Modern Cryptography Jan 31, 2019

Solutions to Final Exam

Lecturer: Daniel Slamanig, TA: Karen Klein

Put your name and student ID (if applicable) on every sheet that you hand in. Leave space in the top left corner of every sheet to allow for stapling. Any permanent pen is allowed, but text written in erasable pens (e.g. pencils) will not be considered during grading.

1. (12 points)

True or False

(a) Let p be prime, then $\mathbb{Z}_p^*$ is a cyclic group of order $p-1$.

(b) The Fermat primality test on input candiate integer n picks $a \xleftarrow{\$} \{1, \dots n-1\}$ and if $a^{n-1} \neq 1 \bmod n$ outputs "composite" and "prime" otherwise. If the input n is prime, then the algorithm will err with a certain probability.

(c) Textbook RSA encryption is OW-CPA secure under the Factoring assumption.

(d) If the message space of a public key encryption scheme is large enough, even a deterministic scheme can be $\mathsf{IND\text{-}CPA}$ secure.

(e) Every $\mathsf{IND\text{-}CCA}$ secure public key encryption scheme is also $\mathsf{IND\text{-}CPA}$ secure.

(f) The hash-and-sign paradigm is used to extend signature schemes to support arbitrarily long messages.

Solution:

(a) True.

(b) False. The Fermat primality test always returns "prime" when n is prime; however for certain composite numbers (e.g., the Carmichael numbers) it also returns "prime".

(c) False. Textbook RSA encryption is OW-CPA secure under the *RSA* assumption. (It is not known whether it is secure under the Factoring assumption).

(d) False.

(e) True. To see this, assume for a contradiction that the scheme is *not* $\mathsf{IND\text{-}CPA}$ secure, and let $\mathcal{A}$ be the adversary in the $\mathsf{IND\text{-}CPA}$ game. Note that $\mathcal{A}$ works as an adversary *also* in the $\mathsf{IND\text{-}CCA}$ game (it just doesn't invoke the decryption oracle).

(f) True.

□

2. (15 points)

(a) Specify the algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ for ElGamal encryption with respect to group parameters (G, q, g), where G is a group of prime order q generated by g.

Solution: Let $\mathsf{G.Gen}$ denote the algorithm that on input a security parameter n (in unary) generates a group (G, q, g) (such that the DDH problem in G is hard for poly(n) adversaries). The algorithms for ElGamal encryption are:

- Gen takes as input a security parameter $n \in \mathbb{N}$ (in unary) and first generates the group (G, g, q) by invoking $\mathsf{G.Gen}$. Then it picks $x \in \mathbb{Z}_q$ uniformly at random and sets $y = g^x$. It returns x as the secret key and y as the public key (we assume that the group description is implicit in the keys).
- Enc takes as input the public key $y \in G$ and a message $m \in G$. It picks $r \in \mathbb{Z}_q$ uniformly at random and then returns $c = (c_1, c_2) \in G^2$ as the ciphertext, where

$$c_1 = g^r \text{ and } c_2 = m \cdot y^r.$$

- Dec takes as input a ciphertext $c = (c_1, c_2) \in G^2$ and a secret key $x \in \mathbb{Z}_q$ and returns $c_2 \cdot c_1^{-x}$ as the decrypted message.

$\square$

(b) Suppose that you use exponential ElGamal where the message m is represented as $m \in \mathbb{Z}_q$ and you encrypt g^m with the ElGamal scheme from above. Show that when you intercept a ciphertext of this scheme that encrypts an unknown message $m \in \mathbb{Z}_q$, you can change the message to $m' = 10 \cdot m$ without decrypting. What does this say about the CCA security of ElGamal (explain in one to two sentences)?

Solution: In the exponential ElGamal scheme, the encryption algorithm works as follows:

- Enc takes as input the public key $y \in G$ and a message $m \in \mathbb{Z}_q$. It picks $r \in \mathbb{Z}_q$ uniformly at random and then returns $c = (c_1, c_2) \in G^2$ as the ciphertext, where

$$c_1 = g^r \text{ and } c_2 = g^m \cdot y^r.$$

(Note that decryption might not be efficient in general.) To convert a ciphertext $c = (c_1, c_2)$ for a message m into a ciphertext $c' = (c'_1, c'_2)$ for the message $10 \cdot m$, one has to set $c'_1 = c_1^{10}$ and $c'_2 = c_2^{10}$. This is a valid ciphertext for m' since:

$$\begin{aligned} c' &= (c'_1, c'_2) = (c_1^{10}, c_2^{10}) \\ &= (g^r)^{10}, (g^m \cdot y^r)^{10} \\ &= (g^{10\cdot r}, g^{10\cdot m} \cdot y^{10\cdot r}) \\ &= (g^{r'}, g^{m'} \cdot y^{r'}) \end{aligned}$$

where $r' = 10 \cdot r$.

The algebraic manipulation shown above renders exponential ElGamal prone to chosen-ciphertext attacks: the attacker has to simply take the challenge ciphertext c (for some message m), turn it into the ciphertext c' (for the message $m' = 10 \cdot m$). The attacker can then pass c' to the decryption oracle to learn the underlying message $10 \cdot m$, and subsequently retrieve m (as 10 is invertible modulo q). $\square$

3. (10 points)

Write down the OW-CPA and the IND-CPA experiments for public key encryption. Show, by providing an explicit reduction, that IND-CPA security implies OW-CPA security.

Solution: In the OW-CPA experiment $\mathsf{PubK}^{\mathsf{ow-cpa}}_{\mathcal{A},\Pi}(n)$, first, the key generation algorithm is run to obtain a key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)$, a message $m \leftarrow \mathcal{M}$ is chosen uniformly at random from the message space $\mathcal{M}$, and an encryption $c^* \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m)$ of m under the public key pk is generated. Then pk and c^* are sent to the adversary $\mathcal{A}$. Since $\mathcal{A}$ knows pk, it can obtain encryptions of arbitrary messages by running $\mathsf{Enc}_{\mathsf{pk}}$. Finally, $\mathcal{A}$ outputs a message m^*. The output of the experiment is 1 if and only if $m^* = m$, and 0 otherwise.
Note, the OW-CPA experiment is a *search game* where the adversary has to find the correct solution m^* from an exponentially large set of possible messages. A public-key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ has one-way encryptions in the presence of an eavesdropper if for all probabilistic polynomial-time adversaries $\mathcal{A}$ there is a negligible function negl s.t.

$$\Pr[\mathsf{PubK}^{\mathsf{ow-cpa}}_{\mathcal{A},\Pi}(n) = 1] \leq \mathsf{negl}(n).$$

In the IND-CPA experiment $\mathsf{PubK}^{\mathsf{ind-cpa}}_{\mathcal{A},\Pi}(n)$, first, the key generation algorithm is run to obtain a key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)$, and the public key pk is sent to the adversary $\mathcal{A}$. Since $\mathcal{A}$ knows pk, it can obtain encryptions of arbitrary messages by running $\mathsf{Enc}_{\mathsf{pk}}$. Then the adversary chooses two arbitrary messages m_0, m_1 from the message space $\mathcal{M}$. Upon receipt of m_0, m_1, a bit $b \leftarrow \{0, 1\}$ is chosen uniformly at random, and an encryption $c^* \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m_b)$ of m under the public key pk is generated. Then c^* is sent to the adversary $\mathcal{A}$. Finally, $\mathcal{A}$ outputs a bit b^*. The output of the experiment is 1 if and only if $b^* = b$, and 0 otherwise.
Note, the IND-CPA experiment is a *decision game* where the adversary knows it is in one of two worlds and has to guess which one it is. Thus, it can always just guess uniformly at random and win with probability $\frac{1}{2}$. A public-key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ has indistinguishable encryptions in the presence of an eavesdropper if for all PPT adversaries $\mathcal{A}$ there is a negligible function negl such that

$$\Pr[\mathsf{PubK}^{\mathsf{ind-cpa}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathsf{negl}(n).$$

We prove that IND-CPA security implies OW-CPA security: Let $\mathcal{A}^{\mathsf{ow}}$ be an arbitrary adversary against OW-CPA security of a public-key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with exponentially large message space $|\mathcal{M}| = \Omega(\mathsf{exp}(n))$. We construct an adversary $\mathcal{A}^{\mathsf{ind}}$ against IND-CPA security as follows. Upon receipt of the public key pk, $\mathcal{A}^{\mathsf{ind}}$ chooses $m_0, m_1 \leftarrow \mathcal{M}$ uniformly at random and sends them to the challenger. After receiving the challenge ciphertext $c^* \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m_b)$, it runs $\mathcal{A}^{\mathsf{ow}}$ on pk, c^* and gets some output m^*. If $m^* = m_0$, $\mathcal{A}^{\mathsf{ind}}$ outputs $b^* := 0$, otherwise $b^* := 1$.
We analyse the success probability of the algorithm $\mathcal{A}^{\mathsf{ind}}$:

$$\Pr[\mathcal{A}^{\mathsf{ind}} \text{ wins}] = \Pr[\mathcal{A}^{\mathsf{ow}} \text{ wins } \wedge b = 0] + \Pr[\mathcal{A}^{\mathsf{ow}} \text{ wins } \wedge b = 1 \wedge m_0 \neq m_1]$$

$$+\Pr[\mathcal{A}^{\mathsf{ow}} \text{ fails } \wedge b = 1 \wedge m^* \neq m_0]$$

$$= \frac{1}{2}\mathsf{Pr}[\mathcal{A}^{\mathsf{ow}} \text{ wins }] + \frac{1}{2}\frac{|\mathcal{M}|-1}{|\mathcal{M}|}\mathsf{Pr}[\mathcal{A}^{\mathsf{ow}} \text{ wins }] + \frac{1}{2}\frac{|\mathcal{M}|-1}{|\mathcal{M}|}(1 - \mathsf{Pr}[\mathcal{A}^{\mathsf{ow}} \text{ wins }])$$

$$= \frac{1}{2} + \frac{1}{2}\mathsf{Pr}[\mathcal{A}^{\mathsf{ow}} \text{ wins }] - \frac{1}{2|\mathcal{M}|}.$$

By IND-CPA security of Π there exists a negligible function negl such that

$$\mathsf{Pr}[\mathcal{A}^{\mathsf{ind}} \text{ wins}] \leq \frac{1}{2} + \mathsf{negl}(n).$$

Hence, $\mathsf{Pr}[\mathcal{A}^{\mathsf{ow}} \text{ wins }] \leq 2\mathsf{negl}(n) + \frac{1}{|\mathcal{M}|}$, which is negligible. This proves the claim. □

4. (10 points)

Consider the following variant of the Schnorr signature scheme. Let $\mathsf{pp} := (G, q, g) \leftarrow \mathcal{G}(1^n)$ be public group parameters that are the description of a cyclic group G of prime order q and a generator g. These parameters pp are implict input to the Sign and Vrfy algorithms. Furthermore, we fix hash functions $H_1 : \{0,1\}^* \to \mathbb{Z}_q$ and $H_2 : \mathbb{Z}_q \to \mathbb{Z}_q$.

Gen: on input pp, choose $x, z \xleftarrow{\$} \mathbb{Z}_q$, compute $y := g^x$, $w := x+z \bmod q$ and output the public and private key $(\mathsf{pk}, \mathsf{sk}) := ((y, w), (x, z))$.

Sign: on input a secret key $\mathsf{sk} = (x, z)$ and a message $m \in \{0,1\}^*$, compute $k := H_2(z)$, $I := g^k$, $r := H_1(I, m)$ and $s := rx + k \bmod q$. Output the signature $\sigma := (r, s)$.

Vrfy: on input a public key $\mathsf{pk} = (y, w)$ a message $m \in \{0,1\}^*$, a signature $\sigma = (r, s)$, compute $I := g^s \cdot y^{-r}$ and output 1 if $H_1(I, m) = r$.

Construct an adversary that can recover the secret key (x, z) of the scheme given the public key pk and two valid message-signature pairs (m, σ) and (m', σ') with $m \neq m'$.

Solution: Let (m, σ) and (m', σ') with $m \neq m'$ be two valid signature pairs. We write the signatures as

$$\sigma := (r, s) = (r, rx + k) \text{ and } \sigma' := (r', s') = (r', r'x + k).$$

Since $k := H_2(z)$ is the same in both expressions and $r := H_1(I, m) \neq H_1(I, m') =: r'$ by collision resistance of H_1, we can compute x as

$$x = (s - s')(r - r')^{-1} \bmod q.$$

Given the public key $\mathsf{pk} = (y, w)$ with $w := x + z \bmod q$, we can now also compute the second part of the secret key as $z = w - x \bmod q$. □

5. (10 points)

(a) What is the random oracle model (ROM)? Explain the term “programming” in context of random oracles and how is this technique used?

Solution: A random oracle (RO) is a truly random function. One can think of a RO as a black-box that for every input produces a uniformly random output (while maintaining consistency by storing previous input/output pairs and responding accordingly to repeated queries). In the random oracle model we assume that every entity has access to this function. It is used to model cryptographic hash functions in situations, where we do not know how to prove security in the standard model (i.e. without the presence of a random oracle).

When using a reduction to transform an adversary against a scheme using the RO into an adversary against a different primitive, the reduction needs to simulate the RO. This is usually done by emulating the procedure above, i.e. for every query of the adversary the reduction chooses a uniformly random value, stores the input/output pair in a table and returns the corresponding value. Programming the RO refers to the possibility of inserting a specific value instead of choosing it uniformly at random for some input queried by the adversary. This can be done as long as this value is also uniformly distributed. This is usually used to embed the challenge the reduction is trying to solve, which then prompts the adversary to aid in this task. □

(b) The Diffie-Hellman key-exchange protocol only provides passive security (i.e., security in the presence of an eavesdropper). Which measures could be taken to obtain active security (i.e., counter man-in-the-middle attacks)? Would your so obtained version of the protocol intuitively also provide forward-secrecy? Forward-secrecy means that breaking into the server and receiving the server state does not allow to compute shared keys from previous sessions.

Solution: In order to achieve active security, we need to provide some authentication. This can be done using signatures, where both parties sign their shares $A = g^a$ and $B = g^b$ with their respective secret signing keys. Before processing the message of the other party, both parties verify the correctness of the signatures using the corresponding public keys and abort, if the check fails. Of course, this assumes that some PKI is in place.

For forward-secrecy, observe that the DH protocol does have this property as long as the keys are not stored on the server beyond the session. Our modification from above now adds a long term secret key, which could be stolen by an adversary breaking into the server. However, since the key is only used for signing, it can only be used to compromise future sessions. Past sessions remain secure and thus our protocol achieves forward-secrecy. □