Introduction

Mobile Device Forensics involves the collection of forensically sound evidence (or admissible evidence that can be presented to the court of law). Using the program you developed in lab 2, find the hash code of the following two files MDF-SF1.docx and MDF-SF2.docx. Since these two files are password protected, write a Python program to break the password. After breaking the password find the hash code again. Compare the values of hash code before and after breaking passwords. List and use two more password management tools for breaking password. Compare the hash codes with those generated by your program. Make sure that you are program is documented. It is important that your program is uploaded in GitHub as well or any other web based repository

Researching the problem

Office 2007-2010 documents are Microsoft file type .

According to the Microsoft whitepaper (microsoft, 2016)the following specs are defined:

* Key derivation is performed using 50,000 iterations[ of SHA-1 (Leblanc, 2016)
* Uses a 16-byte (128-bit) random salt.
* AES is the block cipher used to encrypt the document.
* By default, 128-bit key are used. There is a registry tweak to change this to 256-bit.

Tools to crack:

There are many opensource and paid applictoins that clam to crack office documanrs some are:

* Ophcrack:

http://ophcrack.sourceforge.net/

* Cain and able

http://www.oxid.it/cain.html

* John the ripper

http://www.openwall.com/john/

* Advanced Office Password Recovery

<https://www.elcomsoft.com/aopr.html>

* Hascat

Http://hashcat.net

I wanted to use an open source cracking tool and there are many. I picked john the ripper

MD5 Hash of both documents before any cracking :

MDF-SF1.docx : dd06b1c03f4ad67c99d4f371f4001fa4

MDF-SF2.docx : 69a6db156d051c48e655d5a95cff84b4

How to crack a password.

There are three main way to crack

Dictionary Attack

Dictionary attacks are just what they sound like: you [use the dictionary](http://www.quickanddirtytips.com/education/grammar/the-oxford-english-dictionary-needs-your-help)to find a password. Hackers basically have very large text files that include millions of generic passwords, such as password, iloveyou, 12345, admin, or 123546789.

Mask/Character Set Attack

If a hacker can’t guess your password from a dictionary of known passwords, their next option will be to use some general rules to try a lot of combinations of specified characters. This means that instead of trying a list of passwords, a hacker would specify a list of characters to try.

For example, if I knew your password was [just numbers](http://www.quickanddirtytips.com/education/math), I would tell my program to only try number combinations as passwords. From here, the program would try every combination of numbers until it cracked the password. Hackers can specify a ton of other settings, like minimum and maximum length, how many times to repeat a specific character in a row, and many more. This decreases the amount of work the program would need to do.

So, let's say I had an 8 character password made up of just numbers. Using my graphics card, it would take about 200 seconds--just over 3 minutes--to crack this password. However, if the password included lowercase letters and numbers, the same 8 character password would take about 2 days to decode.

Bruteforce

If an attacker has had no luck with these two methods, they may also "bruteforce" your password. A bruteforce tries every character combination until it gets the password. Generally, this type of attack is impractical, though--as anything over 10 characters would take millions of years to figure out!

Choosing the method

The first method is to use a Directory accctack. As we have been told the password s easy and is between 4 and 5 characters. This seems the best option to try. A Directory attack needs a wordlist of as many passwords as possible. There are many available online .they are mostly from data breaches that have acoured in companies over the last few years. (CITEAN)

You can also create your own list using a Linux tool crunch. This will populate a list with whatever parameters you request. Below is an example to create a list with minimum word size of 4 and maximum word size of 5 and use the character set of all the alphabet and numbers 0-9. A file will be created called mix-wordlist.txt

*/usr/bin/crunch 3 3 -f /usr/share/crunch/charset.lst mixalpha-numeric-all-space -o mix\_wordlist.txt*

One of the best seems to be the rockyou list. (BOWES, 2016)

This list is the result of a hack on rockyou.com. It had a data breached that resulted in the exposure of over 32 Million user accounts. To compound the severity of the security breach, it was found that RockYou are storing all user account data in plain text in their database, exposing all that information to attackers

(Cubrilovic, 2016)

Precess of the crack

The are two main stages of cracking the file stage one is using office2john paython program:

Pass your file into office2john a python program . it extract the hashes. One example is: *365-2013-password.docx:$office$\*2013\*100000\*256\*16\*d4fc9302eedabf9872b24ca700a5258b\*7c9554d582520747ec3e872f109a7026\*1af5b5024f00e35eaf5fd8148b410b57e7451a32898acaf14275a8c119c3a4fd*

Stage two

Now that we have the Hash and world list . We input this hash and other options into john.

./john --wordlist=rockyou.lst doc.hash

Results

The two passwords were cracked over a few hours :

The two passwords are:

MDF-S1.docx :4321

MDF-SF2.docx :1a2b3

MD5 Hash of both documents after cracking :

MDF-SF1.docx : dd06b1c03f4ad67c99d4f371f4001fa4

MDF-SF2.docx : 69a6db156d051c48e655d5a95cff84b4

.

Other programs

advanced office password

I used advanced office password recovery tool in windows

The results were the same . As I knew the passwords were simple I could set the tool to use a limited set of crachtors which speeds the recovery highly. The Hash of the files as excepted did not change

MDF-SF1.docx : dd06b1c03f4ad67c99d4f371f4001fa4

MDF-SF2.docx : 69a6db156d051c48e655d5a95cff84b4

Hashcat

This is another Linux tool which like john need the hashes already exported for it to work I again used the office2john.py tool . The command is

cudaHashcat64.exe -a 0 -m 9400 --username -o found.txt hash.txt pass.txt

MDF-SF1.docx : dd06b1c03f4ad67c99d4f371f4001fa4

MDF-SF2.docx : 69a6db156d051c48e655d5a95cff84b4

I did not have the skills to write a program to do this. While I researching the process I found it over comp laced when using multiple programs. I decided to automate the process. I wrote a script. It will ask the user for the office file name and which word list you want to use. It will then pass the details to office2john and output the results to a file, then input this with outer variables into the john application. The cracking process will then start automatically .

The script is called office2john2john.sh and is avable a a complete package at github:

https://github.com/rdunne/lab3

apendex

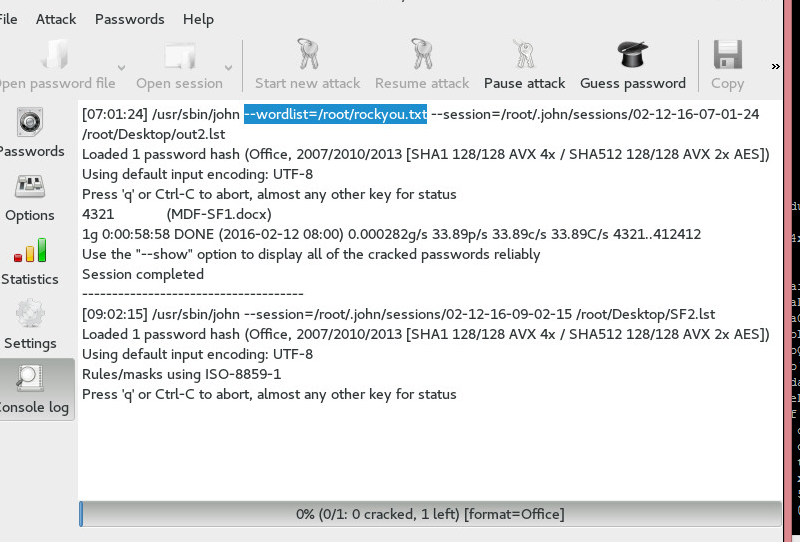


Figure : using john to crack file MDF1

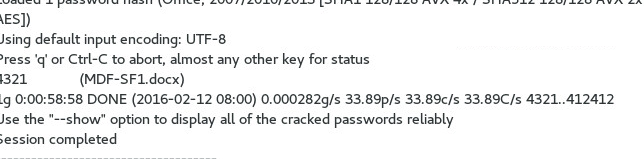


Figure : first hash cracked

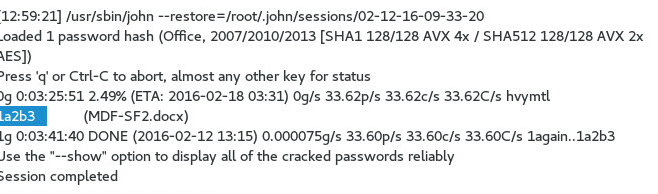


Figure second hash cracked

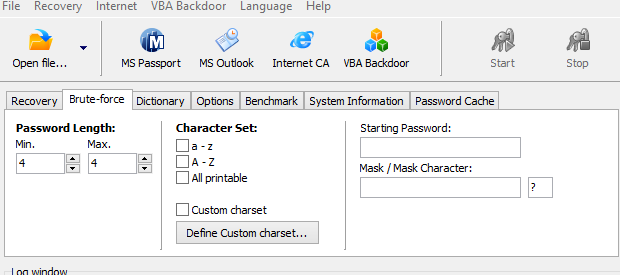


Figure office recovery tool set for a 4X4 attack

Office2john2john.sh

*#!/bin/bash*

*echo -n "Enter your office documen[ENTER]: "*

*read doc*

*echo -n "Enter your wordlist, if none just leave blank[ENTER]: "*

*read list*

*echo "You entered:$doc"*

*python office2john.py $doc > $doc.hash*

*echo " your hash is now stored in $doc.hash"*

*./john --wordlist=$list --session=`date +%Y%m%d\_%H%M` $doc.hash*

# References

BOWES, R. (2016, 2 19). *skullsecurity*. Retrieved from skullsecurity: http://downloads.skullsecurity.org/passwords/rockyou.txt.bz2

Cubrilovic, N. (2016, 2 19). *RockYou Hack: From Bad To Worse*. Retrieved from techcrunch.com: http://techcrunch.com/2009/12/14/rockyou-hack-security-myspace-facebook-passwords/

Leblanc, D. (2016, 2 15). *blogs.msdn.com*. Retrieved from blogs.msdn.com: http://blogs.msdn.com/b/david\_leblanc/archive/2008/12/05/office-crypto-kdf-details.aspx

microsoft. (2016, 2 15). *technet*. Retrieved from microsoft: http://technet.microsoft.com/en-us/library/cc179080.aspx