**3010 Computer Security Semester 1**

**Week 11 Tutorial**

**Authentication/Password/Crypto Tutorial**

1. Suppose that a certain email system uses hash of time when document is encrypted & emailed together. Example: time 20220203073000 means(3 February 2022 7:30:00)

* Alice encryption software works as follow:
* It uses 256-bit AES-encryption.
* Key used is 256-bit key, namely SHA256(Date & Time of email).
* When Alice use this encrypted email document and want to send to Bob, system looks at the time, then perform SHA256(Date & Time of email) and use this as 256-bit key for AES.
* Bob downloaded document into his pc. One day his pc is hacked and this document is in hands of hacker. Assume that hacker knows encryption algorithm, and encryption method of key being SHA256(Date & Time of email), **but not the date and time when document is sent.**
* (a) Explore if hacker has a chance to read this document.
* (b) What are the key lessons we can learn from this implementation of very strong algorithm 256-bit AES?

a) No the hacker cannot read the document. This is as SHA256 is a cryptographic hash that takes an input and produces a unique hash output which cannot be reverse engineered to get the input data from the hash value. (wrong)

Assuming his document was written in 2022, with the format measured up to seconds, we only need to compute how many seconds there are in 2022,hash of time when document is encrypted

That will be the number of possible AES keys

This number is very small compared to 2^256. 1 year=365days=365\*24h = 365\*24\*60mins = 365\*60\*60\*24s = 31536000s = 2^25 only

Can be cracked in 1 minute

b) Files of any length are hashed into fixed size file of 256 bits. The mapping of the bits is a many to one function where different files hash to the same value. Padding is done to the last block since messages do not always come in sizes of 256 bits. (wrong)

Strong algorithm alone is not good enough

Long key alone is also not good enough

Keys generated must be of as high entropy as possible AND cryptographically random

1. Why do we want to use slow hash for password hashing?

It is so that it takes longer for the attacker to reverse the hash function to get the password input given the hash output

Slow hash is to slow down the attacker’s bruteforce or dictionary attack

1. Explain why for CTR mode in block cipher encryption, your counter values must all be distinct to be secure.

In CTR mode, plaintext is encrypted by XORing with a stream generated using E, key k and counter CTR. If the counter values are not distinct, then Ek(CTR) would be the same for the whole encryption. So, the plaintext would XOR with the same key throughout. If the attacker finds the key, he can just XOR the cipher to get the plaintext.

If CTRi and CTRj is the same, then the XORing of the corresponding plaintext is EQUAL to the XORing of the associated ciphertext. Keys have disappeared because of XORing. Analogous to one time pad reuse.

1. If the hash reflected on the web page of download coincides with your computed hash, does it always mean the files have not been tampered with, assuming the hash used is a real secured hash?

No it does not always mean the files have not been tampered with, The attacker could have access to the server, perform Man in the Middle attacks through intercepting the download, compromise a Content Distributed Network, or perform phishing or spoofing.

The only thing that can go wrong is such an instance:

If the webpage has been hacked, the hacker can simply replace the software by malware, and then replace the original hash checksum with the new malware hash checksum. Thus very important for administrators to regularly check software and checksum have not been tampered with



(-1,1,0,0,0,0,1), (0,0,-1,1,1,-1,0), …

Let x = (x1 x2 … x7) Solve xA = (0 1 0 1)

Write the system down as a system of 4 equations with 7 unknowns. System is already in echelon form. Use gaussian elimination, express x4 in terms of 3 free variables x5 x6 x7: 8 possible choices

Then, similarly express x3 x2 x1 in terms of x5 x6 x7. Draw a table

System of equations:

X1 + x2 + x3 + x4 = 0

X2 + x3 + x4 + x5 = 1

X3 + x4 + x5 + x6 = 0

X4 + x5 + x6 + x7 = 1

There are eight solutions for (x1,…,x7): (1 1 1 1 0 0 0),(1 1 0 0 0 0 1),(1 0 1 0 0 1 0),(1 0 0 1 0 1 1),(0 1 1 0 1 0 0),(0 1 0 1 1 0 1),(0 0 1 1 1 1 0),(0 0 0 0 1 1 1)

What can we learn from this?

There can be a lot of preimages for x. This shows that the hash function is not easily reversible.

1. Why do we need 256-bit hash length instead of 128-bit length to pair with AES-128?

For a 256bit hash function, its security in terms of finding hash collisions is only at best 128 bit strength. That is why a 256bit hash is paired with the 128bit AES. Collision resistance is where is it computationally infeasible to find 2 files M != N with identical hashes

Hash space = 2^256

By birthday paradox, hash will repeat approximately after sqrt(s^256) = 2^128

So cracking 256 bit secure hash takes 128 bit complexity, same as AES128

1. Explain why a long codeword such as **hippopotamus** is not secure for use in Vignere cipher.

The keyword has many repeated letters which makes it vulnerable to frequency analysis attacks. The period of the keystream is also small considering the ‘popo’ part in hippopotamus.

Dictionary word

Attacker just needs to bruteforce each dictionary word as a possible keyword and do decryption. Since all operations are simple additions, even if dictionary has 10 million words, the bruteforced plaintext can be brtueforced out within minutes since PCs can crack millions of such cases within seconds, at most minutes. To flush out the correct plaintext, use some frequently occurred words such as “THE” and search for this keyword. It should appear often in the correct plaintext.

1. Keystream of {0,1} generated by pseudo-random number generators will be periodic. Secure keystream must necessarily have long period. ***Show by an example, that even with extremely long period, some keystreams may not be suitable for use to generate encryption keys.***

An example of this is Linear Feedback Shift Registers.

Initially we have s0,s1,s2 as the current state bits that are unknown and s3 as the next state bit

S3 is represented through s3 = s0 XOR s2

If the attacker intercepts the sequence 101010, equations can be set up based on the observed bits

S3 = s0 XOR s2

S4 = s1 XOR s3

S5 = s2 XOR s4

The system of equations can then be solved and the attacker can recover the internal state of the LFSR and predict future output bits

Let N be as large as you want

Consider keystream 11…100…0 (N 1s followed by N 0s, and repeat. Clearly long period, but totally predictable, so bad keystream.

So long period alone NOT ENOUGH to guarantee secure keystream