Information Security class
Laboratory session 3

instructors: Nicola Laurenti, Francesco Ardizzon

Fall semester 2020-21

Naïve entity authentication scheme

Your aim is to implement and evaluate the weakness of the following naïve challenge-response scheme for entity authentication

entities the prover A, the verifier B

setup A and B have shared a secret key k of ℓ_k bits, randomly and uniformly generated

1

A → B : $u_1 = \mathrm{id}_\mathrm{A}$

2

B : generates a random and uniform challenge c of ℓ_c bits

B : updates an integer counter n

B → A : $u_2 = (c, n)$

3

A : converts c to its decimal (base 10) representation and computes the sum of its decimal digits, call the sum s_c;

A : reads k as an integer (base 2) and computes $t = k + n$; ("+" is the usual sum between integers)

A : converts t to its decimal (base 10) representation and computes the sum of its decimal digits, call the sum s_t;

A : computes the product $s = s_c s_t$;

A : convert s to its binary representation, let the resul be the response r;

A → B : $u_3 = r$

4

B : performs the same computations and obtains the expected response $\hat{r}$

B : if the result are identical $r = \hat{r}$ A is accepted, otherwise A is rejected

Your tasks

1. Implement the protocol in a programming language of your choice so that its complexity is polynomial in ℓ_c and ℓ_k.
2. Design and implement an attack to the above protocol such that, without knowing the key k, and having observed a previous legitimate round of the protocol where the counter had the value $n' = n - 25$, a malicious entity C pretends to be A and attempts to be accepted by B. Evaluate through simulations the computational complexity and success probability for this attack with several values of ℓ_c and ℓ_k.
3. Design and implement an attack such that, without knowing the key k nor observing any previous run of the protocol, a malicious entity C pretends to be A and attempts to be accepted by B. Evaluate through simulations its computational complexity and success probability by simulation with several values of ℓ_c and ℓ_k.

What you need to turn in

Each team must turn in, through the Moodle assignment submission procedure:

1. the source code for your implementation (either as a single file, many separate files, or a compressed folder)

2. a short report (to be submitted as a separate file from the source code file / compressed folder) in a graphics format (PDF, DJVU or PostScript are ok; Word, TEX or LATEX source are not), including:

   (a) a brief description of your designs and implementations for Tasks 1-3, explaining your choices;

   (b) the evaluated efficiency and security metrics for your system:

      i. a plot of the computational complexity of a legitimate protocol run vs ℓ_k, for several different values of ℓ_c

      ii. a plot of the computational complexity for the attack devised in point 2 above, vs ℓ_k, for several different values of ℓ_c

      iii. a plot of the success probability for the attack devised in point 2 above, vs ℓ_k, for several different values of ℓ_c

      iv. a plot of the computational complexity for the attack devised in point 3 above, vs ℓ_k, for several different values of ℓ_c

      v. a plot of the success probability for the attack devised in point 3 above, vs ℓ_k, for several different values of ℓ_c