Massachusetts Institute of Technology
6.857: Network and Computer Security
Professor Ronald L. Rivest

Handout 2
February 8, 2016
Due: February 22, 2016

Problem Set 1

This problem set is due on *Monday, February 22, 2016* 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 with your assigned group of three or four people. Please see the course website for a listing of groups for this problem set. If you have not been assigned a group, please email `6.857-tas@mit.edu`. 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.

Our department is collecting statistics on how much time students are spending on psets, etc. For each problem, please give your estimate of the number of person-hours your team spent on that problem.

Problem 1-1. Security Policy for IoT Automobiles

It's not uncommon to find that today's cars have anywhere between 30 and 100 computers on board. As more manufacturers continue to integrate Internet capabilities in their vehicles, this opens up a whole new realm of possible attack surfaces, and with particularly dangerous consequences if they are not mitigated. A 1.4 million vehicle recall was announced by Chrysler last fall due to software vulnerabilities that allowed malicious parties to take full control of critical systems such as steering and brakes.

Describe a security policy for the onboard computer systems in an IoT enabled automobile. Be sure to address the relevant principals, actions, and policies.

Try to make your policy reasonably complete. However, given time constraints and the complexity of the problem, we expect your solutions to be less than comprehensive. That being said, keep in mind that there are potentially many types of parties that may interact with a car other than the owner and the manufacturer. Do your best to consider as many relevant principles as you can.

If you can't find relevant material on automotive computer systems, invent new material as appropriate. Try to be as complete as you can, but emphasize the aspects that are specific to IoT enabled vehicles—in particular, what security goals are the most relevant for this class of automobile. What roles does this security policy encompass, and what actions should each principal be allowed to do?

(This problem is a bit open-ended, but should give you excellent practice in writing a security policy. Also, you may actually care about such security policies for designing systems used by large numbers of people—or if you use one yourself! We have included sample solutions from similar questions in previous years on the course website.)

Problem 1-2. Two-time pads

It is well known that re-using a "one-time pad" can be insecure. This problem explores this issue, with some variations.

In this problem all characters are represented as 8-bit bytes with the usual US-ASCII encoding (e.g."A" is encoded as 0x41). The bitwise exclusive-or of two bytes x and y is denoted $x \oplus y$.

Let $M = (m_1, m_2, \ldots, m_n)$ be a message, consisting of a sequence of n message bytes, to be encrypted. Let $P = (p_1, p_2, \ldots, p_n)$ denote a pad, consisting of a coresponding sequence of (randomly chosen) "pad bytes" (key bytes).

In the usual one-time pad, the sequence $C = (c_1, c_2, \ldots, c_n)$ of ciphertext bytes is obtained by xor-ing each message byte with the corresponding pad byte:

$$c_i = m_i \oplus p_i, \text{ for } i = 1 \ldots n \ .$$

When we talk about more than one message, we will denote the messages as $M_1, M_2, \ldots, M_k$ and the bytes of message M_j as m_{ji}, namely $M_j = (m_{j1}, \ldots, m_{jn})$; we'll use similar notation for the corresponding ciphertexts.

(a) Here are two 11-character English words encrypted 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.

```
99 ca d2 81 77 62 30 8c 2a e8 28
95 cd c6 80 6d 67 38 9b 22 f3 23
```

(b) Ben Bitdiddle decides to fix this problem by making sure that you can't just "cancel" pad bytes by xor-ing the ciphertext bytes.

In his scheme the key has two parts, $P = (p_1, p_2, \ldots, p_n)$ and $Q = (q_1, q_2, \ldots, q_n)$ where $p_i \neq 0$ for $i = 1, \ldots, n$. The ciphertext bytes $c_1, c_2, \ldots, c_n$ are obtained as follows:

$$c_i = p_i \cdot m_i + q_i$$

where all the operations above are in $GF(2^8)$. Note that addition in $GF(2^8)$ is just xor.

Prove that Ben's scheme is "two-time secure". More precisely, prove that for any position i, and any distinct ciphertexts c_i, c_i', choosing an appropriate p_i, q_i can yield any pair of distinct plaintexts m_i, m_i'. (Note that if the ciphertexts c_i, c_i' are equal, the plaintexts m_i, m_i' must also be equal.)

(c) Ben is confident that he can now re-use his pad, since there is no apparent way that one can "cancel" the effect of the pad on the message to obtain the ciphertext. For example, xor-ing ciphertexts doesn't seem to do anything useful for an adversary. So, he feels that he can now re-use a pad freely.

You are given the file **eightciphs.txt**, containing eight ciphertexts $C_1, C_2, \ldots, C_8$ produced by Ben, using the *same* pad P, Q. You know that these messages contain only a-z and space.

To facilitate this task, we provide Python code (**ffield.py**) to perform finite field operations.

Submit the messages and the pad, along with a careful explanation of how you found them, and any code you used to help find the messages. The most important part is the explanation.

Problem 1-3. NSA TAO Chief on Disrupting Nation State Hackers

This problem is related to the youtube video "USENIX Enigma 2016 - NSA TAO Chief on Disrupting Nation State Hackers" by Rob Joyce. The URL for the video is: https://www.youtube.com/watch?v=bDJb8WOJYdA

Watch the video (it's approximately 30 minutes long) and answer the following questions:

1. Mention a few security practices that networked systems should be following. Why are they effective?
2. Which of these practices do you think are the easiest/hardest to implement? Why?
3. Rob mentions "passing the hash" as an attack method. Read on this method and briefly explain how it works.