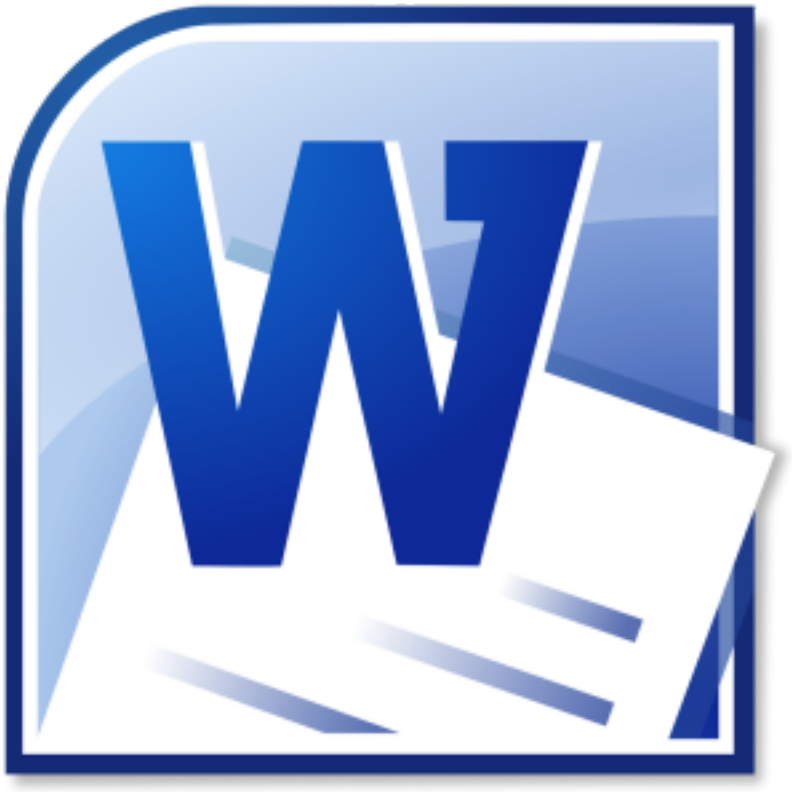

Custom Wordlists

How to make and when to use custom wordlists

[Task 1] When to use custom wordlists?

(This room is depreciated and the information provided is out-dated. If you still wish to learn how to do this manually read on, but there are better methods available. This room will be updated in the future to include them)

You've likely encountered a website that requires you to have Special Characters, Capital Letters, and a Number in your password.

These are password rules, aimed to make your account more secure and harder for attackers to guess.

Many users just alter their current easy to guess passwords into something that'll be accepted.
As an example, password -> password1 -> Password1 -> P@ssword1 - Are all variants of the easy to guess password.

If you're trying to bruteforce someones password and you know the password requirements are: 1 Capital Letter, 1 Number, 8 characters.
You wouldn't want to use a wordlist that is 80% lowercase strings.

In this room we'll be going over how to edit existing wordlists as to not attempt passwords that dont follow the password requirements.

We'll go over Wordlist Filtering, Wordlist Modifying, and Wordlist Generating.

This room will begin with simple ways to modify wordlists, and then move on to more robust albiet more complicated ways to modify wordlists.

#1
Read the above

No answer needed

[Task 2] Tools used in this Room

Throughout this room we'll be testing these new wordlists against various password encrypted zip files.

You can crack these files either using fcrackzip (Installable with: `sudo apt install fcrackzip`)

Fcrack Syntax: `fcrackzip -v -u -D -p wordlist.txt flagfile.zip`

Or John the Ripper:
Pass .zip to a readable hash:
`zip2john flagfile.zip > flaghash.txt`
John Syntax:
`john ziphash.txt --wordlist=wordlist.txt`

The wordlist being used in this room can be found here: https://github.com/danielmiessler/SecLists/blob/master/-Passwords/Leaked-Databases/rockyou.txt.tar.gz

#1
Read the above

No answer needed

[Task 3] Filter Wordlists

Filtering wordlists is getting rid of lines that DONT contain the minimums you want. So if your testing against a site that has
a minimum length requirement of 8 characters, you'd want to get rid of all strings that are less than 8 characters.
Below is a list of grep commands that will filter output, and write the filtered output to a file.

You can combine these commands to create more specific wordlists.
As an example the command: `cat rockyou.txt | grep -o '[^ ]*[a-z][^ ]*' > contains_lowercase.txt` would only keep lines of rockyou.txt that contain lowercase characters.

Use these commands to alter rockyou.txt to crack the various zipfiles.

```
grep -x '.\{8,20\}' > 8-20_length_wordlist
grep -o '[^ ]*[a-z][^ ]*' > contains_lowercase.txt
grep -o '[^ ]*[A-Z][^ ]*' > contains_uppercase.txt
grep -o '[^ ]*[0-9][^ ]*' > contains_numbers.txt
grep -v "^[A-Za-z0-9]*$" > contains_special.txt
```

#1
Get flag 1 Password Requirements: 8 length minimum

--**ran cat rockyou.txt | grep -x '.\{8,20\}' > 8-20_length_wordlist**
--then ran zip2john flag1.txt.zip > flag1hash.txt and cracked with johnny
password: megsandhealie

THM{n1c3_w0rk}

#2
Get flag2 Password Requirements: 1 Uppercase

--ran cat rockyou.txt | grep -o '[^]*[A-Z][^]*' > contains_uppercase.txt
--then ran zip2john flag2.txt.zip > flag2hash.txt
password: CASPER12MAY6

THM{N3xt_0n3_r3qu1r35_2_c0mm4nd5}

#3
Get flag 3 Password Requirements: 10 length minimum, Uppercase, Special Characters

--ran cat 8-20_length_wordlist | grep -o '[^]*[A-Z][^]*' | grep -v "^[A-Za-z0-9]*$" > contains_special.txt
--then ran zip2john flag3.txt.zip > flag3hash.txt
password: HIZLER_SK8_OR_DIE@hotmail.com

THM{g00d_j0b!}

[Task 4] Modify Wordlists

In some instances, rather than filter out strings that don't match your requirements, you'd want to modify your wordlists lines to MEET those requirements.

So let's say you have a wordlist that contains "password", but your target has a minimum password requirement of 1 capital letter,
being able to change each first lowercase letter to capital, so in this case password -> Password, can be very beneficial.

To replace the first character in a string with a capital version: `sed -e "s/\b\(.\)/\u\1/g"`

To append SUFFIX to the end of each line: `sed -e 's/$/\SUFFIX/'`

#1
Get flag 4 Password Requirements: 1 Uppercase

--ran `cat rockyou.txt | sed -e "s/\b\(.\)/\u\1/g" > 1upper.txt`
--then ran `zip2john flag4.txt.zip > flag4hash.txt` and cracked with johnny
password: Basketball

THM{M0d1f13r5_4r3_c00l}

#2
Get flag 5 Password Requirements: 1 Special Character

--ran `cat rockyou.txt | sed -e 's/$/\!/' > special.txt`
--then ran `zip2john flag5.txt.zip > flag5hash.txt`
password: rylan17!

THM{h0m3_str3tch}

[Task 5] Generating Wordlists

The big guns, Generating Wordlists with Hashcat rules:

Now that we've gone over simple ways to alter wordlists using grep and sed, it's time to discuss their limitations. Sed and grep are useful for simple changes to existing wordlists, but what about generating advanced alterations, like modifying a wordlist to l33t sp33k? Or performing many of the kinds of modifications we just completed? Thats where hashcat rules come in. Hashcat rules will take an input and apply whatever 'rules' you tell it too, you can pass as many of these
rules as you want.

Download HashcatRulesEngine and navigate to the directory you want HashcatRulesEngine to be installed to
`git clone https://github.com/llamasoft/HashcatRulesEngine cd HashcatRulesEngine make`

Download some Hashcat rules from https://contest-2010.korelogic.com/rules-hashcat.html

We'll only be using KoreLogicRulesL33t.rule and KoreLogicRulesAppendMonthCurrentYear.rule

For convenience I reccommend making and placing these in a RULES folder in your HashcatRulesEngine directory

To use HashcatRulesEngine the syntax is as follows:
`./hcre rulefile.rule < wordlist.txt > new_wordlist.txt`

==========

The final flag. We won't be using rockyou for this as we'll be generating our own wordlist. Gathered intel suggests that the user bases all their passwords around their pets name "HorseShark".
Make a text file containing "HorseShark" (without the quotes) and pass that file to the HashcatRulesEngine using the syntax discussed above.
Apply the two rulefiles KoreLogicRulesL33t.rule and KoreLogicRulesAppendMonthCurrentYear.rule

Then, with your new wordlist try to crack the last .zip

#1
What is flag 6?

--ran ./hcre '/home/taj702/software/HashcatRulesEngine/hashcat-rules/KoreLogicRulesL33t.rule' '/home/taj702/-software/HashcatRulesEngine/hashcat-rules/KoreLogicRulesAppendMonthCurrentYear.rule' < '/media/taj702/CTF-DATA/tryhackme/Custom_Wordlists/FlagsPart3/Flags_Part3/horseshark.txt' > newHorse.txt

--then ran zip2john Flag6.txt.zip > flag6hash.txt
password: H0r3h4rk

THM{w0rdl15t_k1ng}