Assignment 5

1. **Suppose that a secure cryptographic hash function generates hash values that are n bits in length. Explain how a brute force attack could be implemented. What is the expected work factor?**

To find two messages that will hash into the same hash value of n bits in length, we would need to perform a brute force attack. A brute force attack could be implemented by hashing random or sequential messages until we find a pair of different message that has the same hash value. The expected work factor of a brute force attack to break a secure hash function is about 2n/2 to find a pair of messages with the same hash value.

1. **Suppose that h is a secure hash that generates an n-bit hash value.**
   1. What is the expected number of hashes that must be computed to find one collision?

The expected number of hashes that must be computer to find one collision is around 2n/2 hashes

* 1. What is the expected number of hashes that must be computed to find 10 collisions? That is, what is the expected number of hashes that must be computed to find pairs (xi, Zi) with h(xi) = h(zi), for i = 0, 1, 2, . . . , 9?

The expected number of hashes to find 10 collisions is around , in part C, we will explain how this expected number is found.

* 1. What is the expected number of hashes that must be computed to find m collisions?

The expected number of hashes to be computed to find m collisions is around, this formula is derived by assuming that there are number of input/hashes and there are pairs of hashes/input which equals to 2nm, since the probability of 2 different input to collide is equals to 2-n, we perform 2nm \* 2-n which equals to m, therefore the number of hashes computed to find m collisions is . (Source found at bottom reference section)

1. **A program implementing your crafty author's Bobcat hash algorithm can be found on the textbook website. This hash is essentially a scaled down version of Tiger-whereas the Tiger hash produces a 192-bit output (three 64-bit words), the Bobcat hash produces a 48-bit value (three 16-bit words).**
   1. Find a collision for the 12-bit version of Bobcat, where you truncate the 48-bit hash value to obtain a 12-bit hash. How many hashes did you compute before you found your first 12-bit collision?

To find the first 12-bit collision for the first time, it took 69 hashes. Since we are trying random input for collision, the number changes every time and overall it takes between 60 to 70 hashes to find the first collision. (code and output.txt included)

* 1. Find a collision for the full 48-bit Bobcat hash.

Due to program limitations, we were not able to find a collision for 48bit hash, because to find a collision we would be doing around 2^24 hashes which is around 16 million, This number is not supported by the Array in C program therefore the program is not working. But in order to get a hash collision, we would need to hash around 16 to 17 million times.

1. **Recall the online bid method discussed in Section 5.8.1.**
   1. What property or properties of a secure hash function h does this scheme rely on to prevent cheating?

This scheme relies on the one-way property and on both weak and strong collision resistance in order to prevent cheating. Since hash function are one way, other bidders will not be able to know the real message behind the hash value, and since the hash function is collision resistant, bidders would not be able to change their bid after submitting their hash value.

* 1. Suppose that Charlie is certain that Alice and Bob will both submit bids between $10,000 and $20,000. Describe a forward search attack that Charlie can use to determine Alice's bid and Bob's bid from their respective hash values.

Since Charlie knows that Alice and Bob will submit bids between $10,000 and $20,000, Charlie can easily perform a forward search attack by hashing every number from $10,000 to $20,000 and compare each of the hash values to the hash they submitted.

* 1. Is the attack in part b a practical security concern?

Yes, the attack in Part b is a practical security concern, since in an auction bid, the range of other people’s bid can be easily guessed and then use the forward search attack to determine the real value of their bids.

* 1. How can the bidding procedure be modified to prevent a forward search such as that in part b?

To prevent a forward search attack in this bidding procedure, we can ensure that the message is long enough by padding it with random bits.

1. **Alice's computer needs to have access to a symmetric key KA. Consider the following two methods for deriving and storing the key KA.**

**(i) The key is generated as KA = h(Alice's password). The key is not stored on Alice's computer. Instead, whenever KA is required, Alice enters her password and the key is generated.**

**(ii) The key KA is initially generated at random, and it is then stored as E(KA, K ), where K = h(Alice's password). Whenever KA is required, Alice enters her password, which is hashed to generate K and K is then used to decrypt the key KA.**

**Give one significant advantage of method (i) as compared to (ii), and one significant advantage of (ii) as compared to (i).**

The advantage of (i) over (ii) is that KA is generated every time it is needed which means that intruders cannot get the key even if they lay their hands in Alice’s computer, KA could also be dynamic, since it changes whenever Alice changes her password.

The advantage of (ii) over (i) is that KA cannot be easily obtained by guessing Alice’s password, in (i) if Alice has a weak password, KA could be easily obtained, but in (ii) KA is randomly generated and has one more layer of protection by being encrypted with K, Alice’s password hash.

1. **Suppose that Sally (a server) needs access to a symmetric key for user Alice and another symmetric key for Bob and another symmetric key for Charlie. Then Sally could generate symmetric keys KA, KB, and KC and store these in a database. An alternative is key diversification, where Sally generates and stores a single key KS. Then Sally generates the key KA as needed by computing KA= h(Alice, KS), with keys KB and KC generate in a similar manner. Give one significant advantage and one significant disadvantage of key diversification as compared to storing keys in a database.**

One advantage of key diversification is that only one key is needed; so it can generate a symmetric key for any user it needs, as well as it is storage efficient when dealing with large amount of users.

One disadvantage of key diversification is that, when the key KS is compromised, the keys of all users will be compromised as well, but by storing keys in a database, if one key is compromised, only that key needs to be replaced.

1. **The MD5 hash is considered broken, since collisions have been found and, in fact, a collision can be constructed in a few seconds on a PC. Find all bit positions where the following two messages differ. Verify that the MD5 hashes of these two messages are the same.**

Message difference:

At line 00000010 4th hex value, or the 20th byte of the message, in the first message the hex is 0x87 and in the second message the hex is 0x07

At line 00000020 14th hex value, or the 46th byte of the message, in the first message the hex is 0x71 and in the second message the hex is 0xF1

At line 00000030 12th hex value, or the 60th byte of the message, in the first message the hex is 0xF2 and in the second message the hex is 0x72

At line 00000050 4th hex value, or the 84th byte of the message, in the first message the hex is 0xC7 and in the second message the hex is 0x47

At the line 00000060 14th hex value, or the 110th byte of the message, in the first message the hex is 0xCC and in the second message the hex is 0x4c

At the line 00000070 12th hex value, or the 124th byte of the message, in the first message the hex is 0xD8 and in the second message the hex is 0x58

MD5 verification: both the first and the second messages hash into the same MD5 hash value of A4C0D35C95A63A805915367DCFE6B751

1. **The MD5 collision in Problem 25 is said to be meaningless since the two messages appear to be random bits, that is, they do not carry any meaning. Currently, it is not possible to generate a meaningful collision using the MD5 collision attack. For this reason, it is sometimes claimed that MD5 collisions are not a significant security threat. The goal of this problem is convince you otherwise. Obtain the file MD5\_collision.zip from the textbook website and unzip the folder to obtain the two Postscript files, rec2.ps and auth2.ps.**
2. What message is displayed when you view rec2.ps in a Postscript viewer? What message is displayed when you view auth2.ps in a Postscript viewer?

**Message in rec2.ps:**

To Whom it May Concern:

Tom Austin and Ying Zhang have demonstrated decent programming ability. They should do OK in any programming position, provided that the work is not too complex, and that the position does not require any independent thought or initiative.

However, I think they like to steal office supplies, so I would keep a close eye on them. Also, their basic hygiene is somewhat lacking so I would recommend that you have them telecommute.

Sincerely,

Mark Stamp

**Message in auth2.ps:**

To Bank of America:

Tom Austin and Ying Zhang are authorized access to all of my account information and may make withdrawals or deposits.

Sincerely,

Mark Stamp

1. What is the MD5 hash of rec2.ps? What is the MD5 hash of auth2.ps? Why is this a security problem? Give a specific attack that Trudy can easily conduct in this particular case. Hint: Consider a digital signature.

The MD5 hash of rec2.ps is C321325ACFF48137D62844E481AB01C5

The MD5 hash of auth2.ps is C321325ACFF48137D62844E481AB01C5

This is a security concern because two different message is with very different meaning has the same MD5 hash. An attack that Trudy can easily conduct in this particular case is the Middle Man Attack, by intercepting the rec2.ps from the sender and sending the auth2.ps to the receiver, and since the message hash into the same hash value as the one signed by the sender, the receiver cannot detect any tampering.

1. Modify rec2.ps and auth2.ps so that they display different messages than they currently do, but they hash to the same value. What are the resulting hash values?

I have modified rec2.ps and auth2.ps so that they display different messages, their md5 is 10B9FE80D373C0F4FDFD21B249E3992B

1. Since it is not possible to generate a meaningful MD5 collision, how is it possible for two (meaningful) messages to have the same MD5 hash value? Hint: Postscript has a conditional statement of the form

(X)(Y)eq{T0}{T1}ifelse

where T0 is displayed if the text X is identical to Y and T1 is displayed otherwise.

For two messages to have the same MD5, both messages would need to be really similar and any bit of difference would need to trigger which message is hidden and which one is shown. An example is the original rec2.ps and auth2.ps, if viewed in hex editor, the difference between the two messages is only about 4 different bytes.

1. **Suppose that you receive an email from someone claiming to be Alice, and the email includes a digital certificate that contains**

**M = (“Alice”, Alice’s public key) and [h(M)]CA**

**where CA is a certificate authority.**

1. How do you verify the signature? Be precise.

To verify the signature, we have use message M to calculate h(m) and then use the CA’s public key to decrypt [h(M)]CA by doing {[h(M)]CA}CA and if both results are the same then the signature has been verified.

1. Why do you need to bother to verify the signature?

We need to bother to verify the signature because that is the only way we can know who can read the messages we encrypt with the public key.

1. Suppose that you trust the CA who signed the certificate. Then, after verifying the signature, you will assume that only Alice possesses the private key that corresponds to the public key contained in the certificate. Assuming that Alice's private key has not been compromised, why is this valid assumption?

This is a valid assumption because [h(M)]CA has been encrypted with the a trusted CA’s private key and from that we can know that if the public key is signed by CA it means that the CA is certifying that the public key belongs to Alice and only the Alice has the private key (assuming Alice’s key has not been compromised).

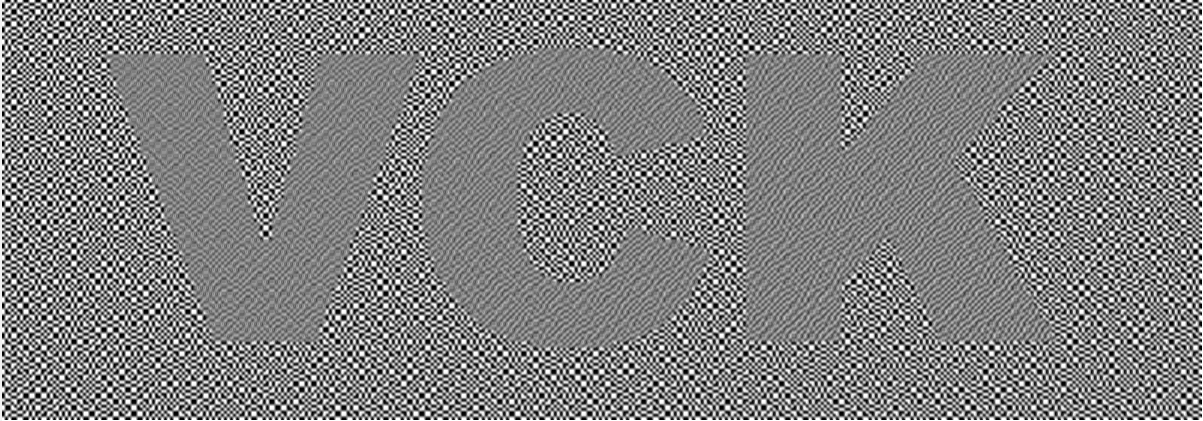
1. Assuming that you trust the CA who signed the certificate, after verifying the signature, what do you know about the identity of the sender of the certificate?

After verifying the signature, the sender of the certificate still remains unknown because M is public as well as [h(M)]CA, therefore anyone could of sent it to us and we cannot assume anything about the sender.

1. **Obtain the file visual.zip from the textbook website and extract the files.**
   1. Open the file visual.html in your favorite browser and carefully overlay the two shares. What image do you see?

Due to the fact that Visual.html is based on old java version, it cannot be opened. Instead I used the example provide by the professor on the website <http://www.cl.cam.ac.uk/~fms27/vck/>

By overlaying the two shares, we can see the letters VCK which is the hidden message between the two shares, as shown in the image below:



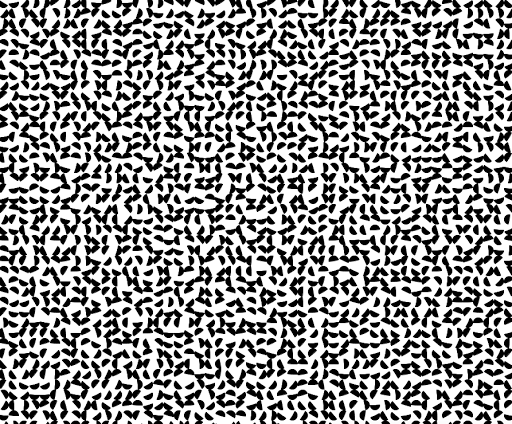
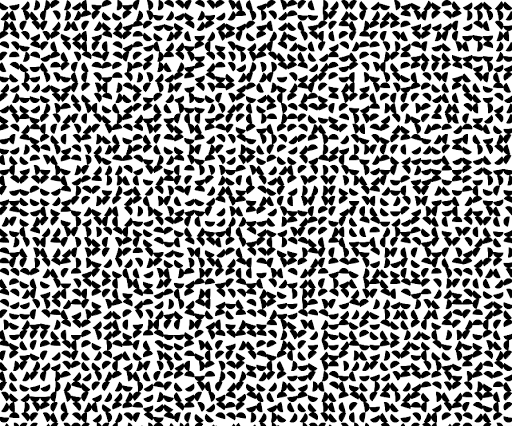
* 1. Use the program with a different image file to create shares. Note that the image must be a gif file. Give a screen snapshot showing the original image, the shares, and the overlaid shares.

By using the VCK program downloaded from the website above, I used the image included of “guido”

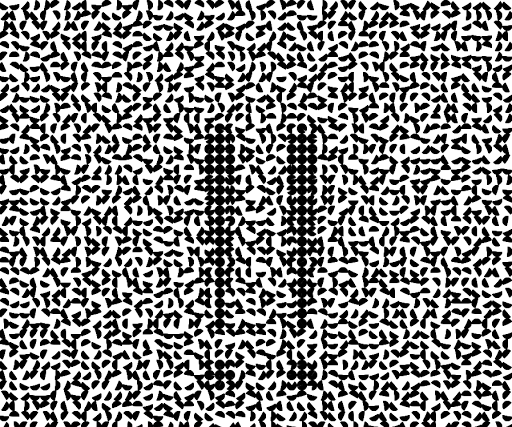
Original:

http://s3.amazonaws.com/static.graphemica.com/glyphs/i50s/000/012/472/original/203C-50x50.png?1275331240

Shares:



Overlay:



1. **Obtain the file stego.zip from the textbook website.**
   1. Use the program stegoRead to extract the hidden file contained in aliceStego.bmp

After extracting the hidden file contained in aliceStego.bmp, I found the hidden file is Alice in Wonderland book in pdf format

* 1. Use the programs to insert another file into a different (uncompressed) image file and extract the information.

I used an image of gundam, and hid my resume inside of that image. After getting the image with hidden file, I extracted the information and got back the same file, my resume.

* 1. Provide screen snapshots of the image file from part b, both with and without the hidden information.

Original: With hidden file:

1. **This problem deals with the uses of random numbers in cryptography.**
   1. Where are random numbers used in symmetric key cryptography?

Random numbers are used in symmetric keys when it requires an initialization vector, as well as different symmetric encryption method uses random numbers as the key itself like the one time pad.

* 1. Where are random numbers used in RSA and Diffie-Hellman?

In RSA the prime numbers are random numbers and in the Diffie-Hellman key exchange, secret numbers are random numbers.

REFERENCES  
Problem 5 b and c: <http://crypto.stackexchange.com/questions/22551/how-much-work-is-required-to-detect-multiple-collisions-for-a-hash-function>

Classmates: Jay Patel and Jeffrey Su