hw01

Daven Farnham

February 2017

1. Trade-offs

   (a) I think the whole idea of two-factor authentication is a good example of a trade-off where your accounts, such as an email account, are more secure if you have to provide both a password and, let's say, a code sent to your phone, but it's sometimes very inconvenient to always have to refer to your phone to sign in.

   Similarly, some password managers like 1password will create random passwords for you and then store them in an app. This means your passwords are more random and there's less likely to be any overlap between them, however, if you try to sign into an account and you don't have access to your password manager (because you lost your phone, for example) or you forget the master password to get into your password manager, you're essentially locked out of everything.

   (b) A lot of times when you forget your password, there's something like a 'security question' you can answer to recover it. This opens up people, especially famous people, to attacks since a security question might be something like 'what's your dog's name?' or 'what was your first car?' etc, essentially personal information that's easily googleable.

   Although this makes certain accounts more susceptible to attacks, for the vast majority of people these kinds of questions are a much more convenient way to recover a forgotten password, which happens frequently considering the number of passwords people need to remember. Although doing away with security questions might make certain accounts more secure (for famous people, for example) it would make things more inconvenient for everyone, many of whom wouldn't be vulnerable to the kind of attack described above.

2. Attacker Model

   (a) Posted to Piazza!

3. Entropy

   (a) 2^{256} possibilities, therefore, entropy = 256

   (b) Transformations of random variables will only ever preserve or decrease entropy. Coming from a true random source and then creating an 128-bit string gives 2^{128} possibilities, therefore, entropy = 128

   (c) Similar to above, transformations won't increase entropy. The initial user supplied password has only 40 bits, therefore entropy = 40.

   (d) There are a million microseconds in a second. If the key is broadcast within a 10 second window, I think this essentially means there are $2 * 10^7$ possible microseconds, meaning you need roughly 25 bits to represent all the values. Entropy, therefore, would be:

   $$log_2 20000000 = 24.2535$$

   (e) If the seed is biased then the entropy will be lower than it was originally in part b. Think about an extreme case, where the bias for a bit is 100%. In this case, it's clear the entropy of the system is lower. This reasoning holds for less extreme biased systems.

4. Performance

   (a) i. $2 * \frac{1 \text{ second}}{114 \text{ Mb}} * \frac{1000 \text{ Mb}}{1 \text{ Gb}} = 17.54$ seconds

   ii. $\frac{.16 \text{ milliseconds}}{1 \text{ operation}} * \frac{1 \text{ second}}{10^3 \text{ millisecond}} * \frac{\text{operation}}{256 \text{ bytes}} * \frac{10^9 \text{ bytes}}{1 \text{ Gb}} + \frac{6.08 \text{ milliseconds}}{1 \text{ operation}} * \frac{1 \text{ second}}{10^3 \text{ millisecond}} * \frac{\text{operation}}{256 \text{ bytes}} * \frac{10^9 \text{ bytes}}{1 \text{ Gb}} = 24375$ seconds

   (b) So the ratio of public key cryptography to symmetric key cryptography is roughly $\frac{24375}{17.54} \approx 1389$

5. Hybrid

   (a) Since the messages are encrypted, you don't know what they say and it's not feasible to decrypt them (we talked in class about how it's not feasible to break certain kinds of reliable encryption methods, such as RSA.)

   (b) What the attacker could do is when Alice and Bob are exchanging symmetric keys, Mallory could instead, using the public keys available, encode her own symmetric key and pass that along to Alice and Bob. Now, Alice will have two keys, K_A and K_C, while Bob will have K_B and K_C.

   Following the protocol, when Alice wants to send a message to Bob, she'll encrypt it with K_C (which she believes is K_B). This means

Mallory will be able to easily decrypt and read the message. Likewise for when Bob tries to send a message.

(c) The problem now, though, is Bob will try to decrypt with K_B the message coming from Mallory in the middle (since he assumes Alice used K_B to encrypt the message). This message, however, was actually encrypted with a key provided by Mallory — in the above case K_C. Bob trying to decrypt this message from Mallory will then, most likely, result in gibberish meaning it's likely Alice and Bob will quickly become aware of the attack.

What Alice and Bob could do to help prevent this kind of attack would be to send messages using their generated symmetric key instead of the other person's. So, in the above example, have Alice send her message encrypted with K_A. Now, even if Mallory inserted her own key K_C like in the above attack, she wouldn't be able to see the message.