Information Systems Security
Series 2 : Entropy

October 6th, 2021

Reminder - Entropy

Let X be a random variable, with n possible values, each value x_i with probability p_i. The entropy of X, noted $H(X)$, is defined as :

$$H(X) = -\sum_{i=1}^{n} p_i log_2(p_i) = \sum_{i=1}^{n} p_i log_2(\frac{1}{p_i})$$

Intuitively, the entropy represents the minimal expected number of bits needed to encode the information contained inside this variable. The maximum entropy is obtained when all probabilities p_i are equal.

The joint entropy for two random variables X and Y is noted $H(X,Y)$, and is computed the same way as the standard entropy, but for each possible pair :

$$H(X,Y) = -\sum_{x \in X} \sum_{y \in Y} p(x,y) log_2(p(x,y))$$

The conditionnal entropy of X knowing Y is defined as :

$$H(X|Y) = -\sum_{y \in Y} \sum_{x \in X} p(y)p(x|y) log_2(p(x|y))$$

The conditionnal entropy has a special interest in the cryptographic field, as when we're encrypting something, the conditionnal entropy $H(Plaintext|Ciphertext)$ (entropy of the plaintext knowing the ciphertext) should aim to be equal, or almost equal, to the entropy of the plaintext itself $H(Plaintext)$.
It would assert that knowing the ciphertext doesn't give any kind of information about the plaintext.

Exercise 1 : Encryption and Entropy

We have the set of possible plaintexts $P = \{m_1, m_2, m_3\}$, teh set of possible ciphertexts $C = \{1, 2, 3, 4, 5\}$, and the set of keys $K = \{k_1, k_2, k_3\}$, with the following encryption rules :

	m_1	m_2	m_3
k_1	1	4	2
k_2	4	3	5
k_3	1	2	3

Each key has the same probability, and the plaintexts have the following probabilities of apparition :

$$p(m_1) = \frac{1}{4} \quad p(m_2) = \frac{3}{20} \quad p(m_3) = \frac{6}{10}$$

Compute the following entropies, with the exact values :

1. $H(P)$.
2. $H(K)$.
3. $H(C)$.
4. $H(P|C)$.
   Reminder : $p(A|B) = \frac{p(A\cap B)}{p(B)}$

Exercise 2 : Random Generators

Let G be a bad ”random” generator, generating 0s and 1s with probabilities $P(0) = 0.5 + \delta$ and $P(1) = 0.5 - \delta$.

Let’s use G to create a new generator A. To generate a bit with A, we do as follows : we generate two bits with G. If the result is 00 or 11, we just ditch the pair, and we generate a new one. If it’s 01, A generates a 0. If it’s 10, A generates a 1.

1. Compute the probability for each pair of bits to be generated with G.
2. Compute the probability for 0 and 1 with the new generator A.
3. What is the expected number of bits G needs to generate in order for A to generate x bits ?
4. What is the advantage of A compared to G ? At which cost did it come ?
5. Instead of taking independant pairs of bits from G, we’re now taking intertwined pairs (i.e., bits 1 and 2 from G, then bits 2 and 3 from G, then bits 3 and 4, ...). Would A still be a viable generator ?

Exercise 3 : Password Entropy

We consider characters encoded in ASCII 8-bit (for simplication purposes, we'll consider that every character in ASCII is printable and can be used in a password).

1. Compute the entropy of a password of 10 characters that is a valid date (i.e. a two digit number for the month, a two digit number for the day, a four digit number for the year, and two "/" : "MM/DD/YYYY"). Consider every year between 0000 and 2021 to be valid (both included), and ignore leap years (i.e. consider February 29th never exists).
2. Compute the entropy of a password with 11 randomly chosen characters (in ASCII 8-bit).
3. Compute the entropy of a password with 11 characters (from ASCII 8-bit), including at least one number, at least one lowercase letter, at least one uppercase letter, and at least one special character (a character that is neither of the first three categories).
4. Compute the entropy of a password that is the concatenation of five randomly chosen words from an english dictionnary containing 200000 words.
5. Compute the entropy of a password that is the concatenation of the name of six different species of Pokémon, each chosen randomly (written in ASCII 8-bit). A Pokémon name length is from 3 to 12 characters, and as of right now, there are 901 known Pokémon species.

Help : To simplify calculations, for password 3, consider independent characters. For passwords 4 and 5, to simplify calculations, consider there is no collision, i.e. each possible choice of words creates a different password (i.e. we consider there is no case where different combinations of words give the same password, like "abc" + "de" = "a" + "bcde". Each different groups of words gives a different key).