7. Problem Session
Cryptographic Hash Functions
(Summer Term 2014)

Bauhaus-Universität Weimar, Chair of Media Security
Prof. Dr. Stefan Lucks, Jakob Wenzel
URL: `http://www.uni-weimar.de/de/medien/professuren/mediensicherheit/teaching/`

Date: 08.07.2014 (15:15)

Inform yourself about Shannon entropy and Min entropy.

Task 1 (4 Credits) Entropy
Consider a source $\mathcal{Q}$ which is used to generate passwords:

$$\mathcal{Q} = \{a, \ldots, z, A, \ldots, Z, 0, \ldots, 9\}$$

Compute the Shannon entropy and the Min entropy of this source under the following conditions:

a) The source $\mathcal{Q}$ follows a uniform distribution. Thus, all elements have the same probability.

b) The probability that a lower case letter occurs is twice as high as the probability that a non-lower case letter occurs.

c) The probability for an element from $\{0, \ldots, 9\}$ is given by 0.7.

If you use a program to solve this task, append it to your solution.

Task 2 (4 Credits) Variant of HMAC
Consider a variant of the keyed hash function HMAC (see Slide 240 for the original one), called HMAC*, where the outer call to H is replaced by an invertible function H'. Thus, HMAC* is defined as:

$$\text{HMAC*}(K, U) = H'(\text{const}_2 \oplus K \mid\mid H(\text{const}_1 \oplus K \mid\mid U)).$$

Show that it is possible to efficiently provide a forgery for HMAC*. Is a similar forgery attack possible when only the inner invocation of H is replaced with an invertible function H'? Explain your answer.

Task 3 (4 Credits) Weakness of ROMix
Consider the ROMix function as given on Slide 247. Assume that an adversary is able to run a spy process on the machine of its target (the person whose password it wants to crack). Using this spy process, the adversary cannot directly get the original password, but it is able to see which values V_i (see both `for`-loops in the ROMix algorithm) were accessed from memory. Explain how the adversary can use this information to significantly decrease the amount of time of a brute-force attack.

Task 4 (8 Credits) Programming Task – Entropy
One method to generate a password is to think of a sentence and take the initials of each word, e.g., the sentence "This is a secure password generated from a sentence" leads to "Tiaspgfas". Now, consider a given source, e.g., a long text. Then, a more general way of computing the strength of passwords generated using this source, is to consider all initials of all words within this text, and to compute the entropy of the resulting data.

Under the following links you can find two texts, one in English and one in German.

English: Complete Poetical Works of Edgar Allan Poe

German: Die Göttliche Komödie (Dante Alighier)

Consider the English text and extract the initial letters of all words as well as their frequencies. Consider the gathered data as an entropy source $\mathcal{Q}$. Note that since we are interested in the natural language, you should ignore special characters (*hint*: you can use the `isalpha()` function to test, if a character is alphabetic).

a) Compute the Shannon entropy and the Min entropy in bits of $\mathcal{Q}$. Explain why these two values differ so much.

b) Have a deeper look at the data from the source Q, i.e., the characters and their frequencies. Find a way to slightly adapt the way of building the source (extracting the data, without changing the original files itself) so that the Shannon entropy and the Min entropy converge. Explain what you did and compute both values again.

c) Analyze the German text the same way as in Tasks a) and b), and compare the results with the results gained from the English text. What do you notice? Explain your observation.