The exam grade follows a linear scale in which each question has the same weight.

1 Σ Protocol for Discrete Log Equality

We assume that public parameters $pp$ describe a group, how to do operations and comparison in the group, and also give its prime order $p$. We use additive notation and 0 denotes the neutral element in the group. We define the relation $R((pp, G, X, Y, Z), x)$ for group elements $G, X, Y, Z$ and an integer $x$ which is true if and only if $G \neq 0$, $X = xG$, and $Z = xY$. We construct a Σ-protocol for $R$ with challenge set $\mathbb{Z}_p$. The prover starts by picking $k \in \mathbb{Z}_p$ with uniform distribution, computing and sending $A = kG$ and $B = kY$. Then, the prover gets a challenge $e \in \mathbb{Z}_p$. The answer is an integer $z$ to be computed in a way which is a subject of the following question. The final verification is also a subject of the following question. The protocol looks like this:

Q.1 Inspired by the Schnorr proof, finish the specification of the prover and the verifier.

---

Essentially, we do a Schnorr proof in the group of $(X, Z)$ pairs. That is, we prove knowledge of $x$ such that $(X, Z) = x(G, Y)$. Based on that, the prover sends $(A, B) = k(G, Y)$, gets $e$, and answers by $z = k + ex \mod p$. The final verification is $z(G, Y) = (A, B) + e(X, Z)$, i.e. $zG = A + eX$ and $zY = B + eZ$. The verifier should verify $G \neq 0$ too.
Q.2 Specify the extractor and the simulator.

Given two valid transcripts \((A, B, e_1, z_1)\) and \((A, B, e_2, z_2)\) with the same \((A, B)\) and different \(e_1 \neq e_2\), we set

\[
x = \frac{z_2 - z_1}{e_2 - e_1} \mod p
\]

and we prove \((X, Z) = x(G, Y)\) like in the Schnorr proof. Given \(e\) and a random \(z\), we define \((A, B) = z(G, Y) - e(X, Z)\) and obtain a simulated transcript \((A, B, e, z)\) with same distribution, like in the Schnorr proof:

\[
x(G, Y) = \frac{1}{e_2 - e_1}(z_2(G, Y) - z_1(G, Y))
\]

\[
= \frac{1}{e_2 - e_1}((A, B) + e_2(X, Z) - (A, B) - e_1(X, Z))
\]

\[
= (X, Z)
\]

Frequent mistake in exams: writing \(z_i = k + e_i x\) is incorrect because the prover is malicious and there is no way to be sure that \(z_i\) was computed this way.

Q.3 Fully specify another \(\Sigma\)-protocol for the relation \(R((pp, G, X, Y, Z, U, V), (a, b))\) which is true if and only if \(U = aG + bY\) and \(V = aX + bZ\).
By defining a group action \((a, b) \ast ((G, X), (Y, Z)) = a(G, X) + b(Y, Z)\), we easily extend the previous protocol: the prover picks \((k, k') \in \mathbb{Z}_p^2\), computes and sends \((A, B) = (k, k') \ast ((G, X), (Y, Z))\). The verifier sends a challenge \(e \in \mathbb{Z}_p\). The prover computes and sends \((z, z') = (k, k') + e(a, b) \mod p\). The verifier checks \((z, z') \ast ((G, X), (Y, Z)) = (A, B) + e(U, V)\). The protocol looks as follows:

<table>
<thead>
<tr>
<th>Prover</th>
<th>Verifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>witness: (a, b)</td>
<td>instance: ((pp, G, X, Y, Z, U, V))</td>
</tr>
<tr>
<td>((U = aG + bY \text{ and } V = aX + bZ))</td>
<td></td>
</tr>
<tr>
<td>pick (k, k' \in \mathbb{Z}_p)</td>
<td>pick (e \in \mathbb{Z}_p)</td>
</tr>
<tr>
<td>(A = kG + k'Y, B = kX + k'Z)</td>
<td>verify:</td>
</tr>
<tr>
<td>(z = k + ea \mod p)</td>
<td>(zG + z'Y = A + eU)</td>
</tr>
<tr>
<td>(z' = k' + eb \mod p)</td>
<td>(zX + z'Z = B + eV)</td>
</tr>
</tbody>
</table>

Given \((A, B, e_1, z_1, z'_1)\) and \((A, B, e_2, z_2, z'_2)\), the extractor computes \(a = \frac{z_2 - z_1}{e_2 - e_1}\) and \(b = \frac{z'_2 - z'_1}{e_2 - e_1}\).

Given \(e\) and a random \((z, z')\), the simulator sets \((A, B) = (z, z') \ast ((G, X), (Y, Z)) - e(U, V)\).

Common mistake: a similar protocol with \(k' = k\) does not work as it leaks \(z' - z = b - a\). The simulator should fail.

Another common mistake is to send \(kG, k'Y, kX,\) and \(k'Z\) which is not zero-knowledge either. The simulator does not generate the right distribution.
2 Distinguisher for Lai-Massey Schemes

The Lai-Massey scheme is an alternate construction to the Feistel scheme to build a block cipher from round functions. Let \( n \) be the block size and \( r \) be the number of rounds. We denote by \( \oplus \) the bitwise XOR operation over bitstrings. Let the \( F_i \) be secret functions from \( \{0, 1\}^{\frac{n}{2}} \) to itself and \( \pi \) be a fixed public permutation over \( \{0, 1\}^{\frac{n}{2}} \). Let \( x, y \in \{0, 1\}^{\frac{n}{2}} \) and \( x\|y \) denote the concatenation of the two bitstrings. We define

\[
\varphi(F_1, \ldots, F_r)(x\|y) = \varphi(F_2, \ldots, F_r)(\pi(x \oplus F_1(x \oplus y))||(y \oplus F_1(x \oplus y)))
\]

for \( r > 1 \) and

\[
\varphi(F_r)(x\|y) = (x \oplus F_r(x \oplus y))||(y \oplus F_r(x \oplus y))
\]

when there is a single round. In what follows, we assume that the permutation \( \pi \) is defined by

\[
\pi(x_L\|x_R) = (x_R||(x_L \oplus x_R))
\]

where \( x_L, x_R \in \{0, 1\}^{\frac{n}{4}} \). For example, a 2-round Lai-Massey scheme is represented as follows:

Q.1 If \( \varphi(F_1, \ldots, F_r) \) is the encryption function, what is the decryption function?
We define \( \varphi' \) for \( r > 1 \) by

\[
\varphi'(F_r, \ldots, F_1)(x \| y) = (((\pi^{-1}(x') \oplus F_1(\pi^{-1}(x') \oplus y')))(y' \oplus F_1(\pi^{-1}(x') \oplus y')))
\]

where \( \varphi'(F_r, \ldots, F_2)(x \| y) = (x' \| y') \), and for \( r = 1 \) by \( \varphi'(F_1) = \varphi(F_1) \). We prove by induction that \( (\varphi(F_1, \ldots, F_r))^{-1} = \varphi(F_r, \ldots, F_1) \).

This is clear for \( r = 1 \). Actually, \( \varphi'(F_1) = \varphi(F_1) \) and we can directly see that \( \varphi(F_1, \ldots, F_r)) = x \| y \).

Assuming this is true for \( r - 1 \) rounds, we show that \( (\varphi'(F_r, \ldots, F_1) \circ \varphi(F_1, \ldots, F_r))(x \| y) = x \| y \) for any \( x \) and \( y \) as follows:

\[
(\varphi'(F_r, \ldots, F_1) \circ \varphi(F_1, \ldots, F_r))(x \| y) = (((\pi^{-1}(x') \oplus F_1(\pi^{-1}(x') \oplus y')))(y' \oplus F_1(\pi^{-1}(x') \oplus y')))
\]

where

\[
(x' \| y') = \varphi'(F_r, \ldots, F_2) (\varphi(F_2, \ldots, F_r)(\pi(x \oplus F_1(x \oplus y)))(y \oplus F_1(x \oplus y)))
\]

By the induction hypothesis, we have

\[
(x' \| y') = (\pi(x \oplus F_1(x \oplus y)))(y \oplus F_1(x \oplus y))
\]

By substituting \( x' \) and \( y' \) in the above equation, we obtain \( \varphi'(F_r, \ldots, F_1) \circ \varphi(F_1, \ldots, F_r))(x \| y) = x \| y \) which proves the property on \( r \) rounds.

Q.2 Give a distinguisher between \( \varphi(F_1) \) and a random permutation with a single known plaintext and advantage close to 1. (Compute the advantage.)

We have

\[
\varphi(F_1)(x \| y) = (x \oplus F_1(x \oplus y))(y \oplus F_1(x \oplus y))
\]

So, if \( x \| y \) is a known plaintext and \( x' \| y' = \varphi(F_1)(x \| y) \) is the corresponding ciphertext, we have

\[
x' \oplus y' = x \oplus y
\]

which is a property being satisfied with probability \( 2^{-\frac{n}{2}} \) for the random cipher. Hence, by checking this property, we have a distinguisher with advantage \( 1 - 2^{-\frac{n}{2}} \).

Q.3 Give a distinguisher between \( \varphi(F_1, F_2) \) and a random permutation with two chosen plaintexts and advantage close to 1. (Compute the advantage.)
We let $x_L, x_R, y_L, y_R, \alpha, \beta \in \{0, 1\}^n$. We assume that $x_L || x_R || y_L || y_R$ and $(x_L \oplus \alpha || (x_R \oplus \beta || (y_L \oplus \alpha || (y_R \oplus \beta)$ are the chosen plaintexts. Clearly, the input to $F_1$ is the same in both messages. We let $u || v$ denote the common output. The input and output to $\pi$ are

$$\pi((x_L \oplus u) || (x_R \oplus v)) = (x_R \oplus v) || (x_L \oplus x_R \oplus u \oplus v)$$

and

$$\pi((x_L \oplus \alpha \oplus u) || (x_R \oplus \beta \oplus v)) = (x_R \oplus \beta \oplus v) || (x_L \oplus \alpha \oplus x_R \oplus \beta \oplus u \oplus v)$$

If the two ciphertexts are $x'_L || x'_R || y'_L || y'_R$ and $x''_L || x''_R || y''_L || y''_R$ respectively, we have

$$x'_L \oplus y'_L = x_R \oplus v \oplus y_L \oplus u$$

$$x'_R \oplus y'_R = x_L \oplus x_R \oplus u \oplus y_R$$

$$x''_L \oplus y''_L = x_R \oplus v \oplus y_L \oplus u \oplus \alpha \oplus \beta$$

$$x''_R \oplus y''_R = x_L \oplus x_R \oplus u \oplus y_R \oplus \alpha \oplus \beta$$

and we can eliminate $u$ and $v$ and obtain

$$x'_R \oplus y'_R \oplus x''_R \oplus y''_R = \alpha \oplus \beta$$

$$x'_L \oplus x'_R \oplus y'_L \oplus y'_R = x''_L \oplus x''_R \oplus y''_L \oplus y''_R$$

These two properties are satisfied with probability close to $2^{-n}$ for the random cipher. Hence, by checking this property, we have a distinguisher with advantage close to $1 - 2^{-\frac{n}{2}}$. 
3 Bias in the Modulo $p$ Seed

We assume a setup phase $\text{Setup}(1^\lambda) \to p$ to determine a public prime number $p$ with security parameter $\lambda$. We consider the following generators:

**Generator $\text{Gen}_0(1^\lambda, p)$:**
1: pick $y \in U \mathbb{Z}_p$
2: return $y$

**Generator $\text{Gen}_1(1^\lambda, p)$:**
1: $\ell \leftarrow \lceil \log_2 p \rceil$
2: pick $x \in_U \{0, 1, \ldots, 2^\ell - 1\}$
3: $y \leftarrow x \mod p$
4: return $y$

**Generator $\text{Gen}_2(1^\lambda, p)$:**
1: $\ell \leftarrow \lceil \log_2 p \rceil$
2: pick $x \in_U \{0, 1, \ldots, 2^{\ell+\lambda} - 1\}$
3: $y \leftarrow x \mod p$
4: return $y$

Here, “pick $x \in_U E$” means that we sample $x$ from a set $E$ with uniform distribution. The value $\ell$ is the bitlength of $p$. In what follows, we consider distinguishers with unbounded complexity but limited to a single query to a generator.

**Q.1** Estimate how $\ell$ is usually fixed to have $\lambda$-bit security for typical cryptography in a (generic) group of order $p$. (For instance, in an elliptic curve.)

*Typically, we need the discrete logarithm to be hard. Due to generic attacks, this requires $\ell \geq 2\lambda$ to have $\lambda$-bit security. In a generic group, $\ell = 2\lambda$ is enough.*

**Q.2** Compute the advantage of the best distinguisher between $\text{Gen}_0$ and $\text{Gen}_1$. Could it be large?
We know that the best advantage of an unbounded distinguisher limited to one sample is equal to the statistical distance between the two distributions. We let $d_1$ be the statistical distance between the outputs of $\text{Gen}_0$ and $\text{Gen}_1$. We have

$$d_1 = \frac{1}{2} \sum_{y=0}^{p-1} \left| \frac{1}{p} - \Pr[x \mod p = y] \right|$$

where $x$ is uniform in $\{0, 1, \ldots, 2^\ell - 1\}$. Hence, $\Pr[x \mod p = y] = 2^{-\ell}$ if $y \geq 2^\ell \mod p$ and $\Pr[x \mod p = y] = 2 \times 2^{-\ell}$ otherwise. Thus,

$$d_1 = \frac{1}{2} \sum_{y=0}^{(2^\ell \mod p) - 1} \left| \frac{1}{p} - \frac{2}{2^\ell} \right| + \frac{1}{2} \sum_{y=2^\ell \mod p}^{p-1} \left| \frac{1}{p} - \frac{1}{2^\ell} \right|$$

$$= \sum_{y=0}^{(2^\ell \mod p) - 1} \left| \frac{1}{p} - \frac{2}{2^\ell} \right|$$

$$= \left(2^\ell \mod p\right) \left(\frac{2}{2^\ell} - \frac{1}{p}\right)$$

(The second line comes from that the difference between the two sums is equal to the sum of the two sums without absolute values which is zero.) We write $2^\ell = p + r$ with $0 \leq r < 2^\ell - 1 < p$. We have

$$d_1 = r \left(\frac{2}{2^\ell} - \frac{1}{2^\ell - r}\right)$$

As we can see, for $r \approx 2^\ell - 2$, we have $d_1 \approx \frac{1}{6}$. So $d_1$ can be pretty high. ($\frac{1}{6}$ is not negligible.)

Q.3 Compute the advantage of the best distinguisher between $\text{Gen}_0$ and $\text{Gen}_2$.

Hint: use the Euclidean division $2^{\ell+\lambda} = qp + r$.

We let $d_2$ be the statistical distance. We write $2^{\ell+\lambda} = qp + r$ with $0 \leq r < p$. For $y \geq r$ we have $\Pr[x \mod p = y] = \frac{q}{2^{\ell+\lambda}}$ and $\Pr[x \mod p = y] = \frac{q+1}{2^{\ell+\lambda}}$ otherwise. Hence, with the same computation,

$$d_2 = \sum_{y=0}^{r-1} \left(\frac{q+1}{2^{\ell+\lambda}} - \frac{1}{p}\right) = \frac{r}{2^{\ell+\lambda} - r} \left(\frac{q+1}{2^{\ell+\lambda}} - \frac{q}{2^{\ell+\lambda} - r}\right) \leq \frac{r}{2^{\ell+\lambda} - r} \frac{2^{\ell+\lambda} - r(q+1)}{2^{\ell+\lambda} - r} \leq 1$$

The upper bound increases with $r$ but we know that $r < p \leq 2^\ell$ so

$$d_2 \leq \frac{1}{2^{\lambda} - 1} \approx 2^{-\lambda}$$
Q.4 Based on the computations, what do you conclude about the generator algorithms?

To obtain a $\lambda$-bit security with generators in the group, we should certainly not use $\text{Gen}_1$. The $\text{Gen}_2$ generator is enough if we select a single element. If we rather need to use it $n$ times, we better pick $x$ of bitlength $\ell + \lambda + \lceil \log_2 n \rceil$. 