Network Security class

Nicola Laurenti, 2012-13

Laboratory session 3

student: ______________________________ ID: ____________

A and B want to establish a secure connection based on symmetric cryptography, sharing a secret key k. In order to do that, since each of them has a connection with asymmetric cryptography to C, they want to use the following protocol:

k_A	private key of A
k'_A	public key of A (known to C)
k_B	private key of B
k'_B	public key of B (known to C)
k_C	private key of C
k'_C	public key of C (known to A and B)

$\boxed{1}$ $\mathsf{A}:$ generates nonce $r_\mathsf{A} \sim \mathcal{U}(\mathcal{R})$
$\mathsf{A} \to \mathsf{C}:$ $[\mathrm{id}_\mathsf{A}, \mathrm{id}_\mathsf{B}, r_\mathsf{A}]$

$\boxed{2}$ $\mathsf{C}:$ generates $k \sim \mathcal{U}(\mathcal{K})$, $u_1 = [k, r_\mathsf{A}]$, encrypts $x_1 = E_{k'_\mathsf{A}}(u_1)$
$\mathsf{C} \to \mathsf{A}:$ x_1
$\mathsf{A}:$ decrypts $u_1 = D_{k_\mathsf{A}}(x_1)$

$\boxed{3}$ $\mathsf{A}:$ signs $t_2 = S_k([\mathrm{id}_\mathsf{A}, \mathrm{id}_\mathsf{B}, r_\mathsf{A}])$
$\mathsf{A} \to \mathsf{B}:$ $x_2 = [\mathrm{id}_\mathsf{A}, \mathrm{id}_\mathsf{B}, r_\mathsf{A}, t_2]$

$\boxed{4}$ $\mathsf{B} \to \mathsf{C}:$ $[\mathrm{id}_\mathsf{A}, \mathrm{id}_\mathsf{B}, r_\mathsf{A}]$

$\boxed{5}$ $\mathsf{C}:$ encrypts $x_3 = E_{k'_\mathsf{B}}(u_1)$
$\mathsf{C} \to \mathsf{B}:$ x_3
$\mathsf{B}:$ decrypts $u_1 = D_{k_\mathsf{B}}(x_3)$

$\boxed{6}$ $\mathsf{B}:$ verifies $b = V_k([\mathrm{id}_\mathsf{A}, \mathrm{id}_\mathsf{B}, r_\mathsf{A}], t_2)$

Your assignment is to:

1) implement the above protocol in your favourite language, and check its correctness;
2) identifiy its vulnerabilities and devise an attack that exploits them, under reasonable assumptions;
3) implement the attack and evaluate its success probability in dependence of the protocol parameters;
4) suggest improvements to the protocol and implement them.

Provide a description of your solution, with justification of your choices, the code for your implementations, and adequate discussion of the results.

General notes: Implementing your protocol may require you to use several cryptographic primitives, such as symmetric or asymmetric encryption/decryption, signing/verification, one-way functions, cryptographic hash functions, key generation, etc. Feel free to use any reasonable (and correct) implementation of such primitives you find for the language you've chosen, or even simplified models such as a (pseudo-)random oracle. The vulnerability of the protocols should not depend on a poor implementation of such primitives. Also, unlike for cryptographic keys, you should model the random choice of a password as (strongly) non uniform.

Network Security class

Nicola Laurenti, 2012-13

Laboratory session 3

student: ______________________________ ID: ____________

A wants to authenticate himself to B, with whom he shares a secret key $k \in \{0,1\}^\ell$. B employs the following N-round protocol, which does not require A to send k over the channel.

For $n = 1$ to N

$\boxed{n.1}$ B : generates a challenge $x_n \sim \mathcal{U}(\{0,1\}^\ell)$
$\mathsf{B} \to \mathsf{A}$: x_n

$\boxed{n.2}$ A : calculates $y_n = x_n \oplus k$ and $b_n = \begin{cases} 1 & \text{if } y_n \text{ has an even number of ones} \\ 0 & \text{if } y_n \text{ has an odd number of ones} \end{cases}$
$\mathsf{A} \to \mathsf{B}$: b_n

$\boxed{n.3}$ B : checks b_n. If b_n is incorrect, the authentication is rejected

If b_n is correct for all $n = 1, \ldots, N$, the authentication is accepted

Your assignment is to:

1) implement the above protocol in your favourite language, and check its correctness;
2) identifiy its vulnerabilities and devise an attack that exploits them, under reasonable assumptions;
3) implement the attack and evaluate its success probability in dependence of the protocol parameters;
4) suggest improvements to the protocol and implement them.

Provide a description of your solution, with justification of your choices, the code for your implementations, and adequate discussion of the results.

General notes: Implementing your protocol may require you to use several cryptographic primitives, such as symmetric or asymmetric encryption/decryption, signing/verification, one-way functions, cryptographic hash functions, key generation, etc. Feel free to use any reasonable (and correct) implementation of such primitives you find for the language you've chosen, or even simplified models such as a (pseudo-)random oracle. The vulnerability of the protocols should not depend on a poor implementation of such primitives. Also, unlike for cryptographic keys, you should model the random choice of a password as (strongly) non uniform.

Network Security class

Nicola Laurenti, 2012-13

Laboratory session 3

student: ____________________ ID: __________

A chooses a password p_{A} with up to $L_{\max}$ decimal digits and a one-way function $h : \{0, 1, \ldots, 9\}^{L_{\max}} \to \{0, 1, \ldots, 9\}^{L_{\max}}$. Let $x_1 = h([p_A, 0, \ldots, 0]), x_N = \underbrace{h \circ h \circ \ldots h}_{N}(p_{\mathsf{A}})$. A securely delivers x_N to B. For each protocol run, $n = 1, \ldots, N-1$

$\boxed{1}$ $\mathsf{A} \to \mathsf{B}$: $u_1 = [\mathrm{id}_{\mathsf{A}}]$

$\boxed{2}$ B : $r_n \sim \mathcal{U}(\mathcal{R})$
$\mathsf{B} \to \mathsf{A}$: $u_2 = [n, r_n]$

$\boxed{3}$ A : $x_n = h(x_{n-1})$
A : $u_3 = [x_n, r_n]$
$\mathsf{A} \to \mathsf{B}$: u_3

$\boxed{4}$ B : checks whether $\underbrace{h \circ h \circ \ldots h}_{N-n}(x_n) = x_N$ and the received r_n is consistent with the transmitted one.

Your assignment is to:

1) implement the above protocol in your favourite language, and check its correctness;
2) identifiy its vulnerabilities and devise an attack that exploits them, under reasonable assumptions;
3) implement the attack and evaluate its success probability in dependence of the protocol parameters;
4) suggest improvements to the protocol and implement them.

Provide a description of your solution, with justification of your choices, the code for your implementations, and adequate discussion of the results.

General notes: Implementing your protocol may require you to use several cryptographic primitives, such as symmetric or asymmetric encryption/decryption, signing/verification, one-way functions, cryptographic hash functions, key generation, etc. Feel free to use any reasonable (and correct) implementation of such primitives you find for the language you've chosen, or even simplified models such as a (pseudo-)random oracle. The vulnerability of the protocols should not depend on a poor implementation of such primitives. Also, unlike for cryptographic keys, you should model the random choice of a password as (strongly) non uniform.

Network Security class

Nicola Laurenti, 2012-13

Laboratory session 3

student: ____________________________ ID: ____________

Given $\mathcal{M} = \{0,1\}^n$, $\mathcal{K} = \{0,1\}^{n+m-1}$, $\mathcal{T} = \{0,1\}^m$, let $(S(\cdot,\cdot), V(\cdot,\cdot,\cdot), \mathcal{M}, \mathcal{K}, \mathcal{T})$ be a symmetric authentication / integrity protection mechanism such that

$$t = S(k, r) = T(k)r, \quad t \in \mathcal{T}, r \in \mathcal{M}$$

being $T(k)$ the $m \times n$ Toeplitz matrix identified by k, and the verification function is defined accordingly.

Suppose A wants to authenticate himself to B, with whom he shares the secret key k, by using the following protocol

$\boxed{1}$ $\mathsf{A} \to \mathsf{B}:$ id_{A}

$\boxed{2}$ $\mathsf{B}:$ generates a nonce $r \sim \mathcal{U}(\mathcal{M})$
$\mathsf{B} \to \mathsf{A}:$ r

$\boxed{3}$ $\mathsf{A}:$ calculates $t = S(k, r)$
$\mathsf{A} \to \mathsf{B}:$ t

$\boxed{4}$ $\mathsf{B}:$ checks if $V(k, r, t) = \mathsf{ok}$

Your assignment is to:

1) implement the above protocol in your favourite language, and check its correctness;
2) identifiy its vulnerabilities and devise an attack that exploits them, under reasonable assumptions;
3) implement the attack and evaluate its success probability in dependence of the protocol parameters;
4) suggest improvements to the protocol and implement them.

Provide a description of your solution, with justification of your choices, the code for your implementations, and adequate discussion of the results.

General notes: Implementing your protocol may require you to use several cryptographic primitives, such as symmetric or asymmetric encryption/decryption, signing/verification, one-way functions, cryptographic hash functions, key generation, etc. Feel free to use any reasonable (and correct) implementation of such primitives you find for the language you've chosen, or even simplified models such as a (pseudo-)random oracle. The vulnerability of the protocols should not depend on a poor implementation of such primitives. Also, unlike for cryptographic keys, you should model the random choice of a password as (strongly) non uniform.