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

Handout 2
Feburary 22, 2021
Due: March 9, 2021

Problem Set 1

This problem set is due on *Tuesday, March 9, 2021* 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-staff@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 Vaccination Certificates

With the advent of the COVID-19 vaccine, debate has ensued regarding how freely individuals with and without vaccines should be allowed to travel, eat in restaurants, and so on. One way to differentiate between those who have and have not received a vaccine might be to use a digital vaccination certificate. By presenting a COVID-19 vaccination certificate, an individual could then receive privileged access to places and services that would otherwise be restricted due to the risk of individuals becoming infected.

Write the desired **functionalities** of a digital vaccination certificate system and a **security policy** for it. You may assume the following:

- All players in the system have access to a mobile phone or other digital device that can allow them to receive, access, and present a vaccination certificate.
- All players in the system have some form of official ID that proves their identity.

Make sure to specify who the players are (users, vaccination authority, vaccine providers, verifiers, ...), what properties your system provides (such as privacy and authenticity), what properties your system does not provide, and any additional assumptions you make (in particular about potential adversaries). Hint: One way to guarantee certificate authenticity would be through a digital signature scheme. More info on digital signatures can be found in `https://en.wikipedia.org/wiki/Digital_signature`.

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 this application. Given the time constraints and the complexity of the problem, we expect your solutions to be less than comprehensive. That being said, **make sure to state any assumptions you make and clearly explain the guarantees and limitations of your security policy.**

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

In this problem, we will explore a generalization of the one-time pad. Let U be a set of *pad values* and let V be a set of *letter values* (e.g., all 8-bit values, all length-5 tuples of values mod 27, etc). Then, let $f : U \times V \to V$ be a *component encryption function*, which applies one character of a pad to one letter of the plaintext to compute one letter of the ciphertext. Further, f must satisfy the important property that, for any fixed pad p, the function $g(x) \coloneqq f(p, x)$ is a bijection.

For example, we can set V to be the set of all 8-bit strings and U to be all pairs of the form (q, r), where $q \in V$ and $r \in [0, 7]$. Then, the function $f((q, r), x) \coloneqq (q \oplus x) <<< r$ is a component encryption function. The notation $a <<< b$ means the rotation of a left by r bit positions.

We can then use a component encryption function to encrypt a message. Namely, for a message $M \coloneqq (m_1, \dots, m_n)$ and a pad $P \coloneqq (p_1, \dots, p_n)$, where $m_i \in V$ and $p_j \in U$, each letter of M is encrypted by applying f to it with the corresponding pad letter to generate a ciphertext $C = (f(p_1, m_1), \dots, f(p_n, m_n))$. Note that the pad P is computed with n independent uniform draws from U, and shared a priori between the sender and the receiver.

- **(a)** Show that any component encryption function f is decryptable. That is, given a ciphertext C and a pad P, we can find the (unique) plaintext M which, when encrypted with P, yields C.
- **(b)** Argue that information-theoretic confidentiality is implied by the condition that, for any letters $x, y \in V$, $\Pr_{p \leftarrow U}[f(p, x) = y] = \frac{1}{V}$, where the probability is taken over the uniform choice of p.
- **(c)** Suppose we have t ciphertexts $C_1, \dots, C_t$, each with n letters. Describe a method for determining whether two ciphertexts were created by encrypting (different) English texts using the *same* pad P and an arbitrary component encryption function f. Your solution should not make any further assumptions about f. The words are encoded using the standard UTF-8 encoding (which is equivalent to ASCII for the usual characters).

  Hint: the index of coincidence (`https://en.wikipedia.org/wiki/Index_of_coincidence`) may be useful.
- **(d)** The file **ciphertexts.txt** contains four ciphertexts. Two of these were encrypted using the same pad. Using your method from **(b)**, determine the two ciphertexts in question. Note that, as before, the component encryption function is arbitrary.

Problem 1-3. Supply Chains

In software development, a piece of software has a supply chain analogous to supply chains for manufactured goods such as cars, going through a structured series of steps to be brought to consumers. Read this article to get an overview of software supply chains:

`https://github.blog/2020-09-02-secure-your-software-supply-chain-and-protect-against-supply-chain-threats-github-blog/`

The article opens with the motivation of open source software, but software supply chains apply to the development of any shared software product.

One aspect of software supply chains relevant for security is ensuring that every change to the software as it is developed and reaches users was meant to occur. To do so, the software supply chain becomes an example of a chain of trust, which you should recall from 6.033 and certificate authorities. We will explore one example of this conceptually: Read pages 1-6 of the “in-toto Specification” document:

`https://github.com/in-toto/docs/blob/v0.9/in-toto-spec.pdf`

(a) What, in your words, is in-toto trying to solve or what gap is it trying to close? Be as concrete as possible; describe the features in-toto tries to provide in relation to that problem.

(b) How could the system fail to achieve its stated goals even in the case where all of these properties held? Try to provide a high-level solution to the failure(s) you identify.

You may have heard of the recent breach of many US Government computer systems via the defense contractor SolarWinds. The news broke on December 8, 2020, that FireEye (a cybersecurity company) had a number of its systems compromised. But this was only the beginning. The Treasury, Commerce, Justice, and State Departments also succumbed to the breach—as did Miscrosoft, Cisco, and Intel. The attack is believed to have been perpetrated by Russia's Foreign Intelligence Service (SRV, the successor to the Soviet KGB), which gained a foothold into software used by many of these entities by a compromised access point in the SolarWinds supply chain. To get a sense of the scale of the breaches, consider the following two quotes from the New York Times:

> While the Russians did not have the time to gain complete control over every network they hacked, they most certainly did gain it over hundreds of them. It will take years to know for certain which networks the Russians control and which ones they just occupy.[1]
>
> Nearly all Fortune 500 companies, including The New York Times, use SolarWinds products to monitor their networks. So does Los Alamos National Laboratory, where nuclear weapons are designed, and major defense contractors like Boeing, which declined on Monday to discuss the attack.[2]

SolarWinds issues guidance regarding its compromised products in a Security Advisory here:

`https://www.solarwinds.com/sa-overview/securityadvisory`

(c) Read the SolarWinds Security Advisory. For at ***least three*** of the properties listed on page 3 of the Specification document, describe a plausible way in which the properties could have alerted developers or clients that the SUNBURST product was compromised.

(d) Do some research on supply chain attacks to find another example of such an attack. Briefly describe it, and describe how the two properties not mentioned in part (c) could catch the compromise.

[1] `https://www.nytimes.com/2020/12/16/opinion/fireeye-solarwinds-russia-hack.html`

[2] `https://www.nytimes.com/2020/12/14/us/politics/russia-hack-nsa-homeland-security-pentagon.html`