Paxson
Spring 2017

CS 161 Computer Security

Discussion 5

Week of February 27, 2017

Question 1 ***Activity: Cryptographic security levels*** **(20 min)**

Say Alice has a randomly-chosen symmetric key $S \in \{0,1\}^{128}$ (that is, a 128-bit key) that she uses to encrypt her messages to Bob.

Eve is very suspicious of these messages and would like to brute-force guess the key. She does this by getting a pair (M, C) where she knows that C is Alice's encryption of M. She keeps guessing keys k until $E_k(M) = C$.

(a) **Probability review.** How many attempts does Eve expect to have to try in order to guess Alice's key, if she guesses keys completely at random (with repetition)? What about if she guesses in order (without repetition)?

(b) Eve sits down at her computer and starts brute-forcing the key. If her computer can attempt 1 billion keys per second, how much time does Eve expect to wait?

How long of a time is this?

(c) Eve decides to enlist the help of her friend Ed, who works at the NSA and has access to a cluster of 1,000,000 servers[1] running in parallel that can each guess 10 billion keys per second.

Now how long will Eve be waiting? How much faster is this?

(d) Alice starts getting worried about Eve and decides to increase the key size to 256 bits. Bob claims this is pointless since the key is only twice as big as before, and so Eve needs only double as much time as before. Is he right?

(e) **Bonus.** The quantum computing *Grover's algorithm* lets you brute force a function using only $\mathcal{O}(N^{1/2})$ evaluations, instead of the $\mathcal{O}(N)$ required in classical computing.

If Eve gets a quantum computer, now how many attempts does Eve have to try for a 128 bit key? How much faster is this?

If we wanted to increase key size to combat this, how much of an increase do we need? Should we be concerned about possible future quantum computing attacks against symmetric-key cryptography?

[1]This is estimated to be around the number of servers that Google has. `https://what-if.xkcd.com/63/`

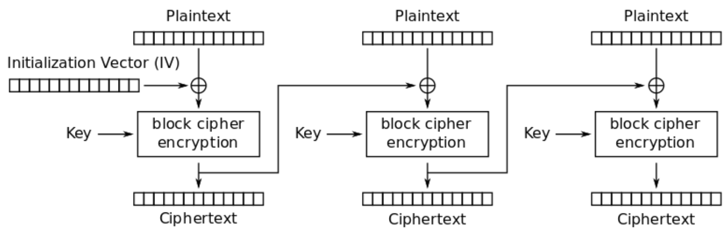
Question 2 ***PRNGs and Stream Ciphers*** **(10 min)**

(a) Pretend I have given you a pseudo-random number generator R. R is a function that takes a 128-bit seed s, an integer n, and an integer m, and outputs the n^{th} (inclusive) through m^{th} (exclusive) pseudo-random bits produced by the generator when it is seeded with seed s. Use R to make a secure symmetric-key encryption scheme. That is, define the key generation algorithm, the encryption algorithm, and the decryption algorithm.

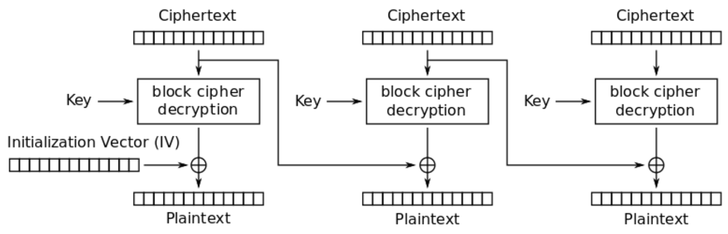
(b) Explain how using a block cipher in counter (CTR) mode is similar to the scenario described above.

Question 3 *Block cipher security and modes of operation* (15 min)

As a reminder, the cipher-block chaining (CBC) mode of operation works like this:



Cipher Block Chaining (CBC) mode encryption



Cipher Block Chaining (CBC) mode decryption

The output of the encryption is the ciphertext + the IV that was used.

(a) Does the initialization vector (IV) have to be independent of the plaintext? Why?

(b) Is a random IV enough? Imagine you picked IVs out of a list of random numbers, like *A Million Random Digits with 100,000 Normal Deviates* (RAND, 1955).

Say Alice encrypts the one-block long message m_1 with initialization vector IV_1 to get C_1 and encrypts m_2 using IV_2 to get C_2. She gives these to Mallory and challenges her to tell which C came from which m.

Mallory knows that Alice's next IV will be IV_3, and can ask Alice to encrypt messages for her (a *chosen plaintext attack*). Can Mallory distinguish the two ciphertexts?