Cryptography and Architectures for Computer Security

Exam Code: 095947 (old 090959), A.Y. 2016–2017, Semester: 2

Prof. G. Pelosi

July 5th, 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 [9 pts]

(a) CBC-MAC works as the classic *Cipher Block Chaining* (CBC) mode of operation for encryption, with the only difference that the Initialization Vector (`IV`) must be a fixed value (usually zero), while the outputs consists of a single block – which is last block of the CBC-ciphertext.

Given a generic block cipher and a plaintext message m, explain why the composition of the following authenticated ciphertext does not provide message integrity.

$$C \leftarrow \texttt{CBC-Enc}_k(m||\texttt{MAC}_k(m))$$

where $\texttt{CBC-Enc}_k(\cdot)$ denotes the CBC-encryption mode of a block cipher encryption primitive $\texttt{Enc}_k(\cdot)$, while $\texttt{MAC}_k(m)$ denotes a MAC-CBC computation performed with the same key k.

(b) Compare the security guarantees of the following authenticated encryption procedures.

$$\begin{aligned} C &\leftarrow \texttt{ModeOfOperation-Enc}_{k_1}(m||\texttt{MAC}_{k_2}(m)), \text{ with } k_1 \neq k_2 \\ C &\leftarrow \texttt{ModeOfOperation-Enc}_{k_1}(m||\texttt{Signature}_{k_{\text{priv}}}(\texttt{hash}(m))) \end{aligned}$$

where `ModeOfOperation-Enc` is a primitive executing the encryption mode of a block cipher, `MAC` is a generic *Message Authentication Code*, and `Signature` is a generic public-key signature scheme.

(c) Public-key algorithms are usually used for encrypting short messages. But if we need to encrypt a longer message we can split it into blocks, use RSA for each block and then use a mode of operation to compute the ciphertext.
Which of the two modes between CBC and CTR would you recommend in such a situation? Why?

Solution:

(a) The decryption equation of the CBC mode of operation is: $m_i{=}\mathrm{Dec}_k(c_i){\oplus}c_{i-1}$. A corrupted c_i implies that m_i and m_{i+1} will be wrongly decrypted; however, m_{i+2} does not depend on c_i and will be decrypted correctly. As the MAC_k primitive keeps only the last ciphertext block, the proposed scheme does not guarantee integrity of the message.

More in detail, we can do the following observations.
Given a message $m{\leftarrow}\langle m_1, m_2, \ldots, m_t\rangle$ as a sequence of t blocks, the MAC-CBC digest is computed as:

$$\begin{array}{rcl} c_0 & \leftarrow & \mathtt{IV} \\ c_i & \leftarrow & \mathrm{Enc}_k(c_{i-1} \oplus m_i),\ 1 \le i \le t \end{array}$$

Thus, $\mathtt{MAC}_k(m){=}c_t$.
The CBC-encryption transformation takes as input the $t{+}1$ blocks:
$m_1, \ldots, m_t, \mathtt{MAC}_k(\langle m_1, m_2, \ldots, m_t\rangle){=}c_t$, and computes the sequence of ciphertext blocks as:

$$\begin{array}{rcl} c_0 & \leftarrow & \mathtt{IV} \\ c_i & \leftarrow & \mathrm{Enc}_k(c_{i-1} \oplus m_i),\ 1 \le i \le t \\ c_{t+1} & \leftarrow & \mathrm{Enc}_k(c_t \oplus \mathtt{MAC}_k(\langle m_1, m_2, \ldots, m_t\rangle)) = \mathrm{Enc}_k(c_t \oplus c_t) = \mathrm{Enc}_k(0) \end{array}$$

The last block of every ciphertext will always be equal to $\mathrm{Enc}_k(0)$, irrespectively of alterations to other blocks. Indeed, assuming that the ciphertext blocks $c_0, c_1, \ldots, c_t, c_{t+1}$ are altered as c'_0, $c'_1, \ldots, c'_{t-1}, c'_t, c_{t+1}$, the receiver will derive the message m' as:

$$\begin{array}{rcl} m'_i & \leftarrow & \mathrm{Dec}_k(c'_i) \oplus c'_{i-1},\ 1 \le i \le t \\ m'_{t+1} & \leftarrow & \mathrm{Dec}_k(c_{t+1}) \oplus c'_t = 0 \oplus c'_t = c'_t \end{array}$$

To verify the authenticity of the received message, the receiver must check if $m'_{t+1} \stackrel{?}{=} \mathtt{MAC}_k(\langle m'_1, \ldots, m'_t\rangle)$. To this end he will recompute the MAC-CBC digest, obtaining: $\mathtt{MAC}_k(\langle m'_1, \ldots, m'_t\rangle){=}c'_t{=}m'_{t+1}$, thus he will consider the received message as authentic.

(b) see lectures ...

(c) Counter mode is unusable, since here decryption is the same as encryption and anyone can encrypt messages, using the receiver's public key. Hence anyone intercepting a ciphertext can decrypt it. CBC mode does not have this disadvantage, since it uses the decryption function.

Question 2 [8 pts]
Consider the ring of polynomials $\mathbb{F}_2[x]$ in the unknown x.

(a) Is $\pi(x) = x^4 + x + 1$ irreducible? Is it primitive? Prove explicitly your answer.

(b) Representing $\mathbb{F}_{2^4}$ as $\mathbb{F}_2[x]/\pi(x)$, you are planning to design a 4×4 bits S-BOX, which can be described in compact form considering the four input bits $\langle a_3, a_2, a_1, a_0\rangle$ as the coefficients of an element of $\mathbb{F}_{2^4}$, i.e., $a(x) = a_3x^3 + a_2x^2 + a_1x^1 + a_0$. Justify which one between the following three choices is the best one for the S-BOX function, taking into account their resistance to linear and differential cryptanalyses:

- SBOX-1$(a) = a^4$
- SBOX-2$(a) = a^3 + a + 1$
- SBOX-3$(a) = a^{16}$

(c) Complete SBOX-4(a), writing down its complete representation in the lookup table below.

in	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
out	0001	0001	0111	0111	0110	0110	0000	0000								

To this end,you may find useful the following equalities:
$x^4 = x + 1$
$x^5 = x^2 + x$
$x^6 = x^3 + x^2$

Let $\langle b_3, b_2, b_1, b_0\rangle$ be the output bits of the S-BOX above corresponding to the input $\langle a_3, a_2, a_1, a_0\rangle$.
Compute the linear bias for the expression $a_2 \oplus a_0 = b_3 \oplus b_0$.
Compute the probability of the differential $(\Delta a, \Delta b) = (0001, 0011)$ and of the differential $(\Delta a, \Delta b) = (0001, 0000)$.

Solution:

(a) $\pi(x)$ is primitive, and thus irreducible.

(b)
- SBOX-1$(a) = a^4$ is not a good choice, as it is an iteration of a square function – which is indeed linear – and not a good candidate for an S-BOX. More in detail: if $a(x) = a_3x^3 + a_2x^2 + a_1x^1 + a_0$; any of the scalar coefficients of $a(x)^2$ can be computed as a linear combination of the ones of $a(x)$, i.e.: $a(x)^2 \equiv_{\pi(x)} a_3x^6 + a_2x^4 + a_1x^2 + a_0 \equiv_{\pi(x)} a_3x^3 + (a_3 + a_1)x^2 + a_2x + a_2 + a_0$. Similarly, it is easy to infer that also the coefficients of $a(x)^4$ will be computed as a linear combination of the ones of $a(x)^2$ and in turn of the ones of $a(x)$.
- SBOX-3$(a) = a^{16}$ is an even worse choice, since it is the identity function.
- SBOX-2$(a) = a^3 + a + 1$ is non-linear (any coefficient of $a(x)^3$ is composed by a sum of products, where each product is obtained multiplying subsets of coefficients of $a(x)$ – thus, it is definitely better than the other ones.

(c) Here is the full S-Box:

in	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
out	0001	0001	0111	0111	0110	0110	0000	0000	0101	0101	0011	0011	0010	0010	0100	0100

Running all the computations we find that the linear bias for $a_2 \oplus a_0 = b_3 \oplus b_0$ is 0, while the differential probabilities for $(\Delta a, \Delta b) = (0001, 0011)$ and $(\Delta a, \Delta b) = (0001, 0000)$ are 0 and 1 respectively.
Therefore, we can conclude that also SBOX-4(a) is a linear S-BOX.
In particular, it is possible to infer that SBOX-4$(a) = a^2 + a + 1$ or, equivalently, SBOX-4$(\langle a_3, a_2, a_1, a_0\rangle) = \langle 0, a_3 + a_2 + a_1, a_2 + a_1, a_2 + 1\rangle$.

Question 3 [4 pts]
Consider the cyclic group $(G,\cdot)=\langle g\rangle=(\mathbb{Z}_p^*,\cdot)$, with p a large prime, and g a generator, and analyze the following digital signature scheme. The private key is defined as $k_{\text{priv}}=s\in\{0,1,\ldots,p-1\}$, while the public key is defined as $k_{\text{pub}}=g^s\in G$.

To sign a message $m\in\{0,1\}^*$, one first computes the digest $h=H(m)\in(\mathbb{Z}_{p-1}^*,\cdot)$ for some, properly defined, hash function. Obs.: $\mathbb{Z}_{p-1}^*=\{a\,|\,1\le a\le p-1\wedge \texttt{gcd}(a,p-1)=1\}$.

The signature $\sigma\in G$ is computed as $\sigma=g^{s\cdot h^{-1}}$, while the signature verification procedure executes the following check: $\sigma^h\stackrel{?}{=}k_{\text{pub}}$.

(a) Will correct signatures be accepted?

(b) Is it unfeasible to sign an arbitrary message without knowing the private key?

Solution:

(a) Yes! The signature verification procedure will accept a correct signature: $\left(g^{s\cdot h^{-1}}\right)^h=g^s=k_{\text{pub}}$

(b) No! Given an arbitrary message m_1, anyone can hash it, $h_1=H(m_1)$, compute the inverse $h_1^{-1}\bmod(p-1)$, and use the public key of a legitimate user to sign the message m_1 on her behalf as follows: $k_{\text{pub}}^{h_1^{-1}}=(g^s)^{h_1^{-1}}=g^{s\cdot h_1^{-1}}$. Note: the order of the group is a composite number. Thus, the mathematical security of the scheme depends on the efficiency of the Poligh-Hellmann attack.

Question 4 [14 pts]

(a) Apply the Pollard's ρ method to factorize the RSA modulus $n=p\cdot q=899$.
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) Describe two primality tests.

(c) Choose an admissible public exponent e between the values $e=21_{\texttt{dec}}$ and $e=143_{\texttt{dec}}$ and compute the value of the corresponding RSA private key $k_{\text{priv}}=(p,q,\varphi(n),d)$. Show every step of the computation.

(d) Sign the message $m=898_{\texttt{dec}}\in\mathbb{Z}_n$ (without employing any padding scheme) through applying the CRT. Describe each step of the procedure.

(e) Consider a double RSA encryption using two public keys $k_{\text{pub}-1}$, $k_{\text{pub}-2}$ with the same modulus N and two distinct public exponents (namely, e_1 and e_2, $\gcd(e_1,e_2)\neq 1$). Denote as d_1 and d_2 the private exponents corresponding to the the first and the second public key, respectively. A message m is encrypted using the RSA encryption transformation with the first public exponent e_1, and the result is encrypted again using e_2.
Does this scheme increase the security of the RSA cryptoscheme? Please, explain the reasons underlying your answer.

(f) Assume to work into the Montgomery domain: $(\widetilde{\mathbb{Z}}_N,+,\times)$, $N=21$

- Compute the following Montgomery multiplication assuming a binary encoding of the operands. $\widetilde{C}\leftarrow\widetilde{A}\times\widetilde{B}\bmod N$, with $\widetilde{A}=16_{\text{dec}}$ and $\widetilde{B}=15_{\text{dec}}$

Solution:

(a) see lectures ... $q=29$, $p=31$

(b) see lectures ...

(c) $\varphi(n)=840=2^3\cdot 3\cdot 5\cdot 7$, therefore the admissible value for the public exponent is $e=143_{\mathsf{dec}}\in\mathbb{Z}^*_{\varphi(n)}$.

Applying the Extended Euclidean Algorithm: $1=\gcd(\varphi(n),e)=\gcd(840,143)$

$$\text{(i)}\begin{cases} u & \leftarrow & (840,1,0) \\ v & \leftarrow & (143,0,1) \end{cases} \quad \text{(ii)}\begin{cases} q & \leftarrow & \lfloor \frac{840}{143} \rfloor = 5 \\ w & \leftarrow & (125,1,-5) \\ u & \leftarrow & (143,0,1) \\ v & \leftarrow & (125,1,-5) \end{cases} \quad \text{(iii)}\begin{cases} q & \leftarrow & \lfloor \frac{143}{125} \rfloor = 1 \\ w & \leftarrow & (18,-1,6) \\ u & \leftarrow & (125,1,-5) \\ v & \leftarrow & (18,-1,6) \end{cases}$$

$$\text{(iv)}\begin{cases} q & \leftarrow & \lfloor \frac{125}{18} \rfloor = 6 \\ w & \leftarrow & (17,7,-41) \\ u & \leftarrow & (18,-1,6) \\ v & \leftarrow & (17,7,-41) \end{cases} \quad \text{(v)}\begin{cases} q & \leftarrow & \lfloor \frac{18}{17} \rfloor = 1 \\ w & \leftarrow & (1,-8,47) \\ u & \leftarrow & (17,7,-41) \\ v & \leftarrow & (1,-8,47) \end{cases} \quad \text{(vi)}\begin{cases} q & \leftarrow & \lfloor \frac{17}{1} \rfloor = 17 \\ w & \leftarrow & (0,...,...) \\ u & \leftarrow & (\mathbf{1},\mathbf{-8},\mathbf{47}) \\ v & \leftarrow & (0,...,...) \end{cases}$$

$1=\gcd(\varphi(n),e)=\gcd(840,143)=840\cdot(-8)+143\cdot(47) \;\Rightarrow \mathbf{d}\equiv_{\varphi(n)} e^{-1}\equiv_{840} 143^{-1}\equiv_{840}\mathbf{47}$.
$k_{\text{priv}}=(31,29,840,47)$.

(d) $s=m^d \bmod n=898^{47}_{\text{dec}} \bmod 899\equiv_{899}(-1)^{47}=-1$.

Applying the CRT:

$$\begin{cases} m_p & \equiv_p & m^{d \bmod p-1} \\ m_q & \equiv_q & m^{d \bmod q-1} \end{cases} \quad \begin{cases} m_p & \equiv_{31} & (-1)^{23}\equiv_{31} -1\equiv_{31} 30 \\ m_q & \equiv_{29} & (-1)^{3}\equiv_{29} -1\equiv_{29} 28 \end{cases}$$

$m \equiv_n q\cdot(q^{-1} \bmod p)\cdot m_p + 31\cdot(p^{-1} \bmod q)\cdot m_q$
$m \equiv_{899} 29\cdot(29^{-1} \bmod 31)\cdot 30 + 31\cdot(31^{-1} \bmod 29)\cdot 28$
$m \equiv_{899} 29\cdot((-2)^{-1} \bmod 31)\cdot 30 + 31\cdot(2^{-1} \bmod 29)\cdot 28$
$\mathbf{m} \equiv_{899} 29\cdot(-16)\cdot 30 + 31\cdot 15\cdot 28 = -13920+13020 \equiv_{899} -900 \equiv_{899} \mathbf{-1}$.

(e) The general argument against double encryption is that it is subject to a *meet-in-the-middle* attack, which has time complexity similar to that of a single brute force attack. In the particular case of the RSA encryption, double encryption is also not useful at all, since the double encryption is equivalent to a single RSA encryption with public key $e=e_1\cdot e_2$ and private key $d=d_1\cdot d_2$.

(f) see at the next page

(f) $R = 2^b$ s.t. $R{>}N$, $\gcd(R,N){=}1 \Rightarrow b{=}\lceil\log_2 N\rceil{=}\lceil\log_2 21\rceil{=}5$, $R = 2^5 = 32$
$R' = R^{-1} \mod N = 32^{-1} \bmod 21 \equiv_{21} 11^{-1} \equiv_{21} 2$
$N' = -N^{-1} \mod R = (-21)^{-1} \bmod 32 \equiv_{32} 11^{-1} \equiv_{32} 3$
$N' = \langle N'_2, N'_1, N'_0\rangle_2 = \langle 011\rangle_2$. $N'_0 = 1$

$\widetilde{B} = \langle \widetilde{B}_4\widetilde{B}_3\widetilde{B}_2\widetilde{B}_1\widetilde{B}_0\rangle_2 = \langle 01111\rangle_2$,
$\widetilde{A} = \langle \widetilde{A}_4\widetilde{A}_3\widetilde{A}_2\widetilde{A}_1\widetilde{A}_0\rangle_2 = \langle 10000\rangle_2$

00000	+
00000	$\widetilde{A}_0 \cdot \widetilde{B} = 0 \cdot \langle 10101\rangle_2$
	+
00000	$t \cdot N = (N'_0 x_0) \cdot N = \langle 00000\rangle_2$
00000	>>1
00000	end of the 1st iteration
00000	+
00000	$\widetilde{A}_1 \cdot \widetilde{B} = 0 \cdot \langle 10101\rangle_2$
	+
00000	$t \cdot N = (N'_0 x_0) \cdot N = \langle 00000\rangle_2$
00000	>>1
00000	end of the 2nd iteration
00000	+
00000	$\widetilde{A}_1 \cdot \widetilde{B} = 0 \cdot \langle 10101\rangle_2$
	+
00000	$t \cdot N = (N'_0 x_0) \cdot N = \langle 00000\rangle_2$
00000	>>1
00000	end of the 3rd iteration
00000	+
00000	$\widetilde{A}_1 \cdot \widetilde{B} = 0 \cdot \langle 10101\rangle_2$
	+
00000	$t \cdot N = (N'_0 x_0) \cdot N = \langle 00000\rangle_2$
00000	>>1
00000	end of the 4th iteration
00000	+
01111	$\widetilde{A}_4 \cdot \widetilde{B} = 0 \cdot \langle 10101\rangle_2$
	+
10101	$t \cdot N = (N'_0 x_0) \cdot N = \langle 10101\rangle_2$
100100	>>1
10010	end of the 5th iteration

$\widetilde{C} = \mathbf{10010}$

$\widetilde{C} = \langle \mathbf{10010}\rangle_2 = 18_{\text{dec}} < N \Rightarrow$ no final substraction!

Validation: $\widetilde{C} = \texttt{MonPro}(16, 15) = 16 \cdot 15 \cdot 2 \bmod 21 = 480 \bmod 21 = 18$.