Technische Universität München (I7)
Dr. M. Luttenberger / S. Jaax

Winter term 2018/2019
2018/11/06

Solution

Cryptography – Homework 1

Discussed on 6/11/2018.

Exercise 1.1 Warmup

Consider a CPU with n-bit registers. The n-bit strings $\{0,1\}^n$ are naturally interpreted as the integers $\mathbb{Z}_{2^n} = \{0,1,2,\dots,2^n-1\}$ using e.g. either "most significant bit first" or "least significant bit first" interpretation (it is not important which one is used). Addition and multiplication on $\mathbb{Z}_{2^n}$ are defined as usual modulo 2^n, e.g. $1+(2^n-1)=0$ and $2\cdot 2^{n-1}=0$ ("overflow"). We define some sets of functions over $\mathbb{Z}_{2^n}$:

- $M_n := \{f\colon \mathbb{Z}_{2^n} \to \mathbb{Z}_{2^n}\}$ the set of all maps from $\mathbb{Z}_{2^n}$ to $\mathbb{Z}_{2^n}$.
- $P_n := \{f \in M_n \mid f_n \text{is a bijection}\}$ the set of all bijections/permutations.
- $A_n = \{f \in M_n \mid \exists a,b \in \mathbb{Z}_{2^n} \forall x \in \mathbb{Z}_{2^n}\colon f(x) = a\cdot x + b\}$ the set of "affine" functions.

The (mono-alphabetic) substitution cipher generalizes to $\mathbb{Z}_{2^n}$ by taking a subset of P_n as key space with encryption defined by

$$\mathsf{Enc}_f(m_1||m_2||\dots||m_l) = f(m_1)||f(m_2)||\dots||f(m_l) \text{ and } \mathsf{Dec}_f(c_1||c_2||\dots||c_l) := f^{-1}(c_1)||f^{-1}(c_2)||\dots f^{-1}(c_l)$$

for $f \in P_n$ and $m_i, c_i \in \mathbb{Z}_{2^n}$.

(a) Compute $|M_n|$, $|P_n|$, $|A_n|$, and $|A_n \cap P_n|$.

Hint: Why does it suffice to discuss whether $f(x) = ax + b$ is injective or not?

(b) How would you store and compute $f(\cdot)$ and $f^{-1}(\cdot)$ for $f \in P_n$ resp. $f \in P_n \cap A_n$?

(c) Show that the substitution cipher is perfectly secret if (1) we restrict the message space to $\mathbb{Z}_{2^n}$ and (2) we choose the key uniformly at random from P_n resp. from $A_n \cap P_n$.

(d) Why is the previous result no contradiction to the fact that frequency analysis can be used to break the classical mono-alphabetic substitution cipher?

Solution:

- $|M_n| = (2^n)^{2^n} = 2^{n2^n}$ – for each of the 2^n elements in the pre-image we can choose any of the 2^n possible images.
- $|P_n| = (2^n)!$ – for the i-th preimage we can choose from the $2^n - i + 1$ so far unused images.

  Remark: For comparison using Stirling's approximation of $n!$ (see wikipedia)

  $$\frac{|P_n|}{|M_n|} \approx \frac{\sqrt{2\pi 2^n}\left(\frac{2^n}{e}\right)^{2^n}}{2^{n2^n}} = \sqrt{2\pi}\frac{2^{n/2}}{e^{2^n}}$$

  So the fraction of permutations is negligible w.r.t. the total number of maps from $\mathbb{Z}_{2^n}$ to $\mathbb{Z}_{2^n}$.
- Regarding $|A_n|$:

  For $a,b \in \mathbb{Z}_{2^n}$ let $f_{a,b}(x) := ax + b \pmod{2^n}$.

  We immediately get $|A_n| \le 2^n \cdot 2^n = 2^{2n}$ as there are at most 2^n choices for both a and b.

  Assume $(a,b) \neq (c,d)$ but $f_{a,b}(x) = f_{c,d}(x)$ for all $x \in \mathbb{Z}_{2^n}$. Then in particular:

  - For $x = 0$: $b = f_{a,b}(0) = f_{c,d}(x) = d \pmod{2^n}$, i.e. $b = d$ as $b, d \in \mathbb{Z}_{2^n}$.
  - For $x = 1$: $a + b = f_{a,b}(1) = f_{c,d}(1) = c + d \pmod{2^n}$. As $b = d$: $a = c \pmod{2^n}$. And as $a, c \in \mathbb{Z}_{2^n}$ also $a = c$.

  So we get the contradiction $(a,b) = (c,d)$.

  Hence, two different pairs define to different functions. Thus: $|A_n| = 2^{2n}$.

- Regarding $|A_n \cap P_n|$:

  Recall: a map $g\colon A \to A$ with $|A|$ finite is bijective iff it is injective. (This is not true if A is infinite, e.g. consider $g\colon \mathbb{N} \to \mathbb{N}\colon x \mapsto 2x$.)

  So, assume $x, y \in \mathbb{Z}_{2^n}$ with $f_{a,b}(x) = f_{a,b}(y)$, i.e. $ax = ay \pmod{2^n}$ or equivalently $a(x-y) = 0 \pmod{2^n}$. The function $f_{a,b}(x)$ is thus injective if the equation $az = 0 \pmod{2^n}$ has the unique solution $z = 0$. But this only holds iff a is odd:

  - if a is even i.e. $a = 2a'$, then e.g. $2^{n-1}a = 2^n a' = 0 \pmod{2^n}$.
  - if a is odd, then 2^n cannot divide a; so for $az = 0 \pmod{2^n}$ to hold, 2^n has to divide z, which means that $z = 0 \pmod{2^n}$. As $z \in \mathbb{Z}_{2^n}$, this means $z = 0$.

  So, for $f_{a,b}$ to be a bijection, we must restrict our choice of a to the odd numbers in $\mathbb{Z}_{2^n}$. Hence, $|A_n \cap P_n| = 2^{n-1} \cdot 2^n = 2^{2n-1}$.

- Regarding (b):

  Let Γ be the finite alphabet of a programming language like C or Java (or a Turing machine).

  Let s be the length of a program computing one of the functions of P_n.

  There are at most $|\Gamma|^s = 2^{O(s)}$ many such programs (or Turing machines) of a given length, while there are

  $$(2n)! \approx \sqrt{2\pi} 2^{n/2} \left(\frac{2^n}{e}\right)^{2^n} = \sqrt{2\pi} 2^{n/2 + (n - \log_2 e) 2^n}$$

  many functions in P_n. So in general, we have to use programs of length $s \geq 2^n$ if we want to use any permutation $f \in P_n$ as a key for the mono-alphabetic substitution cipher. That is, in general we cannot do much better than to simply store f as a table, and look up the image resp. pre-image.

  For comparison: if we restrict the key space to $A_n \cap P_n$, we only need programs of size $O(n)$. The problem with $A_n \cap P_n$ is that given two "encryptions" $f_{a,b}(x)$ and $f_{a,b}(y)$ of two known plaintexts x, y, we can recover a, b by solving the linear equation system. Secure block ciphers therefore have to strike the fine balance between choosing a sufficiently hard to analyze/break subset of M_n and choosing a subset of M_n which can easily be computed.

- Regarding (c+d):

  Here, the original definition of perfect secrecy is most helpful:

  The ES is perfectly secret iff

  $$\forall m_0, m_1, c \in \mathbb{Z}_{2^n} : \Pr_{k:\xleftarrow{r}\mathsf{Gen}()}[\mathsf{Enc}_k(m_0) = c] = \Pr_{k:\xleftarrow{r}\mathsf{Gen}()}[\mathsf{Enc}_k(m_1) = c]$$

  - Case: $k \xleftarrow{u} P_n$

    Because of uniform distribution, we have to count how many permutations map m_0 resp. m_1 on c.

    In both cases there are $(2^n - 1)!$ many such maps, as we only restrict the choice of the image of m_0 resp. m_1 to c. So, both probabilities are $\frac{(2^n-1)!}{(2^n)!} = 2^{-n}$ – just as in the case of the one-time pad.

  - Case: $k \xleftarrow{u} A_n \cap P_n$

    We have $f_{a,b}(m_0) = c$ iff $b = c - am_0 \pmod{2^n}$. So, we can choose a, but then b is determined. So, the probabilities are in both cases $\frac{2^{n-1}}{2^{2n-1}} = 2^{-n}$ again.

  So both of them are perfectly secret – the point of course is that perfect secrecy only considers the case of a single encryption and that frequency analysis also depends on the size of the underlying alphabet/group (compare $\{A, B, \ldots, Z\}$ to $\{0, 1, 2, \ldots, 2^{128}\}$.)

Exercise 1.2 Generating large odd random numbers

Consider the following algorithm $\mathcal{A}$

- *Input*: Natural number $n > 1$.
- *Output*: A random odd number in $[2^{n-1}, 2^n - 1]$.
- *Algorithm*:
  (a) Set $x_0 = 1$, $x_{n-1} = 1$.
  (b) For i in 1 to $n - 2$: Set $x_i = \text{fair\_coin}()$.
  (c) Let x be the integer represented by the bit string $x_0 x_1 \ldots x_{n-1}$ with the least-significant bit first x_0.

(a) Determine the distribution of the output of $\mathcal{A}(n)$.

(b) What is (a lower bound on) the probability that the output number is a prime?

Hint: Use that within $[2^{n-1}, 2^n - 1]$ there are at least $\frac{2^{n-1}}{n}$ primes if n is sufficiently large ($n \geq 20$ suffices).

Solution:

(a) Let $A_n := \{x \in [2^{n-1}, 2^n - 1] : x \text{ odd}\}$. Then $|A_n| = 2^{n-2}$.

Claim: The random variable $\mathcal{A}(n)$ is uniformly distributed over A_n.

Proof: Take an arbitrary but fixed $b \in A_n$. $b = 1b_1 \dots b_{n-2}1$ in binary representation.

$$\Pr[\mathcal{A}(n) = x] = \Pr[x_i = b_i \ (i \in \{1, \dots, (n-2)\})] =$$
$$\overset{x_i \text{ independent}}{=} \prod_{i=1}^{n-2} \frac{1}{2} = 2^{-n+2} = \frac{1}{|A_n|}.$$

(b) Let P_n denote the set of prime numbers in A_n. Using the hint we obtain:

$$\Pr[\mathcal{A}(n) \text{ prime}] = \frac{|P_n|}{|A_n|} \geq \frac{2}{n}$$

Exercise 1.3 **Perfect Secrecy**

(a) Is the following statement true? Every encryption scheme for which the size of the key space equals the size of the message space, and for which the key is chosen uniformly from the key space, is perfectly secret.

(b) Show that every encryption scheme can be transformed into an equivalent encryption scheme defined over a fixed-length key space. More precisely, show that from any encryption scheme $\mathcal{E} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ we can build an ES $\mathcal{E}' = (\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$ such that

1. Gen' simply generates a "truly random" (short for "chosen uniformly at random") key $k' \overset{u}{\in} \{0,1\}^T$ for some sufficiently large T. (T should bound the running time of Gen.)
2. The message and ciphertext spaces of $\mathcal{E}$ and $\mathcal{E}'$ are the same.
3. For every message m and every ciphertext c we have $\Pr_{k:\overset{r}{=}\mathsf{Gen}()}[\mathsf{Enc}_k(m) = c] = \Pr_{k':\overset{r}{=}\mathsf{Gen}'()}\left[\mathsf{Enc}'_{k'}(m) = c\right]$, i.e. the probabilitiy to encrypt m to c is in both ES the same when the key is generated by Gen resp. Gen'.
4. The running times of Enc' and Dec' are bounded by the running times of $\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}$.

Solution:

(a) Consider an encryption scheme $\mathcal{E} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with key space $\mathcal{K}$, plaintext space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ such that $\mathcal{K} = \mathcal{M} = \mathcal{C}$ and $\mathsf{Enc}_k(m) = m$ for every $m \in \mathcal{M}$ and every $k \in \mathcal{K}$, i.e., we do not use the key k at all for encrypting. Then Eve can win the eavesdropping game with probability 1: she takes two different messages $m_0 \neq m_1$, receives the ciphertext c and then outputs i with $c = m_i$.

(b) The idea is that we can transform Gen (which takes *no* input and is *randomized*) into an algorithm $\mathcal{A}$ (which takes *one* input bit-string and is *deterministic*).

Let T be the maximal number of steps Gen runs (recall that Gen is an algorithm, thus it has to terminate in finite time!). It follows that Gen uses at most T internal coin tosses during a computation! If $\mathcal{A}$ is invoked with input $r \overset{u}{\in} \{0,1\}^T$ it produces any key $k \in \mathcal{K}$ with exactly the same probability as Gen does.

So $\mathsf{Gen}' : r \overset{u}{\in} \{0,1\}^T$, $\mathsf{Enc}'_r(m) := \mathsf{Enc}_{\mathcal{A}(r)}(m)$ and $\mathsf{Dec}_r(m) := \mathsf{Dec}_{\mathcal{A}(r)}(m)$ defines our new ES with $\mathcal{K} = \{0,1\}^T$.

Notice that $\mathcal{A}$ is deterministic, so it returns always the same output given the same input!

Exercise 1.4 **Eavesdropping game**

The **q-eavesdropping game** is the version of the eavesdropping game where $\mathcal{A}$ may pass to the oracle two sequences. $m_0 = (m_0^{(1)}, \ldots, m_0^{(q)})$ and $m_1 = (m_1^{(1)}, \ldots, m_1^{(q)})$, and the oracle returns $c = (c^{(1)}, \ldots, c^{(q)})$ where $c^{(i)} :\overset{r}{=} \mathsf{Enc}_k(m_b^{(i)})$ and $m_b^{(i)} \in \mathcal{M}$.

We call an ES *perfectly secure under q encryptions (using the same key)* if there is no $\mathcal{A}$ which can win the q-eavesdropping game with probability strictly greater than $\frac{1}{2}$.

(a) Show that the one-time pad is not perfectly secure under $q = 2$ encryptions.

(b) Propose a q-time pad which is perfectly secure under q encryptions. Does your scheme remember some "state" between two en/decryptions?

Solution:

(a) To break the OTP with $q = 2$ encryptions we take e.g. $m_0 = (0^n, 0^n)$ and $m_1 = (0^n, 1^n)$. If we receive a ciphertext of the form $c = (x, x)$ we know for sure that m_0 was encrypted (and the key is x). If the ciphertext looks like $c = (x, y), x \neq y$ then m_1 was encrypted (and the key is again x).

(b) A perfectly secret q-time pad will generate a q-tuple of keys $k = (k_1, \ldots, k_q)$ (uniformly at random) and use k_i to encrypt the component m_i of the message vector. Perfect secrecy can be shown as well: If we had some hypothetical attack on this q-time pad ($|m_i| = n$) we could turn it into an attack on the OTP with block-length nq.

Remark: If you want to know more about "classical" ciphers and how to break them:

- `http://www.mysterytwisterc3.org` A very nice collection of flash- animated crypto-puzzles (Ceasar, Substitution, Permutation, Knapsack,...) but also programming challenges and real ciphers.
- *H.F. Gaines, Cryptanalysis, a study of ciphers and their solution. Dover, 1956.*

- *Simon Singh, The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography. Anchor, 2000.*