Cryptography and Architectures for Computer Security

Exam Code: 095947 (old 090959), A.Y. 2015–2016, Semester: 2

Prof. G. Pelosi

February 9th, 2017 – Exam Session

Name:.................................... Surname:..

Student ID:............................... Signature:...

Time: 2h:30'. Use of textbooks, notes, or Internet connected devices (including smartphones) is not allowed. The usage of simple calculators is allowed. Prior to turn in your paper, write your name on any additional sheet and sign it.

Question 1 [**2 pts**]

In order to accelerate the computation of the RSA encryption function, it is popular to use e=3 as a public exponent. Although RSA is considered to be a secure public-key cryptosystem, both the implementations and wrong parameter configurations of RSA can nullify the security of the system.

Consider an RSA cryptosystem with 2048-bit modulus, public exponent e=3 and assume each plaintext message to be encoded as a binary string with at most 128-bit length.
Explain why this is completely insecure.

Solution:
employing 128-bit plaintext messages, every ciphertext $c \equiv_n m^3$ will be computed as an integer number which is smaller than the modulus n. Indeed, c will be encoded with only $128 \cdot 3$=384 bits, thus no modular reduction will occur during the computation. A simple way to reverse the effect of the encryption transformation, without knowing the secret key, is computing the plaintext value m as $c^{1/3}$.

Question 2 [**3 pts**]

(a) What is the main advantage of the *One Time Pad* cipher and why is it hard to use in practice?

(b) How do we define perfect secrecy of a symmetric cipher over the sets of plaintext messages $\mathcal{M}$, ciphertext messages $\mathcal{C}$ and the set of keys $\mathcal{K}$?

(c) Assume a modified English alphabet with 24 letters, obtained through excluding from the original one the letters 'J' and 'Q'. Consider a cipher, which encrypts any given letter taking its position in the alphabet ('A'=0, 'B'=1, ..., 'Z'=23) and adding (mod 24) to it a random number obtained as the sum of the results of 4 dice rolls (each die roll is made with a fair six-faced die). Does this encryption scheme provide perfect secrecy? Motivate your answer.

Question 3 [6 pts]

(a) Consider the cyclic group $(\mathbb{Z}^*_{11^2}, \cdot)$. Compute the cardinality of the group, the number of its generators and exhibit at least one generator for five of its proper subgroups.

(b) Consider the finite field $\mathbb{F}_{11^2}$. Find the number of irreducible and primitive polynomials. Determine if $f(x){=}x^2+1{\in}\mathbb{F}_{11}[x]$ is a primitive polynomial.

Solution:

(a) $n{=}|(\mathbb{Z}^*_{11^2}, \cdot)|{=}\varphi(11^2){=}110{=}2\cdot 5\cdot 11$

Number of generators: $\varphi(n){=}40$

Proper divisors of n: 2, 5, 10, 11, 22, 55
$g{=}2$ is a generator as the following powers are all distinct from 1

$g^2{\equiv}4$, $g^5{\equiv}32$, $g^{10}{\equiv}_{121}32^2{\equiv}56$, $g^{11}{\equiv}_{121}112$, $g^{22}{\equiv}_{121}112^2{\equiv}_{121}81$, $g^{55}{\equiv}112^5{\equiv}_{121}120$

Five proper subgroups are the following ones:
$h_1{=}g^{\frac{110}{2}}{\equiv}g^{55}{\equiv}_{120}120$, $|\langle h_1\rangle|{=}2$
$h_2{=}g^{\frac{110}{5}}{\equiv}g^{22}{\equiv}_{120}81$, $|\langle h_2\rangle|{=}5$
$h_3{=}g^{\frac{110}{10}}{\equiv}g^{11}{\equiv}_{120}112$, $|\langle h_3\rangle|{=}10$
$h_4{=}g^{\frac{110}{11}}{\equiv}g^{10}{\equiv}_{120}56$, $|\langle h_4\rangle|{=}11$
$h_5{=}g^{\frac{110}{22}}{\equiv}g^{5}{\equiv}_{120}32$, $|\langle h_5\rangle|{=}22$

(b) The number of irreducible polynomial of degree 2 is $N_2(11){=}\frac{11^2-11}{2}{=}55$, while the number of primitive polynomials of degree 2 is $M_2(11){=}\frac{\varphi(120)}{2}{=}16$.

Assuming $\alpha{\in}\mathbb{F}_{11^2}\backslash\mathbb{F}_{11}$ to be primitive and $f(\alpha){=}0 \Leftrightarrow \alpha^2{\equiv}_{11}10$, as
$|\alpha|{=}|\mathbb{F}^*_{11^2}|{=}120 = 2^3\cdot 3\cdot 5$, then the proper divisors of 120 are: 2, 3, 4, 5, 6, 8, 10, 12, 15, 20, 24, 30, 40, 60. We need to check if
$\alpha^2{\equiv}_{11}10{\not\equiv}_{11}1$ is true,
$\alpha^3{\equiv}_{11}10\alpha{\not\equiv}_{11}1$ is true,
$\alpha^4{\equiv}_{11}100{\not\equiv}_{11}1$ is false,
therefore $f(x)$ is **not primitive**!

Question 4 [6 pts]

(a) Which are the criteria to select a suitable group, $(G, \cdot)$ $n{=}|G|$, for a discrete logarithm based cryptosystem?

(b) Let us denote as G the multiplicative group of the finite field $\mathbb{F}_{2^4}$ where $g(x){=}x^4+x+1{\in}\mathbb{F}_2[x]$ is a primitive polynomial, and $\alpha{\in}\mathbb{F}_{2^4}\backslash\mathbb{F}_2$, $g(\alpha){=}0$.
Establish if the following discrete logarithm exists and, in case of an affirmative answer, compute its value.

$$a = \log_{\alpha^2+1}(\alpha^{32})$$

Solution:

(a) see lectures ...

(b) $\alpha^2{+}1$ is a generator of G ($n{=}|G|{=}15{=}3{\cdot}5$) as:
$(\alpha^2{+}1)^3{\equiv}\alpha^6{+}\alpha^4{+}\alpha^2{+}1{\equiv}\ (\alpha^4+1)\cdot(\alpha^2+1){\equiv}\ \alpha(\alpha^2{+}1){\equiv}\alpha^3{+}\alpha{\not\equiv}1$
$(\alpha^2{+}1)^5{\equiv}(\alpha^2+1)^2{\cdot}(\alpha^3{+}\alpha){\equiv}...{\equiv}\alpha^2{+}\alpha{+}1{\not\equiv}1.$
Therefore, the proposed logarithm exists.
Observing that $\alpha^4{\equiv}\alpha{+}1$, and $\alpha^8{\equiv}(\alpha{+}1)^2{\equiv}\alpha^2{+}1$, we can compute the result as:

$a{=}32{\cdot}\log_{\alpha^2+1}(\alpha){\equiv}_{15}2{\cdot}(\log_{\alpha}(\alpha^2+1))^{-1}\equiv_{15} 2\cdot 8^{-1}\equiv_{15} 2\cdot 2\equiv_{15} 4$

Question 5 [12 pts]

(a) Apply the Pollard's ρ method to factorize the RSA modulus $n = p\cdot q = 703$.
Assume $f(x) = x^2 + 1 \bmod n$ as the "random-walking" function.
Show every step of the computation.
(As a backup alternative, apply a "trivial division" strategy).

(b) Choose an admissible public exponent e between the values $e{=}5_{\mathsf{dec}}$ and $e{=}9_{\mathsf{dec}}$ and compute the value of the corresponding RSA private key $k_{priv}{=}(p, q, \varphi(n), d)$. Show every step of the computation.

(c) Sign the message $m{=}5_{\mathsf{dec}}{\in}\mathbb{Z}_n$ (without employing any padding scheme) through applying the CRT. Describe each step of the procedure.

(d) Assume to work into the Montgomery domain: $(\widetilde{\mathbb{Z}}_p, +, \times)$, $p{=}13$

- Show the definition of the Montgomery Multiplication and the smallest admissible value for the Montgomery Radix: R
- Show the high-level pseudo-code to implement the Montgomery Reduction procedure `MRed(...)`, and prove the correctness of the algorithm.
- Compute a pair of integer values R', p' that satisfy the relation: $\gcd(R, p){=}R\,R'{-}p\,p'{=}1$.
- Compute the Montgomery multiplication $C = A\times B \bmod p$, where $A = 11_{\text{dec}}$ and $B = 4_{\text{dec}}$ are values in the Montgomery domain, assuming a binary encoding of the operands

Solution:

(a) see lectures ... $q = 19$, $p = 37$

(b) $\varphi(n){=}18{\cdot}36{=}2^3{\cdot}3^4$, therefore the admissible value for the public exponent is $e{=}5_{\mathsf{dec}}{\in}\mathbb{Z}^*_{\varphi(n)}$
$d{\equiv}_{648}5^{-1}{\equiv}...{\equiv}389$

(c) $s{=}405_{\text{dec}}$

(d) see lectures ... $R{=}16$, $R'{\equiv}_{13}9$, $p'{\equiv}_{16}11$. $C{=}\text{MonPro}(11, 4){\equiv}_{13}11{\cdot}4{\cdot}9{\equiv}_{13}6$.

Question 6 [3 pts]
Consider encrypted communications taking place between a general purpose remote server and a mobile client, in the following two scenarios:

(a) The server is momentarily running with a depleted entropy pool, while the client has a full one. Is it reasonable to forbid the use of the digital signature algorithm (DSA) in the TLS 1.2 cipher suite, out of security concerns? Motivate your answer.

(b) The client is momentarily running with a depleted entropy pool, while the server has a full one. Is it useful to forbid the use of ECDSA, when the client perform a **user authentication** via SSH login?

Solution:

(a) No. In a TLS handshake there is no signature computation, as the DSA algorithm will be potentially run only for the signature verification of certificate, which is not affected by a server-side depleted entropy pool. Also, in the TLS key exchange there is no use for a signature algorithm.

(b) After the SSH **server authentication** phase (either checking on the client the equality of the server certificate with the local copy, or on a trust-on-first-use base – in case the client does not have a local copy of the server certificate.... see slides) the client will encrypt a *session key* for the server (with the server public key). From this point on every transmission between client and server will be symmetrically encrypted with this session key. The client will also communicate to the server a list of the authentication mechanisms he supports.

Subsequently, the server needs to choose how to perform the SSH **user authentication** step (see lectures...). This exercise focuses on the public-key based one. Therefore, the server will send to the client a challenge that should be returned to him signed with the client private key.

The server may gain knowledge of the client ECDSA private key, in case of multiple login attempts: the server may challenge the client with two different challenges and the client will reply with two ECDSA signatures computed with the same random value (see lectures for ECDSA definition...).