PWr, Inf. algo. **CRYPTOGRAPHY, 2022 assignments list # 10, authentication (corrected)**

1. One of the methods of authentication is to use one-time passwords printed under scratch fields.

   An option is to use a chain of hashes:

   - choose r at random,
   - put $\pi_{1000} = r$
   - for $i < 1000$, $\pi_i := \text{Hash}(\pi_{i+1})$

   Then, the password used in the ith round is π_i.

   How to store efficiently the passwords on the side of a system? How to extend the method so that, say, 1.000.000 one-time passwords can generated?

2. For authentication based on login and password, the password entered by a user should be compared with the password stored for this user by the system.

   Compare the following options for storing users' passwords. For storing a password π of user U, there is the following record with the index U:

   via encryption: $\text{Enc}_K(\pi)$, where K is a secret key of the system,

   via encryption with salt: $[\text{Enc}_K(\pi, r), r]$, where K is a secret key of the system, r is chosen at random for this record,

   via hashing: $\text{Hash}(\pi)$,

   via hashing with salt: $[\text{Hash}(\pi, r), r]$, where r is chosen at random for this record,

   via HMAC with salt: $[\text{HMAC}_K(\pi, r), r]$, where is a secret key of the system, r is chosen at random for this record,

   Which method MUST NOT be used in practice and why?

3. Read the specification available via the following url:

   `https://one.google.com/about/vpn/howitworks`

   List all threats/attacks that apply for the traditional approach and which disappear once the method described in the specification is used.

4. Assume that for Schnorr identification protocol the pseudo-random generator turns out to be weak. What may happen?

5. HB and HB+ authentication protocols have been designed in order to use only linear algebra.

   HB+ works as follows. The shared secret are the binary vectors x and y. Authentication of A requires executing the following round multiple times:

   (a) A chooses a vector b at random and sends b to Verifier,

   (b) Verifier chooses a vector a at random and sends a to A,

   (c) A computes $z = <a, x> \oplus <b, y> \oplus e$, where bit e equals 1 with probability p, $\oplus$ denotes the XOR operation, and $<a, x>$ is the scalar product of vectors a and x

   (d) A sends z to Verifier,

   (e) Verifier computes $z' = <a, x> \oplus <b, y>$, the round succeeds for him if $z = z'$

Verifier accepts A if an appropriate majority of rounds succeeds (what is “appropriate” follows from probability theory).

(a) this scheme is related to the LPN problem. In which way?

(b) HB+ is nevertheless insecure. The attacker changes a sent by the verifier to $a' := a \oplus \delta$ – in each round with the same vector δ – and observes what happens.
How does this help the attacker to collect information on the secret key?

/-/ Mirosław Kutyłowski