1 Digital Signature Scheme via Indistinguishable Obfuscation

A digital signature scheme can be constructed via indistinguishable obfuscation (iO). A digital signature scheme is made up of \((\text{Setup, Sign, Verify})\) as discussed in Lecture 13.

\[
(Pk, Sk) \leftarrow \text{Setup}(1^k) \\
\text{Sk} = \text{key of puncturable function and the seed of the PRF } F_k \\
Pk = iO(P_k) \text{ where } P_k \text{ is the program:} \\
P_k(m, \sigma): \\
\text{for some OWF function } f \\
\quad \text{return 1 if } f(\sigma) = f(F_k(m)) \\
\quad \text{return 0 otherwise}
\]

\[
\sigma = F_k(m) \leftarrow \text{Sign}(Sk, m) \\
\text{Remember that } Sk = k
\]

Output \(P_k(m, \sigma) \leftarrow \text{Verify}(Pk, m, \sigma)\)

Our security requirements will be that the adversary does wins the following game negligibly:

\begin{align*}
\text{Challenger} & \quad \text{Adversary} \\
(Pk, Sk) = \text{Setup}(1^k) \text{ and} \\
picks \text{random } m & \\
\sigma, m^\ast & \\
P_k, m & \\
\leftarrow & \\
\text{Adversary wins game if } \text{Verify}(Pk, m^\ast, \sigma) = 1
\end{align*}

To prove the security of this system, we use a hybrid argument. \(H_0\) is as above. \(H_1\): Adjust \(Pk\) so that \(Pk = iO(P_{k,m,\alpha})\) where \(\alpha = F_k(m)\) and \(P_{k,m,\alpha}\) is the program such that:

\[
P_{k,m,\alpha}(m^\ast, \sigma): \\
\text{for some OWF } f \\
\text{if } m = m^\ast: \\
\quad \text{if } f(\sigma) = f(\alpha) \text{ return 1} \\
\quad \text{otherwise return 0} \\
\text{else proceed as } P_k \text{ from before} \\
\quad \text{if } f(\sigma) = f(F_k(m^\ast)) \text{ return 1} \\
\quad \text{otherwise return 0}
\]
Note that this program does not change its output for any value.

\( H_2 \): Adjust \( \alpha \) so that it is a randomly sampled value.

\( H_3 \): Adjust the program such that instead of \( \alpha \) it relies on some \( \beta \) that is compared instead \( f(\alpha) \) in the third line.

Any adversary that can break \( H_3 \) non-negligibly can break the OWF \( f \) with at the value \( \beta \).

2 Public Key Encryption via Indistinguishable Obfuscation

A public key encryption scheme can be constructed via indistinguishable obfuscation. A public key encryption scheme is made up of \((Gen, Enc, Dec)\) as discussed in Lecture 9. The PRG used below is a length doubling PRG.

\[(Pk, Sk) \leftarrow Gen(1^k)\]

- \( Sk \) is key of puncturable function and the seed of the PRF \( F_k \)
- \( Pk = iO(P_k) \) where \( P_k \) is the program:
  \[
P_k(m, r): \]
  \[
t = PRG(r) \]
  \[
  \text{Output } c = (t, F_k(t) \oplus m) \]

Sample \( r \) and output \( (Pk(m, r)) \leftarrow Enc(Pk, m) \)

Output \( F_k(Sk, c_1) \oplus c_2 \leftarrow Dec(Sk = k, c = (c_1, c_2)) \)

Our security requirements will be that the adversary does wins the following game negligibly:

<table>
<thead>
<tr>
<th>Challenger</th>
<th>Adversary</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Pk, Sk) = Gen(1^k) and</td>
<td></td>
</tr>
<tr>
<td>Randomly sample ( b ) from {0,1} and</td>
<td></td>
</tr>
<tr>
<td>( c^* = Enc(Pk, b) ) and</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>( P_k, c^* )</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>( b^* )</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Adversary wins game if ( b = b^* )</td>
</tr>
</tbody>
</table>

To prove the security of this system, we use a hybrid argument. \( H_0 \) is as above.

\( H_1 \): Adjust \( Pk \) so that \( Pk = iO(P_{k, \alpha, t}) \) where \( \alpha = F_k(t) \) and \( P_{k, \alpha, t} \) is the program such that:

\[
P_{k, \alpha, t}(m, r): \]
\[
t^* = PRG(r) \]
\[
\text{if } t^* = t, \text{ output } (t^*, \alpha \oplus m) \]
\[
\text{else output } (t^*, F_k(t^*) \oplus m) \]

Note that this program does not change its output for any value.
\(H_2\): Adjust \(\alpha\) so that it is a randomly sampled value
\(H_3\): Adjust the program such that \(t^*\) is randomly sampled and is not in the range of the PRG

Any adversary that can win \(H_3\) can guess a random value non-negligibly.

3 Indistinguishable Obfuscation construction from \(NC^1\) iO

A construction of indistinguishable obfuscation from \(\text{iO}\) for circuits in \(NC^1\) is as follows:

Let \(P_{k,C}(x)\) be the circuit that outputs the garbled circuit \(UC(C, x)\) with randomness \(F_k(x)\) which is a punctured (at \(k\)) PRF in \(NC^1\)

Note that \(UC(C, x)\) outputs \(C(x)\) (\(UC\) is the “universal” circuit)
\(iO(C) \rightarrow \text{sample } k \text{ randomly from } \{0, 1\}^{\|x\|} \text{ and output } iO_{NC^1}(P_{k,C})\) padded to a length \(l\)

As before, we use a hybrid argument to show the security for \(iO\) (as from Lecture 20).

\(H_0\): \(iO(C) = iO_{NC^1}(P_{k,C})\) as above.
\(H_{\text{final}} = H_2^n: iO(Pk, c_2)\)

\(H_1 \cdots H_i\): Create a program \(Q_{k,c_1,c_2,i}(x)\) and obfuscate it.
\(Q_{k,c_1,c_2,i}(x)\):
- Sample \(k\) randomly
  - if \(x \geq i\), return \(P_{k,c_1}(x)\)
  - else, return \(P_{k,c_2}(x)\)

Note that \(H_i\) and \(H_{i+1}\) are indistinguishable for any value other than \(x = i\)
\(H_{i,1}\) between \(H_i\) and \(H_{i+1}\): Create a program \(Q_{k,c_1,c_2,i,\alpha}(x)\), where \(\alpha = Q_{k,c_1,c_2,i}(x)\) and obfuscate it.
\(Q_{k,c_1,c_2,i,\alpha}(x)\):
- Sample \(k\) randomly
  - if \(x = i\), return \(\alpha\)
  - else, return \(Q_{k,c_1,c_2,i}(x)\)

\(H_{i,2}\): Replace \(\alpha\) with a random \(\beta\) using fresh coins
\(H_{i,3}\): Create the \(c_2(x)\) value using fresh coins
\(H_{i,4}\): Create the \(c_2(x)\) value using \(F_k(x)\)
\(H_{i,5}\): Finish the migration to \(Q_{k,c_1,c_2,i+1}\)

Note that at \(H_{\text{final}}\), the circuit being obfuscated is completely changed from \(c_1\) to \(c_2\).