1. Bit commitment scheme is to

   (a) Alice creates a commitment c for a a value d,

   (b) Alice presents c to Bob,

   (c) some other steps of Alice and Bob ...

   (d) Alice "opens" c by showing d and proving that she has created c from d

Procedury:

commit(d) → (c, s)

c - wartość wysyłana do Boba

s - wartości potrzebne do openingu

open(d, c, s) → {0, 1} (sprawdzenie poprawności)

Własności bezpiecznego schematu commitmentów:

hiding - c nie ujawnia żadnych informacji na temat d (rozkład c jest nierozróżnialny dla każdego d)

binding - Alice nie jest w stanie otworzyć commitmentu c dla innej wartości niż d

Explore possibilities to construct a commitment scheme based on:

(a) a hash function,

(b) asymmetric encryption,

(c) symmetric encryption.

In each case formulate necessary properties of the underlying scheme.

a) commit(d):
wylosuj s
c = Hash(s,d)
zwróć (c,s)

open(d,c,s)
zwróć Hash(s,d) == c

hiding - wynika z one-wayness (first preimage resistance) funkcji Hash
binding - wynika z collision resistance (conflict freeness) funkcji Hash.
Dla s i d ograniczonych w taki sposób, że nie istnieją kolizje, otrzymujemy perfect binding.
W zależności od doboru rozmiaru parametrów i funkcji Hash może wystąpić perfect hiding (tzn. dla każdego d i c należącego do obrazu funkcji Hash istnieje s, takie że c = Hash(s,d)).

(b) asymmetric encryption,

commit (d):

$(x, y) \leftarrow KeyGen()$

$c = (y, Enc_y(d))$

zwróć $(c, s = x)$

open(d, c, s):

$(y, z) \leftarrow c$

$x \leftarrow s$

1° sprawdź, czy $Dec_x(z) = d$

2° sprawdź, czy $Dec_x(Enc_y(m)) = m$ dla dowolnej wiadomości m.

Jeśli 1° i 2°, zwróć 1. Wpp zwróć 0.

hiding - wynika z CPA-security schematu szyfrowania

binding - z właściwości schematów szyfrowania mamy perfect binding;
dla danego klucza publicznego istnieje jeden klucz prywatny
i funkcja Dec jest deterministyczna.

(c) symmetric encryption.

commit(d):

$k \leftarrow KeyGen()$
$r \leftarrow$ losowe
$z = Enc_k(d \| r)$
zwróć $(c = (z, r), s = k)$

open(d, c, s):

$(z, r) \leftarrow c$
$k \leftarrow s$
zwróć $Dec_k(z) == d \| r$

hiding - wynika z CPA-security schematu szyfrowania

binding - polega na trudności znalezienia klucza $k' \neq k$ i $d' \neq d$, takiego że $Dec_{k'}(z) == d' \| r$ (zatem r musi być odpowiednio duże).

2. AES can be used to create a hash function. (One of the advantages is that in case of a weak embedded device one can implement the code of AES instead of, say, AES and SHA-3.This reduces the code size.) The algorithm is as follows:

   - apply padding: add zeroes so that we have an odd number of blocks of length 128, add the length of the original file in the next block of length 128,
   - put $H_0 = 2^{256} - 1$ (string of 256 ones),
   - $H_i = \mathrm{Enc}_{x_i}(H_{i-1}) \oplus H_{i-1}$, where x_i is the ith block after padding,
   - output the last computed H_i

   Discuss (informally), why this construction should have the properties required from a cryptographic hash function.

Tutaj w treści jest pewne niedopowiedzenie; wymagane jest szyfrowanie 256-bitowych inputów. Dla AES-a, który jest szyfrem działającym na 128-bitowych blokach, należałoby jeszcze zdefiniować, w jaki sposób szyfrować 256 bitów naraz (np. poprzez ustalenie konkretnego mode of encryption). Tak naprawdę w zadaniu nie chodziło o to, żeby użyć AESa, ale Rijndaela z 256-bitowym stanem i kluczami (Rijndael był finalistą konkursu NISTu, którego 128-bitową wersję ustandaryzowano jako AES).

W dalszym rozwiązaniu zakładamy, że Enc to 256-bitowy Rijndael.

One - wayness (first preimage)

Sprowadźmy to do najprostszego przypadku, kiedy mamy do zahashowania 128 bitów. Oznaczmy input jako x.

po paddingu: $x_1 = 0x$ [x] [00 ..... 0080] ← zakodowana długość 128

$H_0 = 0b\,111\ldots1$

$H_1 = Enc_{x_1}(H_0) \oplus H_0$

W wyniku otrzymujemy H_1. H_0 jest znane, więc możemy je odksorować i otrzymać $c = Enc_{x_1}(H_0) = H_0 \oplus H_1$.

Znalezienie przeciwobrazu wymaga odgadnięcia x_1, które tutaj użyte jest jedynie jako klucz. Nie da się tego zrobić, bo Enc jest dobrą funkcją szyfrującą.

conflict freeness (collision resistance)

a) znajdujemy x_i, x_i', takie że $x_i \neq x_i'$ i $Enc_{x_i}(H_{i-1}) = Enc_{x_i'}(H_{i-1})$.

To się ogólnie może zdarzyć, ale prawdopodobieństwo jest zaniedbywalnie małe; dla danego H_{i-1} ppb to $\sqrt{\frac{1}{2^{256}}} = \frac{1}{2^{128}}$ (dla idealnego szyfru blokowego, nierozróżnialnego od losowej permutacji – możemy przyjąć, że Enc jest nierozróżnialny od idealnego szyfru). Możemy spróbować znaleźć taką kolizję tylko dla względnie niedużej liczby wartości H_{i-1}. Zatem nie powinno się dać tego zrobić w sensownym czasie.

b) załóżmy, że mamy H_i i $H_j' \neq H_i$

Znajdźmy x_{i+1}, x_{j+1}', takie że $Enc_{x_{i+1}}(H_i) \oplus H_i = Enc_{x_{j+1}'}(H_j') \oplus H_j'$. (próbujemy dojść do tego samego stanu różnymi ścieżkami).

$$\begin{cases} c = Enc_{x_{i+1}}(H_i) & (= H_{i+1} \oplus H_i) \\ c' = Enc_{x_{j+1}'}(H_j') & (= H_{j+1}' \oplus H_j'), \; H_{i+1} = H_{j+1}' \end{cases}$$

żeby znaleźć x_{i+1} lub x_{j+1}', musielibyśmy w którymś przypadku wyciągnąć klucz do szyfrowania z pary plaintext-ciphertext, co nie jest możliwe przy dobrym schemacie szyfrowania.

second preimage resistance

→ wynika nam z conflict freeness.

Ten schemat jest prawie OK, ale ma jedną wadę: da się na nim wykonać length extension attack. Znając długość x (ale nie jego wartość), po paddingu mamy

$x \| 0\ldots00 \| len(x)$ - załóżmy, że długość tego całego ciągu to k.

Możemy ten ciąg potraktować jako prefix innego ciągu:

$x \| 0\ldots00 \| len(x) \| y \| 0\ldots00 \| k+len(y)$

Jeśli Hash(x) jest użyty do zapewnienia integralności payloadu, stosunkowo łatwo jest coś do payloadu dokleić (nie znając samego payloadu) i przeliczyć hash. Jest to istotne np. gdy Hash wykorzystywany jest jako MAC (Hash(secret || payload)).

Uwaga: Wiekszość używane w praktyce schematy są na to podatne, np. SHA256 i SHA512 (a także używane kiedyś MD5 i SHA-1).

Przedstawiony w zadaniu schemat to AES-Hash bez ostatniego kroku, który miał na celu chronić m.in. przed length extension attack.
Schemat został zaproponowany w 2001 roku, ale nie został ustandaryzowany..

Ostatni krok:

wersja I: $$H = H_{n+1} = Enc_{H_n}(H_n) \oplus H_n$$

wersja II: $$H = H_{n+1} = Enc_{H_n \oplus X_n}(H_n) \oplus H_n$$

Ostatni krok uniemożliwia wykonanie dodatkowych rund na wyniku.

3. In the previous problem, replace the previous formula by $H_i = \mathrm{Enc}_{x_i}(H_{i-1})$. What are the problems for the resulting hashing scheme?

Znając suffix x, jesteśmy w stanie cofnąć część rund (dopóki znamy x_i), a następnie zastąpić suffix czymś innym. W schemacie z zadania 2 nie da się tego zrobić, ponieważ bez znajomości prefixu, w i-tej rundzie nie znamy H_{i-1}, więc nie wiemy, co odtworzyć.

Przykład: Użycie hasha jako MACa: Hash(secret || payload).

Jeśli znamy payload, a nie znamy secretu, jesteśmy w stanie cofnąć hasha do momentu, gdzie kończyła się z payloadu, a następnie policzyć go do końca zastępując payload przez payload'

– w wyniku uzyskamy Hash(secret || payload').

4. (a) Create a hash value of your name using AES Hash, MD4, MD5, SHA-1, SHA-256, SHA-512.

   (b) Install BLAKE2 (`https://github.com/BLAKE2/`) on your computer. You might be asked to hash something with BLAKE2.

(Note that BLAKE was one of the finalists of the NIST competition).

To było zadanie domowe

(bez AES Hash)

5. Urzedowe Poświadczenie Odbioru (UPO) for your electronic tax declaration (assuming you submit your declaration online) contains “skrót dokumentu” and “skrót podpisanego dokumentu”.

   - if you have the xml file for the UPO, browse through this file and find the meaning of these fields,
   - if you have not declared your income online, then compare the proof value of the UPO for tax declaration and a seal of the tax authority on the paper copy of a tax declaration.

URZĘDOWE POŚWIADCZENIE ODBIORU DOKUMENTU ELEKTRONICZNEGO

A. NAZWA PEŁNA PODMIOTU, KTÓREMU DORĘCZONO DOKUMENT ELEKTRONICZNY

Ministerstwo Finansów

B. INFORMACJA O DOKUMENCIE

Dokument został zarejestrowany w systemie teleinformatycznym Ministerstwa Finansów

Identyfikator dokumentu	Dnia (data, czas):
6513a0[redacted]	26.04.2022 10:53:46

Skrót złożonego dokumentu - identyczny z wartością użytą do podpisu dokumentu:

Vw6Uwz[redacted]4I= [2F7348[redacted]2C]

Skrót dokumentu w postaci otrzymanej przez system (łącznie z podpisem elektronicznym):

913CFA[redacted]86

Dokument zweryfikowano pod względem zgodności ze strukturą logiczną:

http://crd.gov.pl/wzor/2022/01/03/11181/schemat.xsd dla PIT-37 wariant 28 schemat 1-0E

Identyfikator podatkowy podmiotu występującego jako pierwszy na dokumencie:	Identyfikator podatkowy podmiotu występującego jako drugi na dokumencie:
numer PESEL: [redacted]	

Urząd skarbowy, do którego został złożony dokument:

URZĄD SKARBOWY WROCŁAW-KRZYKI

Stempel czasu:

MjAyMi0wNC0yNlQxMDo1Mzo0Ni4wMDArMDI6MDA=

Dokument wystawiony automatycznie przez system teleinformatyczny Ministerstwa Finansów

Data i czas wystawienia dokumentu: 26.04.2022 10:53:50

Skrót złożonego dokumentu
→ dwa hashe, pierwszy 256-bitowy (32 bajty zakodowane w base64), prawdopodobnie SHA-256, drugi 128-bitowy (16 bajtów w hexie) - MD5.

Skrót dokumentu w postaci (itd.)
→ 128-bitowy hash - MD5