Massachusetts Institute of Technology
6.857: Network and Computer Security
Professors Ronald L. Rivest and Yael Tauman Kalai

Handout 2
Feburary 5, 2020
Due: Feburary 18, 2020

Problem Set 1

This problem set is due on *Tuesday, February 18, 2020* at **11:59 PM**. Please note our late submission penalty policy in the course information handout. Please submit your problem set, in PDF format, on Gradescope. *Each problem should be in a separate PDF.* Have **one and only one group member** submit the finished problem writeups. Please title each PDF with the Kerberos of your group members as well as the problem set number and problem number (i.e. *kerberos1_kerberos2_kerberos3_pset1_problem1.pdf*).

You are to work on this problem set in groups. For problem sets 1, 2, and 3, we will randomly assign the groups for the problem set. After problem set 3, you are to work on the following problem sets with groups of your choosing of size three or four. If you need help finding a group, try posting on Piazza or email `6.857-tas@mit.edu`. You don't have to tell us your group members, just make sure you indicate them on Gradescope. Be sure that all group members can explain the solutions. See Handout 1 (*Course Information*) for our policy on collaboration.

Homework must be submitted electronically! Each problem answer must be provided as a separate pdf. Mark the top of each page with your group member names, the course number (6.857), the problem set number and question, and the date. We have provided templates for LaTeX and Microsoft Word on the course website (see the *Resources* page).

Grading: All problems are worth 10 points.

With the authors' permission, we may distribute our favorite solution to each problem as the "official" solution—this is your chance to become famous! If you do not wish for your homework to be used as an official solution, or if you wish that it only be used anonymously, please note this in your profile on your homework submission.

Problem 1-1. Security Policy for Video Conferencing Platforms

Video and audio conferencing platforms such as Zoom, Skype, and WebEx are extremely common among businesses in 2020.

Write a set of basic functionalities that these platforms should have, and then describe an ideal security policy for these functionalities. Note that these are large pieces of software with many diverse functionalities - focus mainly on the following 3: video chats, screen sharing, and recordings.

The policies you come up with should address each of the security goals discussed in class, though focus on the one(s) that are most relevant for video and audio. Given the time constraints and the complexity of the problem, we expect your solutions to be less than comprehensive. That being said, keep in mind that an adversarial party can be either inside a private call or outside a private call and try to cause some undesirable result.

(This problem is a bit open-ended, but should give you excellent practice in writing a security policy. We have included sample solutions from similar questions in previous years on the course website under the 'Students Only' section.)

Problem 1-2. One - Time Pad

The goal of this problem is to demonstrate that *one-time pad* can be insecure, if it is used more than once. In addition, a variation of one-time pad is explored.

In the usual one-time pad setting, a message $M = (m_1, \ldots, m_n)$ needs to be encrypted using a secret and random-looking pad $P = (p_1, \ldots, p_n)$. Each 8-bit message byte m_i is encrypted by x-oring it with the 8-bit pad byte p_i to obtain an 8-bit ciphertext byte c_i:

$$c_i = m_i \oplus p_i$$

where $\oplus$ is the x-or operator on eight-bit values.

Message bytes are encrypted using the standard UTF-8 encoding: `https://en.wikipedia.org/wiki/UTF-8` (which is equivalent to ASCII for the usual characters).

We show in lecture that *one-time pad* is informationally-theoretically secure, but can be insecure when the same pad is used more than once. For more information on *one-time pad*, the Wikipedia page on it (`https://en.wikipedia.org/wiki/One-time_pad`) is a good reference.

(a) For the following part we encrypted two 14-character English words with a "one-time pad". Decide whether they were encrypted with the same pad or with different pads. If they are different pads, then explain why they cannot be the same pad. If they are the same pad, then decrypt the ciphertexts.

d3 a4 0a 3e 63 c2 13 3c 41 17 4d 57 85 bb

d1 a3 07 26 72 c3 04 20 50 04 5c 50 89 ac

(b) Ben Bitdiddle, a student of 6.857, knows that *one-time pad* can be broken when the pad is used more than once. He decides to use a different scheme. In his scheme, the message m and the pad p are defined as above, while g is a randomly chosen permutation of bytes. This is represented by a 256-length array G, in which $g(i)$ equals $G[i]$, for all $i = 0, \ldots, 255$. Array G is public and can be found in **gbox.txt**. Then, Ben creates a ciphertext $C = (c_1, \ldots, c_n)$ by

$$c_i = g(m_i \oplus c_{i-1}) \oplus g(p_i \oplus c_{i-1})$$

where $c_0 = 0$. Argue that Ben's scheme is decryptable. Prove that the scheme is *one-time secure*, that is, an adversary who hears one ciphertext, but has no information about the pad, cannot learn anything about the message.

(c) Ben is confident that his scheme is secure, so he broke his favorite book review into 6 messages $\{M_1, \ldots, M_6\}$ and created 6 ciphertexts $\{C_1, \ldots, C_6\}$ using his scheme and the **same** pad. You can find his encrypted messages in the file **gotp.txt**. The review contains only normal letters and punctuation marks. All characters are represented as 8-bit bytes with the usual US-ASCII encoding (e.g."A" is encoded as 0x41)

What is Ben's favorite review? Provide code and explain how you found the answer. (If you can only obtain *part* of the message, give that.)

Problem 1-3. 2FA

Two-factor authentication (2FA) is a type of multi-factor authentication (MFA). It strengthens standard identity verification by requiring the user to *both* prove knowledge of some personal information (your password or other login credentials), and prove possession physical device (such as your phone). Read the following article about 2FA: `https://www.makeuseof.com/tag/two-factor-authentication-sms-apps/`, and answer the following questions.

(a) What are the security benefits of 2FA? Are there any reasons why the second authentication method needs to be via phone or other external physical device? What problems can arise if the second factor to authenticate to a web application was email?

(b) For 2FA smartphone applications, why are the keys allowed to be short six digit codes as opposed to longer/harder passwords? Why do the codes rotate, i.e. change over time? Are there any issues that can arise with using a longer fixed key instead of a short rotating key?

The article mentions several potential flaws with smartphone-based 2FA methods such as SMS and 2FA applications. In 2016, a draft guideline by the US National Institute of Standards and Technology (NIST) explicitly discouraged the use of SMS as a form of 2FA, but retracted this statement in the finalized version in 2017 (`https://blogs.sap.com/2017/07/06/rollback-the-united-states-nist-no-longer-recommends-deprecating-sms-for-2fa/`).

(c) What attacks work on SMS but would not work against other 2FA applications and why? Why would SMS not be deprecated given its vulnerability to certain attacks?

(d) How does U2F work (no need to go into very technical details)? Why is it more secure compared to other channels such as SMS? What attacks would still work against accounts that use U2F as their 2FA method?