Week of February 29, 2016

Question 1  Activity: Cryptographic security levels  (20 min)
This question is intended to clarify a common misunderstanding seen on the midterm, as well as introduce some important cryptographic principles.

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 $O(N^{1/2})$ evaluations, instead of the $O(N)$ 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/](https://what-if.xkcd.com/63/)
Question 2  One–time pads and the XOR operator  

You learned in lecture that the one–time pad is perfectly (information-theoretically) secure. But it has two major disadvantages: the first is that you need as much one–time pad as message data. The second is that the pad must never be used more than once.

Say Alice has encrypted two messages, \( m_1 \) and \( m_2 \), with the same one–time pad \( k \), producing ciphertexts \( m_1 \oplus k \) and \( m_2 \oplus k \).

(a) Assume \( k \) was chosen uniformly at random. Can an attacker Eve ever tell the difference between \( m_1 \oplus k \) and a completely random number \( r \) of the same length? What property of XOR does this use?

(b) Eve has intercepted both encryptions, \( m_1 \oplus k \) and \( m_2 \oplus k \) and knows that they were made using the same “one”–time pad \( k \). Now what can she tell about \( m_1 \) and \( m_2 \)?
Question 3  **Diffie–Hellman key exchange**  
(15 min)

Recall that in a Diffie-Hellman key exchange, there are values \(a\), \(b\), \(g\) and \(p\). Alice computes \(g^a \mod p\) and Bob computes \(g^b \mod p\).

(a) Which of these values are publicly known and which must be kept private?

(b) Eve can eavesdrop on everything sent between Alice and Bob, but can’t change anything. Alice and Bob run Diffie-Hellman and have agreed on a shared symmetric key \(K\). However, Bob accidentally sent his \(b\) to Alice in plain text. If Eve viewed all traffic since the beginning of the exchange, can she figure out what \(K\) is?

(c) Mallory can not only view all Alice—Bob communications but also intercept and modify it. Alice and Bob perform Diffie-Hellman to agree on a shared symmetric key \(K\). After the exchange, Bob gets the feeling something went wrong and calls Alice. He compares his value of \(K\) to Alice’s and realizes that they are different. Explain what Mallory has done and what she can now do.
Question 4  *Block cipher security and modes of operation*  
(15 min)

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

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

(a) Does the initialization vector (IV) have to be random? 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?