

## Solutions to homework 2

**1 Security Definitions [10+20 points]**

Definition of some security property often goes like this: We call some communication scheme  $\Sigma$  *secure in the sense of resistance against attack of type “X”* if for all probabilistic polynomial time algorithms  $A$ , the probability that  $A$  succeeds in an “*attack of type X*” against  $\Sigma$  is negligibly small, i.e. it’s a negligible function of the security parameter  $\tau$ .

For example, the definition of *one-way secure* encryption scheme  $\Sigma = (KGen, Enc, Dec)$  has exactly this form, where “*attack of type X*” of  $A$  against  $\Sigma$  is the “*decryption attack*”, defined as follows: (1)  $KGen$  is executed on the security parameter  $\tau$  to create key  $k$ , (2) random  $m$  is picked in the messages space  $\mathcal{M}$ , (3) ciphertext  $c$  is computed as  $Enc(k, m)$ , and finally (4)  $A$  runs on input  $c$  and outputs some string  $m'$ . We say that  $A$  succeeds in this attack if  $m' = m$ .

**1.1 [10 points]**

Show a (trivial) PPT algorithm which succeeds with a non-zero but negligible probability in an attack against the “one way security” property of the one-time pad encryption scheme defined for message space  $\mathcal{M} = \{0, 1\}^\tau$  and key space  $\mathcal{K} = \{0, 1\}^\tau$ , where  $\tau$  is the security parameter.

Note that this means that even if a scheme is *perfectly secure*, let alone *one-way secure*, there nevertheless usually exist efficient attacks against it which succeed with *negligible* probability. This, in part, is why we usually cannot ask that that the probability of successful break of our scheme be zero for all efficient algorithms.

**Solution:** The attack algorithm  $A$ , on input  $c = Enc(k, m) = k \otimes m$ , for  $k, m \in \{0, 1\}^\tau$ , simply outputs a random string  $m' \leftarrow \{0, 1\}^\tau$ .  $A$  succeeds in inverting the one-time pad encryption if  $m' = m$ .  $A$  is PPT because guessing a  $\tau$ -long string takes  $O(\tau)$  time, while its probability of success is

$$\begin{aligned} Adv_A(\tau) &= Prob[m' = m \mid k \leftarrow \mathcal{K}; m \leftarrow \mathcal{M}; c \leftarrow k \oplus m; m' \leftarrow A(c)] \\ &= Prob[m' = m \mid m \leftarrow \{0, 1\}^\tau; m' \leftarrow \{0, 1\}^\tau] \\ &= 2^{-\tau} \end{aligned}$$

which is non-zero but negligible. □

**1.2 [bonus 20 points]**

Let’s show that the definitions of this type are “robust” in the following sense: Assume that a scheme  $\Sigma$  is secure against “attack of type X” in the above sense, but that there nevertheless exists an efficient algorithm  $A$  which *does* succeed in this attack but only with a negligible probability, for example  $2^{-p(\tau)}$  for some polynomial  $p(\cdot)$ .

Consider a new efficient attack algorithm  $A'$ , which simply runs attack  $A$  for some polynomial number of times, say  $p'(\tau)$ , and succeeds if *any* of these runs of  $A$  return a successful output. Argue why  $A'$  is an *efficient* algorithm, and show that such polynomial-number of repetitions of attack against  $\Sigma$  still has only negligible probability of success.

*[[Hint: First of all, your goal is to argue that the probability that  $A'$  succeeds is smaller than some negligible function for all large enough  $\tau$ , i.e. for all  $\tau$  larger than some  $\tau_0$ . Therefore all intermediate steps you make do not have to hold for all  $\tau$ 's, but only for all sufficiently large  $\tau$ 's. A convenient way to do this is to look at the probability that  $A'$  fails, and try to show that for all large enough  $\tau$ 's, this probability is larger than  $1 - \epsilon$ , where  $\epsilon$  is some conveniently chosen negligible function.*

*You might use the following facts: (1)  $(1 - \frac{1}{a_n})^{a_n} \approx \frac{1}{e}$  for any  $a_n \rightarrow \infty$ , where  $e = 2.718\dots$ , (2)  $\frac{1}{4} < \frac{1}{e} < \frac{1}{2}$ , (3)  $(x^y)^z = x^{yz}$  for all  $x, y, z$ , and therefore also  $x^z = (x^y)^{(z/y)}$  for all  $x, y, z$ , (4) for all  $x, y, c > 0$  inequality  $x > y$  holds if and only if  $x^c > y^c$ , (5) for any polynomials  $g(\cdot), g'(\cdot)$ , inequality  $2^{g(\tau)} > g'(\tau)$  holds for all large enough  $\tau$ 's, (6) therefore, in particular, for any polynomial  $g(\cdot)$ , function  $1/2^{g(\tau)}$  is negligible.]]*

**Solution:** This was definitely more complicated, but read the solution below because this type of argument will come up, except that we'll try to make it easier next time.

We'll show that the advantage of  $A'$  must be smaller than  $\epsilon(\tau)$  for some negligible function  $\epsilon(\cdot)$  by showing that the probability that  $A'$  *fails* is larger than  $1 - \epsilon(\tau)$ .

We do it this way because looking first at the probability of *failure* of  $A'$  rather directly at the probability of its success is easier: The probability that  $A'$  fails is the probability that  $A$  fails on *each* of the  $p'(\tau)$  trials. So we have:

$$\begin{aligned}
 1 - Adv_{A'}(\tau) &= Prob[A' \text{ fails}] \\
 &= (Prob[A \text{ fails}])^{p'(\tau)} \\
 &= (1 - Prob[A \text{ succeeds}])^{p'(\tau)} \\
 &\geq \left(1 - \frac{1}{2^{p(\tau)}}\right)^{p'(\tau)} \\
 &= \left(\left(1 - \frac{1}{2^{p(\tau)}}\right)^{2^{p(\tau)}}\right)^{p'(\tau)/2^{p(\tau)}} \\
 &\approx \left(\frac{1}{e}\right)^{p'(\tau)/2^{p(\tau)}} \\
 &> \left(\frac{1}{4}\right)^{p'(\tau)/2^{p(\tau)}} \\
 &= 2^{-2p'(\tau)/2^{p(\tau)}} \\
 &= 2^{-a_\tau} \quad \text{where } a_\tau = 2p'(\tau)/2^{p(\tau)}
 \end{aligned}$$

And now comes the grungy part. Namely, we want to show that the above expression is actually “negligibly close” to 1. The reason is that exponent  $a_\tau = 2p'(\tau)/2^{p(\tau)}$  approaches 0 very fast, because the exponentiation function  $2^{p(\tau)}$  in the denominator becomes much larger than the polynomial nominator  $2p(\tau)$  for large enough  $\tau$ . And since  $a_\tau \rightarrow 0$  therefore  $2^{-a_\tau} \rightarrow 1$ . But the limits are not enough: We need to show the our failure probability not only

approaches 1 but that it approaches it fast enough so that  $Adv_A(\tau) = 1 - Prob[A' \text{ fails}] \leq 1 - 2^{-a\tau}$  is a negligible function of  $\tau$ .

Showing this is actually pretty grungy, and we'll definitely try to avoid such stuff in the future. But if you are curious how it can be done it's not that hard (just grungy!) and I put it in the appendix.  $\square$

## 2 One-way security vs. indistinguishability [20 points]

Recall the definitions of “one way security” and “indistinguishability” of an encryption scheme. Recall the proof we did in the lecture which showed that if one-way secure encryption schemes exist at all, then there can be an encryption scheme which is one-way secure but not indistinguishable.

Now prove that if an encryption scheme is secure in the sense of indistinguishability then it is also secure in the sense of one-wayness for message space  $\mathcal{M} = \{0, 1\}^\tau$ . This will show that indistinguishability is a strictly stronger security property of encryption than one-wayness.

You can prove this by proving the counterpositive, i.e. assume that some encryption scheme  $\Sigma$  is *not* one-way secure, and then show that it is also *not* indistinguishable. If you assume that  $\Sigma$  is not one-way secure then there exists a PPT algorithm  $A$  that breaks one-wayness of  $\Sigma$ . Using such  $A$  construct a PPT algorithm  $A'$  s.t.

$$Prob[A'(m_0, m_1, c) = 1 \mid k \leftarrow KGen(1^\tau), (m_0, m_1) \leftarrow \{0, 1\}^\tau, c \leftarrow Enc(k, m_1)]$$

is non negligible, while

$$Prob[A'(m_0, m_1, c) = 1 \mid k \leftarrow KGen(1^\tau), (m_0, m_1) \leftarrow \{0, 1\}^\tau, c \leftarrow Enc(k, m_0)]$$

is negligible.

Argue that the existence of  $A'$  leads to breaking the indistinguishability property of  $\Sigma$ .

**Solution:** There are many ways to do this, and here is one. Let's construct algorithm  $A'$  as follows: On input  $(m_0, m_1, c)$ ,  $A'$  will run algorithm  $A(c)$  and if  $A$  outputs  $m_1$  then  $A'$  will output 1, and in every other case  $A'$  will output 0. What we want is that

1.  $A'$  should vote 1 on encryptions  $c = Enc(k, m_1)$  with some decent probability
2. while the probability that  $A'$  votes 1 on encryptions  $c = Enc(k, m_0)$  should be negligible.

And if  $A$  has a non negligible advantage  $Adv_A(\tau)$  in attacking one-wayness of encryption scheme  $\Sigma$  then the above  $A'$  actually manages to achieve both requirements because:

$$\begin{aligned} Prob[A'(m_0, m_1, c) = 1 \mid k \leftarrow KGen(1^\tau), (m_0, m_1) \leftarrow \{0, 1\}^\tau, c \leftarrow Enc(k, m_1)] &= \\ Prob[A(c) = m_1 \mid k \leftarrow KGen(1^\tau), (m_0, m_1) \leftarrow \{0, 1\}^\tau, c \leftarrow Enc(k, m_1)] &= \\ Prob[A(c) = m_1 \mid k \leftarrow KGen(1^\tau), m_1 \leftarrow \{0, 1\}^\tau, c \leftarrow Enc(k, m_1)] &= \\ Prob[A(c) = m \mid k \leftarrow KGen(1^\tau), m \leftarrow \{0, 1\}^\tau, c \leftarrow Enc(k, m)] &= \\ &= Adv_A(\tau) \end{aligned}$$

which satisfies requirement (1).

Requirement (2) is satisfied too because  $A'$  outputs 1 on  $c = Enc(k, m_0)$  only if  $A(c)$  outputs  $m_1$ , i.e.

$$\begin{aligned} Prob[A'(m_0, m_1, c) = 1 \mid k \leftarrow KGen(1^\tau), (m_0, m_1) \leftarrow \{0, 1\}^\tau, c \leftarrow Enc(k, m_0)] &= \\ Prob[A(c) = m_1 \mid k \leftarrow KGen(1^\tau), (m_0, m_1) \leftarrow \{0, 1\}^\tau, c \leftarrow Enc(k, m_0)] & \end{aligned}$$

But notice that  $A$ 's input  $c = Enc(k, m_0)$  is formed independently of the choice of  $m_1$ . And since  $A$ 's input is independent of the choice of  $m_1$ , then  $A$ 's output has to be independent of  $m_1$  as well. Simply,  $A$  does not get any information about  $m_1$ . So the probability that  $A$ 's output happens to be equal to some  $m_1$  chosen at random in  $\mathcal{M} = \{0, 1\}^\tau$  can be at most  $|\mathcal{M}| = 1/2^\tau$ .

Therefore we have

$$\begin{aligned} Prob_{\substack{k \leftarrow KGen(1^\tau); (m_0, m_1) \leftarrow \{0, 1\}^\tau \\ c \leftarrow Enc(k, m_1)}} [A'(m_0, m_1, c) = 1] &- Prob_{\substack{k \leftarrow KGen(1^\tau); (m_0, m_1) \leftarrow \{0, 1\}^\tau \\ c \leftarrow Enc(k, m_0)}} [A'(m_0, m_1, c) = 1] &\geq \\ &\geq Adv_A(\tau) - 1/2^\tau \end{aligned}$$

which is a non negligible function of  $\tau$  because  $Adv_A(\tau)$  is not negligible (and  $1/2^\tau$  is negligible).

Why does this imply an attack on the indistinguishability of the encryption scheme  $\Sigma$ ? Because it shows that an efficient PPT  $A'$  can, with higher than negligible probability, distinguish encryptions of  $m_1$  from encryptions of  $m_0$  for *random* pair  $(m_0, m_1)$  of messages in  $\{0, 1\}^\tau$ . But if the distinguishing advantage of  $A'$  is good on random pairs of messages, there must in particular exist *some* pair of messages  $(m_0, m_1)$  which maximizes this probability, or at least achieves the average. In any case, there exists some two messages  $m_0, m_1 \in \{0, 1\}^\tau$ , and hence  $|m_0| = |m_1|$ , such that  $A$  has a higher than negligible advantage in distinguishing ciphertexts  $c = Enc(k, m_0)$  from ciphertexts  $c = Enc(k, m_1)$ , which violates the indistinguishability property of the encryption scheme  $\Sigma$ .  $\square$

### 3 Inverting ElGamal cryptosystem [25 points]

Consider (a variant of) an ElGamal cryptosystem, where  $SK = (p, g, y, x)$  where  $p$  is a large prime of size polynomial in the security parameter  $\tau$  (similarly to the RSA modulus, 1024-bit  $p$  is believed to offer about  $2^{80}$  security, and therefore is often used today),  $g$  is a *generator* of group  $\mathbb{Z}_p^*$ ,  $x$  is a random number in  $\mathbb{Z}_{p-1}$ , and  $y = g^x \bmod p$ . (See the handout on arithmetic modulo primes.)<sup>1</sup> The public key used for encryption is  $PK = (p, q, y)$  and  $SK = (p, q, y, x)$  is used to decrypt.

The encryption works as follows. The input message is broken-down into blocks  $m$  s.t. each  $m \in \mathbb{Z}_p^*$  and then each  $m$  is encrypted individually as follows:

- $Enc(PK, m) = (c_1, c_2) = (g^r \bmod p, y^r * m \bmod p)$ , where  $r$  is a random number in  $\mathbb{Z}_{p-1}$  picked by the encryption algorithm. (Each ciphertext is a pair of  $(c_1, c_2) \in (\mathbb{Z}_p^*, \mathbb{Z}_p^*)$ .)

---

<sup>1</sup>In particular, recall that  $\mathbb{Z}_p^* = \{1, \dots, p-1\}$ ,  $\mathbb{Z}_{p-1} = \{0, \dots, p-2\}$ , and that if  $g$  is a generator than for every element  $y \in \mathbb{Z}_p^*$  there is a unique element  $x \in \mathbb{Z}_{p-1}$  s.t.  $y = g^x \bmod p$ . Therefore, in particular, if you pick  $x$  at random in  $\mathbb{Z}_{p-1}$ , element  $y = g^x \bmod p$  is itself random, i.e. uniformly distributed, in  $\mathbb{Z}_p^*$ .

- $m = Dec(SK, (c_1, c_2)) = c_2 / (c_1)^x \pmod p$ . (Check that this comes out right!)<sup>2</sup>

Assume that someone creates an (efficient) algorithm  $A$  which decrypts ElGamal ciphertexts knowing just the public key, but only if  $c_1$  starts with at least 5 leading zeroes, i.e.  $c_1 = 00000\dots$ .

What's the advantage of  $A$  in breaking the one-wayness of ElGamal? Is it negligible?

**Solution:** Let's call  $BC$ , "bad ciphertexts", the set of all elements  $c_1$  in  $\mathbb{Z}_p^*$  s.t. the leading 5 bits of  $c_1$  are zeroes. Since the exponentiation function

$$Exp_{(p,g)}(r) = g^r \pmod p$$

is a one-to-one function from  $\mathbb{Z}_{p-1}$  to  $\mathbb{Z}_p^*$ , a uniform choice of randomness  $r \leftarrow \mathbb{Z}_{p-1}$  chosen by the encryption procedure implies that values  $c_1 = Exp_{(p,g)}(r)$  are uniformly distributed in the range  $\mathbb{Z}_p^*$  of function  $Exp_{(p,g)}$ . Therefore the probability that  $c_1 \in BC$  for  $(c_1, c_2)$  which is a random ElGamal encryption of any message  $m$ , is the probability that a randomly chosen  $c_1 \in \mathbb{Z}_p^*$  belongs to  $BC$ , which is  $|BC|/|\mathbb{Z}_p^*| > 1/2^5$ . (This probability is not exactly equal to  $1/2^5$  because if we represent elements in  $\mathbb{Z}_p^*$  as bitstrings in  $\{0, 1\}^{p-1}$ , those bitstrings in  $\{0, 1\}^{p-1}$  which represent numbers greater or equal to  $p$  will never occur, so numbers with a leading 0 bit are always somewhat more likely.)

Finally, since  $1/2^5 = 1/32$  is a constant and a constant function is higher than a negligible function,  $A$ 's probability of success is non negligible.  $\square$

It seems that one could counteract such an attack by modifying ElGamal encryption so that  $c_1$  never starts with 00000 substring. The encryption procedure could simply pick another  $r$  if  $c_1$  happens to start with 00000.

However, show how that algorithm  $A$  can be used as a black box to construct an algorithm  $A'$  which (efficiently) decrypts *every* ciphertext, and not just those s.t.  $c_1$  starts with 5 zeroes. What's the running time of  $A'$  compared to the running time of  $A$ ?

**Solution:** Here's what  $A'$  can do (this is not the only solution!): Faced with a ciphertext  $c = (c_1, c_2)$ ,  $A'$  can *map* it to another ciphertext  $c' = (c'_1, c'_2)$  in such a way so that:

1. Ciphertext  $c'$  can be decrypted by  $A$ , i.e.  $c'_1 \in BC$ .
2.  $A'$  can *translate* the decryption of  $c'$  into a decryption of  $c$ .

Here's one way to do such mapping. Take a random  $\delta \leftarrow \mathbb{Z}_{p-1}$  and compute  $c'_1 = c_1 g^\delta \pmod p$  and  $c'_2 = c_2 y^\delta \pmod p$ . Notice that if  $(c_1, c_2) = (g^r \pmod p, y^r m \pmod p)$  for a random  $r \in \mathbb{Z}_{p-1}$  then

$$(c'_1, c'_2) = (g^r g^\delta, y^r m y^\delta) = (g^{(r+\delta \pmod{p-1})}, y^{(r+\delta \pmod{p-1})} m) = (g^{r'}, y^{r'} m)$$

(where all exponentiations and multiplications are done modulo  $p$ ), for  $r' = r + \delta \pmod{p-1}$ .

Notice that for any  $r \in \mathbb{Z}_{p-1}$ , if  $\delta$  is picked uniformly at random in  $\mathbb{Z}_{p-1}$  then  $r' = r + \delta \pmod{p-1}$  is also uniformly distributed in  $\mathbb{Z}_{p-1}$ . (This is a similar argument to that

---

<sup>2</sup>Note: The security of ElGamal rests on the difficulty of computing discrete logarithms, because otherwise someone could compute  $x = DL_g(y)$  and decrypt on their own. But in fact ElGamal rests on a stronger assumption of its own, namely that given  $(p, g)$  and random  $c_1, y \in \mathbb{Z}_p^*$ , it is hard to compute  $c_1^x \pmod p$  without knowing  $x$  s.t.  $y = g^x \pmod p$ , i.e. without knowing  $x = DL_g(y)$ . This is similar to the case of RSA, which definitely needs the difficulty of factoring to be secure, but really requires a stronger assumption of its own.

we used for the One Time Pad encryption where for *any*  $n$ -bit message string, picking key  $k$  at uniform in the set of  $n$ -bit strings results in ciphertext  $c = m \oplus k$  being uniformly distributed in  $\{0, 1\}^n$  as well.)

And why do we want  $r'$  to be uniform in  $\mathbb{Z}_{p-1}$  for any  $r$ , and thus for any  $c_1$ ? Because this means that  $c'_1 = \text{Exp}_{(p,g)}(r')$  is uniformly distributed in  $\mathbb{Z}_p^*$  (because the exponentiation function is one-to-one, as we argued above). And therefore the probability that  $c'_1$  falls in set  $BC$  is again  $|BC|/|\mathbb{Z}_p^*| > 1/2^5$ .

So, what  $A'$  should really do is to pick  $\delta$ 's at random in  $\mathbb{Z}_{p-1}$  until  $c'_1 = c_1 g^\delta$  is in  $BC$ . The expected number of such tests is at most  $(\text{Prob}[c'_1 \in BC])^{-1} < 2^5$ .

And what if  $A'$  finds  $c'_1 \in BC$ ? It then runs  $A$  on  $c' = (c'_1, c'_2)$  and, since  $c'_1 \in BC$ ,  $A'$  gets back

$$\text{Dec}(SK, c') = \frac{c'_2}{c'_1} = \frac{c_2 y^r}{(c_1 g^r)^x} = \frac{c_2 (g^x)^r}{c_1^x g^{rx}} = \frac{c_2}{c_1^x} * \frac{g^{xr}}{g^{rx}} = \frac{c_2}{c_1^x} = \text{Dec}(SK, c) = m$$

So the *translation* part is very easy: In fact  $c'$  is a ciphertext of the same plaintext as  $c$ , and hence the decryption of  $c'$  is also a decryption of  $c$ .

So  $A'$  succeeds in decrypting *any* ElGamal ciphertext. Finally, the running time of  $A'$  is equal to the running time of  $A$  plus the expected time of at most 32 modular exponentiations and multiplications (one modular exp and one modular mult to compute  $c'_1 = c_1 g^\delta \pmod p$ ), which is a polynomial in the security parameter  $\tau$ , because  $|p| = \text{poly}(\tau)$ . This shows that  $A'$  is a PPT algorithm if  $A$  is.  $\square$

## A The “Grungy” Part of the Bonus Problem

Here’s how we argue that  $\text{Adv}_{A'}(\tau) \leq 1 - 2^{a_\tau}$  is a negligible function of  $\tau$ , for  $a_\tau = 2p'(\tau)/2^{p(\tau)}$ :

We know that for all large enough  $\tau$ 's we have

$$2^{p(\tau)/2} \text{geq} 2p'(\tau)$$

Therefore for all large enough  $\tau$ 's we have

$$2^{p(\tau)}/2p'(\tau) > 2^{p(\tau)/2}$$

And hence for all large enough  $\tau$ 's we have

$$a_\tau = 2p'(\tau)/2^{p(\tau)} < 1/2^{p(\tau)/2}$$

This shows that for polynomial  $g(\tau) = p(\tau)/2$ , we have that for all large enough  $\tau$ 's:

$$1 - \text{Adv}_{A'}(\tau) > 2^{-a_\tau} > 2^{-1/2^{g(\tau)}}$$

Therefore for all large enough  $\tau$  we have

$$\text{Adv}_{A'}(\tau) < 1 - 2^{-1/2^{g(\tau)}}$$

We’ll show that this is actually less than a negligible function by showing that for all large enough  $\tau$  we have

$$1 - 2^{-1/2^{g(\tau)}} < 2^{-f(\tau)} \tag{1}$$

for some polynomial  $f(\cdot)$ .

Why is equation (1) true? It holds if and only if for all large  $\tau$  we have

$$2^{-1/2^{g(\tau)}} > 1 - 2^{-f(\tau)}$$

which holds if and only if for all large  $\tau$  we have

$$(1/2) > \left(1 - 2^{-f(\tau)}\right)^{2^{g(\tau)}} \quad (2)$$

But we know that for large enough  $\tau$ :

$$\left(1 - 2^{-f(\tau)}\right)^{2^{g(\tau)}} = \left(1 - \frac{1}{2^{f(\tau)}}\right)^{2^{f(\tau)} * 2^{g(\tau)} / 2^{f(\tau)}} \approx \left(\frac{1}{e}\right)^{2^{g(\tau)-f(\tau)}}$$

And therefore inequality (2) holds for example for  $f(\tau) = g(\tau)$ , because  $1/2 > 1/e = (1/e)^1 = (1/e)^{2^0}$ . And since inequality (2) holds then so does inequality (1), and we are done.