Problem Set 3
Cryptographic Hash Functions
(Summer Term 2019)

Bauhaus-Universität Weimar, Chair of Media Security
Prof. Dr. Stefan Lucks, Jannis Bossert, Nathalie Dittrich, Eik List
URL: `http://www.uni-weimar.de/de/medien/professuren/mediensicherheit/teaching/`
Due Date: 16 May 2019, 11:00 AM, via email to `jannis.bossert(at)uni-weimar.de`.

Task 1 – Nostradamus Attack (3 Credits)
Prior to this task, read [1] or [2]. To convince Bob from her gift of clairvoyance, Eve offered Bob the following game before the previous soccer world cup. Before the event, she published a hash H. After the event, Eve released a document x with the correct hash $h = H(x)$ containing the correct winner. Explain with your own words, how Eve could have achieved this, and find an optimal computational complexity of your approach for an iterated cryptographic hash function $H : \{0,1\}^* \to \{0,1\}^{64}$.

Task 2 – Block-Cipher-Based Compression Functions (6 Credits)
Analyze the security for the following block cipher based compression functions in terms of the attack classes introduced in the course (Slide 74 in Chapter 4). This means, if there exists an attack, describe it briefly, and otherwise, explain briefly why the construction is secure. Let $E : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be an ideal block cipher with n-bit key and input, and let $\text{const}_1, \text{const}_2 \in \{0,1\}^n$ be known constants.

a) $H_i \stackrel{\text{def}}{=} E_{\text{const}_1}(M_i) \oplus \text{const}_2$.

b) $H_i \stackrel{\text{def}}{=} E_{M_i}(H_{i-1}) \oplus M_i \oplus H_{i-1}$.

c) $H_i \stackrel{\text{def}}{=} E_{M_i \oplus H_{i-1}}(H_{i-1}) \oplus \text{const}_1$.

d) $H_i \stackrel{\text{def}}{=} E_{M_i \oplus H_{i-1}}(M_i \oplus H_{i-1}) \oplus \text{const}_1$.

e) $H_i \stackrel{\text{def}}{=} E_{H_{i-1}}(M_i \oplus H_{i-1}) \oplus H_{i-1}$

f) $H_i \stackrel{\text{def}}{=} E_{M_i}(M_i) \oplus H_{i-1}$.

Task 3 – Indifferentiability (3 Credits)
Let a two-level construction be defined as shown on Slide 43:

$$H[F](x) \stackrel{\text{def}}{=} F(h(x)).$$

This construction is differentiable (that means insecure) if h is a cryptographically secure one-way hash function (COWHF), and F is random function modelled as a random oracle:

1. Ask for $H(x) = y$.
2. Compute $h(x) = z$.
3. Ask for $F(z) = y'$.
4. If $y = y'$ output "real", else "random".

Next, consider the same construction where h is not a COWHF but an invertible public permutation, e.g., the identity. Show that this new construction is secure in the indifferentiability model.

Task 4 – Collision Search (6 Credits)
For any Python implementation tasks, stick to the PEP8 coding guides (in doubt, use `autopep8`), and the `argparse` package for a clean CLI. Please clean your code with `pylint` before submitting.
In this task, you shall read into and implement a set of memory-efficient algorithms for collision and near-collison search:

a) Floyd's cycle-finding algorithm,

b) Brent's cycle-finding algorithm, and

c) Distinguished points.

Implement each algorithm in Python to find partial collisions on the c most significant bits of SHA-1. You do not have to implement SHA-1 yourselves but can use `pycrypto` library for this purpose. Measure the time/calls required for each algorithm for finding a near-collision on $c \in \{16, 24, 32\}$ bits. Use at least two threads. Your programs should be callable by

```
$ ./floyd.py -c 32 -t 4
$ ./brent.py -c 32 -t 4
$ ./distinguished_points.py -c 32 -t 4
```

Bonus: The group who finds the largest partial collision wins a bag of gummy bears. Copy-pasting/adapting SHA-1 collisions from papers etc. is *not* a valid option.

References

[1] John Kelsey, Tadayoshi Kohno: Herding Hash Functions and the Nostradamus Attack. EUROCRYPT 2006: 183-200, `http://link.springer.com/chapter/10.1007/11761679_12`.

[2] John Kelsey, Tadayoshi Kohno: Herding Hash Functions and the Nostradamus Attack. IACR Cryptology ePrint Archive 2005: 281 (2005) (full version), `http://eprint.iacr.org/2005/281`.