**Assignment 4**

**Due next class**

**Instructions**

* **Print (and staple) and bring to class**
* **Answer in this document**

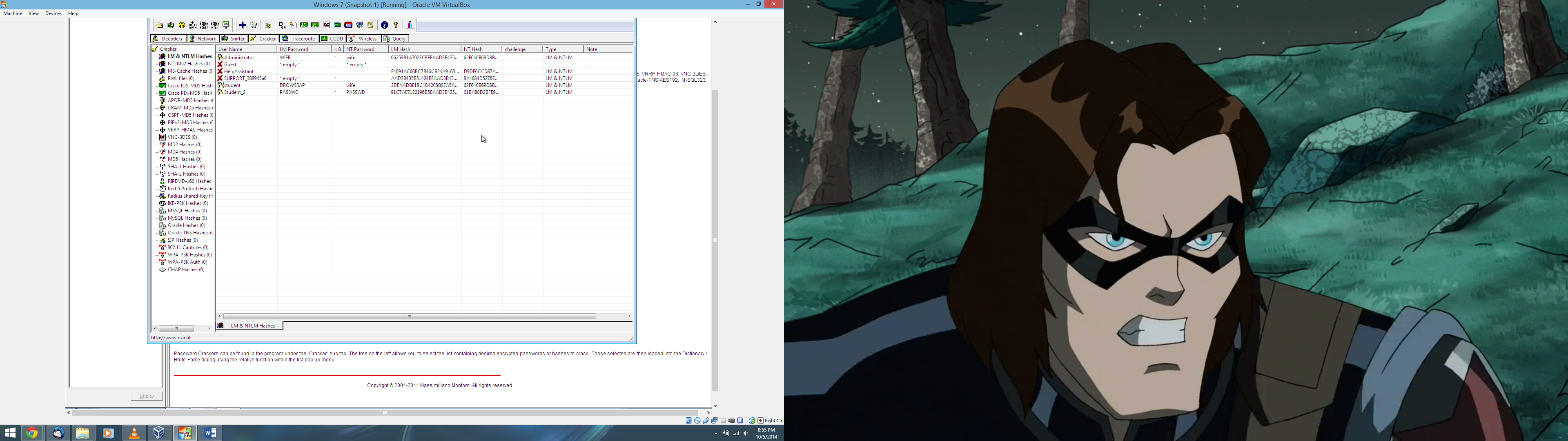
**Q1 (10pts)**

Download and use “Cain and Abel” to break the Windows SAM file included with this assignment for users “Administrator”, “student”, and “student\_2”. You will need some dictionary files and a Boot Key file (both are also included). Describe how you broke the passwords and show a screen shot of the cracked passwords.

Your anti-virus software will NOT like Cain and Abel. I highly recommend running this in a VM or a computer that does not have any of your personal data. You can create a Windows 7 VM using Virtualbox. You can download a copy of Windows 7 from: (<http://e5.onthehub.com/WebStore/ProductsByMajorVersionList.aspx?ws=dcbce046-749b-e011-969d-0030487d8897&vsro=8>).

For Mac or Linux you can try <https://www.winehq.org> (I don’t know if it will work, so let me know if your try this).

The LM passwords are WIFE, DROWSSAP, and PASSWD for Administrator, student and Student\_2 respectively. The NT passwords are wife, wife and PASSWD for the same accounts respectively.

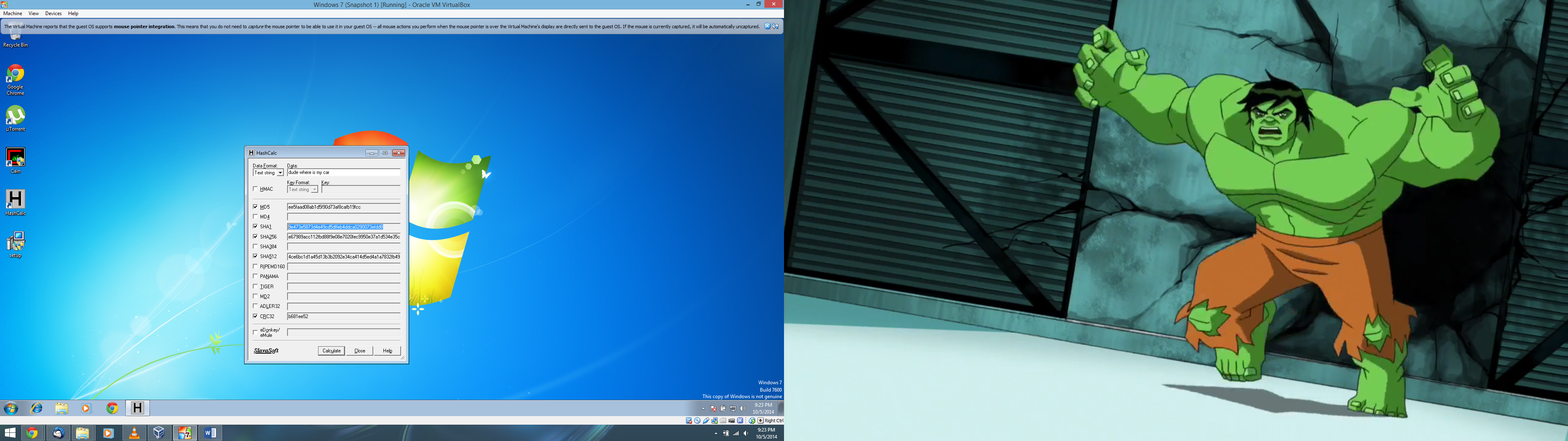


I was able to decrypt the passwords by going to the Cracker tab and selecting the Import (+) button for LM & NTLM Hashes. After selecting the Boot key and the SAM file, I was able to see the user names. For each one, I was able to right click the account and then selected Dictionary Attack > LM Hashes or NTLM Hashes for the LM and NTLM passwords respectively. I included the dictionaries from the tar file and then I selected the greatest number of possible tests including numbers and case perms. This yielded most of the passwords for each of the accounts although it did require resetting the starting positions of the dictionaries at each time. For the DROWSSAP password, it showed the password as DROWSSA????. I guessed it was DROWSSAP since that was PASSWORD spelt backwards. This seemed to work as re-running the dictionary attacks using this test worked.

I was able to confirm that the NTLM passwords for the Administrator and student accounts were the same as they had the same hashes, which means the passwords aren’t salted.

**Q2 (5pts)**

Download the tool HashCalc from <http://download.cnet.com/HashCalc/3000-2250_4-10130770.html> or use <http://www.fileformat.info/tool/hash.htm?hex=0101010>.



What are the first 4 bytes (**represent in hex format**) of the hash of the test string “dude where is my car” using the following hash functions:

* + CRC32: **b681ee52**
  + MD5: **ee5faad0**
  + SHA-1: **9e473e58**
  + SHA-256: **e67989ac**
  + SHA-512: **4ce6bc1d**

**Q3 (5pts)**

What are the indices of the non-zero values within the table T of a bloom filter that has the following characteristics:

* + N = 256
  + The bloom filter uses 4 hash functions (i.e. k =4): MD5, SHA-1, SHA-512, and CRC32.
  + The dictionary used has the following words: Password, drowssaP, wordPass, and ssaPdrow

What value of N should have been used to make the probability of false positives 0.01?

First question using HashCalc and Python to calculate the modulus operations and hex to decimal conversions:

Password indices:

MD5: dc647eb65e6711e155375218212b3964 => 292951958790886807227788297787532851556 % 256 = **100**

SHA1: 8be3c943b1609fffbfc51aad666d0a04adf83c9d => 798629518908746073751996426463562287984034725021 % 256 = **157**

SHA512: e6c83b282aeb2e022844595721cc00bbda47cb24537c1779f9bb84f04039e1676e6ba8573e588da1052510e3aa0a32a9e55879ae22b0c2d62136fc0a3e85f8bb => 12087042095939914286886421484777595552003264103724315653587730213921780698737447210662344513561062973695709305278085255752357076504839866291415026887948475 % 256 = **187**

CRC32: ccb42483 => 3434357891 % 256 = **131**

drowssaP indices:

MD5: f56c803b9d4be545ef3c10223158ed77 => 326224227899597904691135069064814718327 % 256 = **119**

SHA1: c03be152e942adbc70269f36abaf373c2d5fbf35 => 1097461600442435567387577419866055392881287348021 % 256 = **53**

SHA512: 7c81ba1ee0597ce5e33a9a6b05ce3f496841953fa6d4db26055c5dee459639f2510ea9a99eabfb8be01f3aff0048dec8946477ab7123b6ed513ab96a29dae655 =>

6520947419409329126219736571685149312661798834501552551450172295001642370202024429873517610291671406053741547289452691079889041859749179527978908630312533 % 256 = **85**

CRC32: c094e313 => 3230982931 % 256 = **19**

wordPass indices:

MD5: 0d5397f737cfc06858e2b23b434faeb8 => 17714006814941421270508103863647579832 % 256 = **184**

SHA1: 2f154bec517acb073b6775a60cea0472dba4b8e1 => 268797495714278121862356690572784500968626895073 % 256 = **225**

SHA512: 261032a40847c176f16c5009009db834faf5150343384371df0799e150a2cc5745d07172858e0af1d4459312ba533f59cb920b8546e2ac8f93200f494d4c8b02 => 1993535350657929356945922175730112296737438965453341978286186072296420797630637463288136358243604399032931393043620511032678451321325800296456722345396994 % 256 = **2**

CRC32: db17a2ad => 3675759277 % 256 = **173**

ssaPdrow indices:

MD5: 558b75997d3b1a51e05f8081505feda6 => 113708494107643759307980596527541251494 % 256 = **166**

SHA1: aff5a8ccf5f438a78b22743ef825b51ae02cabff => 1004551772076380235493247508912588980061867912191 % 256 = **255**

SHA512: 8f3d8a81885262c5f5296593ad4c75b24a230bc83bf8e99c21243be8bd8db6455fc264bcea0af0b99e8fea4212ed7547cbcbf2947a691959c9438585b3aeb6a0 => 7502108202059678486190838413237706464412249264511543400411025788630247030131833407371836629577918110390300663072348174250640354150512380539723194768996000 % 256 = **160**

CRC32: aef3df05 => 2935217925 % 256 = **5**

**Tl;dr The indices are [2, 5, 19, 53, 85, 100, 119, 131, 157, 160, 166, 173, 184, 187, 225, 255]**

Second question:

With the inputs of k = 4, p = 0.01 and D = 4 and the formula: , we can solve for N by solving: which is . This becomes approximately 42.09 which rounds up to 43.

**Q4 (5pts)**

The MD5 hash of one of the 5 letter words in this assignment is “aab9e1de16f38176f86d7a92ba337a8d”. Can you find out which word it is?

Using the MD5 hash cracking capability of Cain & Abel along with the dictionaries provided for question 1, the password is table.

Q5 (5pts)

The hex message F3 was sent by Alice to Bob and was received along with its MAC 3A8E7235 (also in hex). The MAC was calculated using the mechanism described in figure 2.6a in the book. The encryption mechanism used was the ***simplified*** DES in ECB mode and key 1001100110. The hash function used was CRC32. Has this message been tampered with? Describe your steps clearly (use bullets to describe steps).

We must decrypt the encrypted MAC we received and compare it with a MAC we generate using the key and the original message. Thus:

* Decrypting MAC requires us to split the input into 8 bit parts; thus for key = 1001100110 and MAC = 3A8E7235 we have 4 parts to solve using the Decrypt SDES in which we must compute the binary representation of the hex values before feeding them in along with the key to generate the decrypted values.
  + Part 3A:
    - 3A: 00111010
    - Result: 11110110 = F6
  + Part 8E:
    - 8E: 10001110
    - Result: 10110110 = B6
  + Part 72:
    - 72: 01110010
    - Result: 01001100 = 4C
  + Part 35:
    - 35: 00110101
    - Result: 00101011 = 2B
  + Thus, the decrypted MAC = F6B64C2B
* We must now compute the CRC32 of F3 which is f6b64c2b using HashCalc in hex string mode.
* Finally, to determine if the message has been tampered, we simply compare the decrypted MAC and the computed hash to see if they are not the same. F6B64C2B is equal to F6B64C2B.

Thus, the message has not been tampered with.

**Extra Credit (5pts)**

Can you find any two inputs that would hash to the same value using CRC32?

Yes, according to <http://preshing.com/20110504/hash-collision-probabilities/>, CRC32 will produce the same hashes for **plumless** and **buckeroo** which is 0x4ddb0c25 as confirmed with HashCalc.