                   # User
       \   ,__,           # Passwords
        \  (oo)____       # Profiler
           (__)    )\
              ||--|| *    [ Muris Kurgas | j0rgan@remote-exploit.org ]
                          [ Mebus | https://github.com/Mebus/]


      *************************************************
      *                  WARNING!!!                   *
      *         Using large wordlists in some         *
      *       options bellow is NOT recommended!      *
      *************************************************

> Do you want to concatenate all words from wordlist? Y/[N]: ^CTraceback (most recent call last):
  File "/usr/bin/cupp", line 1078, in <module>
    main()
  File "/usr/bin/cupp", line 1028, in main
    improve_dictionary(args.improve)
  File "/usr/bin/cupp", line 192, in improve_dictionary
    conts = input(
KeyboardInterrupt


┌──(root💀kali)-[~]
└─# sudo apt-get install aircrack-ng                             130 ✘
```

Aircrack-ng:



```
Processing triggers for kali-menu (2021.2.3) ...

┌──(root💀kali)-[~]
└─# aircrack-ng

  Aircrack-ng 1.6  - (C) 2006-2020 Thomas d'Otreppe
  https://www.aircrack-ng.org

  usage: aircrack-ng [options] <input file(s)>

  Common options:

      -a <amode> : force attack mode (1/WEP, 2/WPA-PSK)
      -e <essid> : target selection: network identifier
      -b <bssid> : target selection: access point's MAC
      -p <nbcpu> : # of CPU to use  (default: all CPUs)
      -q         : enable quiet mode (no status output)
      -C <macs>  : merge the given APs to a virtual one
      -l <file>  : write key to file. Overwrites file.

  Static WEP cracking options:

      -c         : search alpha-numeric characters only
      -t         : search binary coded decimal chr only
      -h         : search the numeric key for Fritz!BOX
      -d <mask>  : use masking of the key (A1:XX:CF:YY)
      -m <maddr> : MAC address to filter usable packets
      -n <nbits> : WEP key length :  64/128/152/256/512
      -i <index> : WEP key index (1 to 4), default: any
      -f <fudge> : bruteforce fudge factor,  default: 2
      -k <korek> : disable one attack method  (1 to 17)
      -x or -x0  : disable bruteforce for last keybytes
      -x1        : last keybyte bruteforcing  (default)
      -x2        : enable last  2 keybytes bruteforcing
      -X         : disable  bruteforce   multithreading
      -y         : experimental  single bruteforce mode
      -K         : use only old KoreK attacks (pre-PTW)
      -s         : show the key in ASCII while cracking
      -M <num>   : specify maximum number of IVs to use
      -D         : WEP decloak, skips broken keystreams
      -P <num>   : PTW debug:  1: disable Klein, 2: PTW
      -1         : run only 1 try to crack key with PTW
      -V         : run in visual inspection mode

  WEP and WPA-PSK cracking options:

      -w <words> : path to wordlist(s) filename(s)
      -N <file>  : path to new session filename
      -R <file>  : path to existing session filename

  WPA-PSK options:

      -E <file>  : create EWSA Project file v3
      -I <str>   : PMKID string (hashcat -m 16800)
      -j <file>  : create Hashcat v3.6+ file (HCCAPX)
      -J <file>  : create Hashcat file (HCCAP)
      -S         : WPA cracking speed test
      -Z <sec>   : WPA cracking speed test length of
                   execution.
      -r <DB>    : path to airolib-ng database
                   (Cannot be used with -w)
```

Quiet Mode:

Quiet mode is for running Cupp in a more hush-hush way. If you're the kind of person who does not want a big banner on their screen showing everyone what you're doing, you'll like this option. This basically makes for a cleaner screen while cupp is carrying out the commands you're giving it, without the funny cow popping up on top.

```
cupp -a -q
```

GitHub Link: https://github.com/vishalpoonacha/Zero-Day-Cybersec.git

Video Link: https://drive.google.com/drive/folders/1RpUJiPt-_VbGknrx-jw0HoPsyv2WFI6l?usp=sharing