MTAT.07.003 Cryptology II
Spring 2012 / Exercise session ?? / Example Solution

**Exercise (Standard simulator for the Fiat-Shamir protocol).** *Let $\mathcal{S}(\beta)$ be the canonical simulator for the Fiat-Shamir sigma protocol, which takes in the challenge $\beta$ and outputs $\alpha, \gamma$ such that for uniformly chosen $\beta \xleftarrow{u} \{0,1\}$ the distribution of $(\alpha, \beta, \gamma)$ coincides with the message distribution between honest prover and verifier. Show that the following simulator construction*

$$
\begin{array}{l}
\mathcal{V}_\circ(\phi) \\
\left[\begin{array}{l}
\beta_\circ \xleftarrow{u} \{0,1\} \\
(\alpha, \gamma) \leftarrow \mathcal{S}(\beta_\circ) \\
\beta \leftarrow \mathcal{V}_*(\phi, \alpha) \\
\mathsf{if}\ \beta \neq \beta_\circ\ \mathsf{then}\ \boldsymbol{return}\ \perp \\
\mathsf{else}\ \boldsymbol{return}\ \mathcal{V}_*(\gamma)
\end{array}\right.
\end{array}
$$

*creates an output distribution $\psi_\circ$ that is statistically $\frac{1}{2}$-distant from the output distribution of malicious verifier $\mathcal{V}_*$ that interacts with the honest prover.*

**Solution.** There are many ways how to prove this seemingly evident fact. In this treatment we consider two most revealing approaches.

Proof based on event trees. The simplest way to visualise what happens in the simulation is to draw the event trees for the real protocol execution and for the simulation and compare them. For clarity, let us first consider adversaries $\mathcal{V}_*$ that do not abort in the real protocol run, i.e., do not output $\perp$. Also, let $k$ be the number of all possible commitment messages $\alpha$. In the Fiat-Shamir protocol, $k = \frac{\phi_e(N)}{4}$ where $\phi_e(\cdot)$ in this context is Euler's totient function. The latter follows from the fact that $\alpha$ is a quadratic residue.

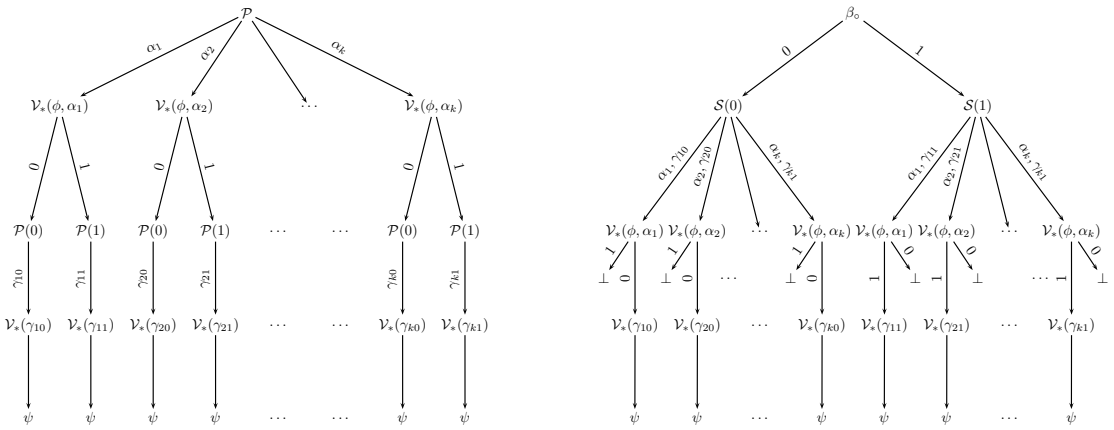


Figure 1: Event trees for real protocol execution and for the protocol simulation.

Figure 1 shows that there are many ways to obtain $\psi$ as an output. During the protocol execution, the prover $\mathcal{P}$ chooses uniformly a challenge $\alpha$ by computing $\alpha \leftarrow r^2$ for $r \xleftarrow{u} \mathbb{Z}_N^*$. Next, $\mathcal{V}_*$ chooses $\beta$ based on its previous knowledge $\phi$ and the commitment $\alpha$. After that the prover $\mathcal{P}$ computes the unique reply $\gamma \leftarrow rs^\beta$. The verifier $\mathcal{V}_*$ then processes the obtained information and produces an output. With some probability for each path in the event tree the outcome is $\psi$. During the simulation $\beta_\circ$ is chosen uniformly. After that the sub-simulator $\mathcal{S}$ creates matching $\alpha, \gamma$ values. Note that the number of possible commitments $\alpha$ that simulator outputs is still $k$. Moreover, the distribution of $\alpha$ values is still uniform for fixed $\beta_\circ$, since the simulator must sample protocol transcripts with the same probability that they occur in the protocol where

both parties are honest. After that the simulator feeds $\phi$ and $\alpha$ value to $\mathcal{V}_*$ to obtain $\beta$. If $\beta = \beta_\circ$, the maching response is fed into $\mathcal{V}_*$ to obtain the output. If $\beta \neq \beta_\circ$, the simulator halts abruptly.

At first glance, you may get a deceptive feeling that all paths are equally provable in the real protocol run. This is not true because $\mathcal{V}_*(\phi, \alpha)$ may bias the output $\beta$ depending on the value of $\alpha$. For the similar reason, the claim that in path during the simulation we reach $\perp$ is $\frac{1}{2}$ is incorrect. However, if we consider path pairs that go through $\alpha_i$ in the simulation, the potential bias cancels out. Thus, we can straightforwardly prove

$$\Pr\left[\begin{matrix}\beta_\circ \xleftarrow{u} \{0,1\}, \alpha, \gamma \leftarrow \mathcal{S}(\beta_\circ), \\ \beta \leftarrow \mathcal{V}_*(\phi, \alpha) : \beta \neq \beta_\circ \wedge \alpha = \alpha_0\end{matrix}\right] = \frac{1}{2} \cdot \frac{1}{k} \cdot \Pr\left[\mathcal{V}_*(\phi, \alpha_0) = 0\right] + \frac{1}{2} \cdot \frac{1}{k} \cdot \Pr\left[\mathcal{V}_*(\phi, \alpha_0) = 1\right] = \frac{1}{2k}$$

for any $\alpha_0$. Consequently,

$$\Pr\left[\begin{matrix}\beta_\circ \xleftarrow{u} \{0,1\}, \alpha, \gamma \leftarrow \mathcal{S}(\beta_\circ), \\ \beta \leftarrow \mathcal{V}_*(\phi, \alpha) : \beta \neq \beta_\circ\end{matrix}\right] = \sum_{\alpha_0} \Pr\left[\begin{matrix}\beta_\circ \xleftarrow{u} \{0,1\}, \alpha, \gamma \leftarrow \mathcal{S}(\beta_\circ), \\ \beta \leftarrow \mathcal{V}_*(\phi, \alpha) : \beta \neq \beta_\circ \wedge \alpha = \alpha_0\end{matrix}\right] = \frac{1}{2} \ .$$

As the second step note that there is a direct one-to-one correspondence of event paths in the real protocol execution and event paths without abortion in the simulation. Namely, each path is determined by $\alpha$ and $\beta$ value. Let us now establish how the probabilities of these paths are related:

$$\Pr\left[\begin{matrix}\beta_\circ \xleftarrow{u} \{0,1\}, \alpha, \gamma \leftarrow \mathcal{S}(\beta_\circ) : \beta_\circ = \beta_0 \\ \mathcal{V}_*(\phi, \alpha) = \beta_0 \wedge \alpha = \alpha_0 \wedge \gamma = \gamma(\alpha_0, \beta_0)\end{matrix}\right] = \frac{1}{2} \cdot \Pr\left[\begin{matrix}\alpha, \gamma \leftarrow \mathcal{S}(\beta_0) : \mathcal{V}_*(\phi, \alpha_0) = \beta_0 \\ \alpha = \alpha_0 \wedge \gamma = \gamma(\alpha_0, \beta_0)\end{matrix}\right]$$

where $\gamma(\alpha_0, \beta_0)$ denotes the correct response for $\alpha_0$ and $\beta_0$. As the simulation always yields a correct response $\gamma$ and the first message is uniformly distributed even for fixed $\beta_0$, we get

$$\Pr\left[\begin{matrix}\beta_\circ \xleftarrow{u} \{0,1\}, \alpha, \gamma \leftarrow \mathcal{S}(\beta_\circ) : \beta_\circ = \beta_0 \\ \mathcal{V}_*(\phi, \alpha) = \beta_0 \wedge \alpha = \alpha_0 \wedge \gamma = \gamma(\alpha_0, \beta_0)\end{matrix}\right] = \frac{1}{2k} \cdot \Pr\left[\mathcal{V}_*(\phi, \alpha_0) = \beta_0\right] \ .$$

The potability that we choose the corresponding path in the real protocol execution is

$$\Pr\left[\begin{matrix}\alpha \leftarrow \mathcal{P}, \beta \leftarrow \mathcal{V}_*(\phi, \alpha) : \\ \alpha = \alpha_0 \wedge \beta = \beta_0\end{matrix}\right] = \frac{1}{k} \cdot \Pr\left[\mathcal{V}_*(\phi, \alpha_0) = \beta_0\right] \ .$$

Thus, we have proven that the probability of simulated paths is exactly twice lower than in the real protocol.

These two facts make the computation of the statistical distance straightforward. Recall that the statistical distance is the maximal advantage of any distinguisher. Also, recall that the bound is achieved by Neyman-Pearson distinguiher, which for each input $x$ outputs the class for which the probability of getting $x$ is larger. In our case,

$$\mathcal{A}_{\mathrm{np}}(\psi_0) = \begin{cases} 0 & \text{if } \psi_0 \neq \perp \ , \\ 1 & \text{if } \psi_0 = \perp \ , \end{cases}$$

since probability of obtaining $\psi \neq \perp$ must be twice as low in the simulation as in the real protocol. As the corresponding advantage

$$\left|\Pr\left[\mathcal{A}_{\mathrm{np}}(\psi_0) = 1 | \mathcal{H}_0\right] - \Pr\left[\mathcal{A}_{\mathrm{np}}(\psi_0) = 1 | \mathcal{H}_1\right]\right| = \frac{1}{2} \ ,$$

we have proven the claim.

Proof based on game rewriting. Another alternative for proving the desired claim is a systematic rewriting of the simulator constructor until we reach the code from which the claim is more or less evident. First, recall that the zero-knowledge property of sigma protocols is defined through the equivalence of

following games:

$$\begin{array}{ll} \mathcal{Q}_0^{\mathcal{B}} & \mathcal{Q}_1^{\mathcal{B}} \\ \lceil \alpha \leftarrow \mathcal{P} & \lceil \beta \xleftarrow{u} \{0,1\} \\ \quad \beta \leftarrow \{0,1\} & \quad \alpha, \gamma \leftarrow \mathcal{S}(\beta) \\ \quad \gamma \leftarrow \mathcal{P}(\beta) & \quad \textbf{return } \mathcal{B}(\alpha, \beta, \gamma) \ . \\ \quad \textbf{return } \mathcal{B}(\alpha, \beta, \gamma) & \end{array}$$

That is, for any distinguishing algorithm $\mathcal{B}$ the distinguishing advantage is zero:

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{Q}_0, \mathcal{Q}_1}(\mathcal{B}) = \left| \Pr\left[\mathcal{Q}_0^{\mathcal{B}} = 1\right] - \Pr\left[\mathcal{Q}_1^{\mathcal{B}} = 1\right] \right| = 0 \ .$$

In other terms, the distributions of $(\alpha, \beta, \gamma)$ generated in both games coincide. As a consequence, we can rewrite the simulator constructor in the following way:

$$\begin{array}{ll} \mathcal{V}_\circ(\phi) & \mathcal{V}_\circ(\phi) \\ \lceil \beta_\circ \xleftarrow{u} \{0,1\} & \lceil \alpha \leftarrow \mathcal{P} \\ \quad (\alpha, \gamma) \leftarrow \mathcal{S}(\beta_\circ) & \quad \beta_\circ \leftarrow \{0,1\} \\ \quad \beta \leftarrow \mathcal{V}_*(\phi, \alpha) \quad \rightsquigarrow & \quad \gamma \leftarrow \mathcal{P}(\beta_\circ) \\ \quad \textsf{if } \beta \neq \beta_\circ \textsf{ then } \textbf{return } \bot & \quad \beta \leftarrow \mathcal{V}_*(\phi, \alpha) \\ \quad \textsf{else } \textbf{return } \mathcal{V}_*(\gamma) & \quad \textsf{if } \beta \neq \beta_\circ \textsf{ then } \textbf{return } \bot \\ & \quad \textsf{else } \textbf{return } \mathcal{V}_*(\gamma) \ . \end{array}$$

Since $\mathcal{V}_*$ response $\beta$ depends only on $\alpha$, we can move the statement upwards together with the if block:

$$\begin{array}{ll} \mathcal{V}_\circ(\phi) & \mathcal{V}_\circ(\phi) \\ \lceil \alpha \leftarrow \mathcal{P} & \lceil \alpha \leftarrow \mathcal{P} \\ \quad \beta_\circ \leftarrow \{0,1\} & \quad \beta \leftarrow \mathcal{V}_*(\phi, \alpha) \\ \quad \gamma \leftarrow \mathcal{P}(\beta_\circ) \quad \rightsquigarrow & \quad \beta_\circ \leftarrow \{0,1\} \\ \quad \beta \leftarrow \mathcal{V}_*(\phi, \alpha) & \quad \textsf{if } \beta \neq \beta_\circ \textsf{ then } \textbf{return } \bot \\ \quad \textsf{if } \beta \neq \beta_\circ \textsf{ then } \textbf{return } \bot & \quad \gamma \leftarrow \mathcal{P}(\beta) \\ \quad \textsf{else } \textbf{return } \mathcal{V}_*(\gamma) & \quad \textsf{else } \textbf{return } \mathcal{V}_*(\gamma) \ . \end{array}$$

From this it is clear that the simulator outputs $\bot$ with the probability $\frac{1}{2}$, since $\beta_\circ$ is generated independently after the $\beta$ is fixed. The reasoning is fundamentally the same we obtained by analysing the event trees. The code rewriting steps that do not alter the semantics of the program literally reshape the event trees without altering the output distribution and thus implicitly mimic the analytic reasoning without nasty details.

As the refactored simulator code outputs $\bot$ with probability $\frac{1}{2}$ we can simplify the code even further:

$$\begin{array}{lll} \mathcal{V}_\circ(\phi) & \mathcal{V}_\circ(\phi) & \mathcal{V}_\circ(\phi) \\ \lceil \alpha \leftarrow \mathcal{P} & \lceil \alpha \leftarrow \mathcal{P} & \lceil b \xleftarrow{u} \{0,1\} \\ \quad \beta \leftarrow \mathcal{V}_*(\phi, \alpha) & \quad \beta \leftarrow \mathcal{V}_*(\phi, \alpha) & \quad \textsf{if } b = 0 \textsf{ then } \textbf{return } \bot \\ \quad \beta_\circ \leftarrow \{0,1\} \quad \rightsquigarrow & \quad b \xleftarrow{u} \{0,1\} \quad \rightsquigarrow & \quad \alpha \leftarrow \mathcal{P} \\ \quad \textsf{if } \beta \neq \beta_\circ \textsf{ then } \textbf{return } \bot & \quad \textsf{if } b = 0 \textsf{ then } \textbf{return } \bot & \quad \beta \leftarrow \mathcal{V}_*(\phi, \alpha) \\ \quad \gamma \leftarrow \mathcal{P}(\beta) & \quad \gamma \leftarrow \mathcal{P}(\beta) & \quad \gamma \leftarrow \mathcal{P}(\beta) \\ \quad \textsf{else } \textbf{return } \mathcal{V}_*(\gamma) \ . & \quad \textsf{else } \textbf{return } \mathcal{V}_*(\gamma) \ . & \quad \textsf{else } \textbf{return } \mathcal{V}_*(\gamma) \ . \end{array}$$

From this representation it is also evident that

$$\Pr\left[\mathcal{V}_\circ(\phi) = \psi\right] = \frac{1}{2} \cdot \Pr\left[\alpha \leftarrow \mathcal{P}, \beta \leftarrow \mathcal{V}_*(\phi, \alpha), \gamma \leftarrow \mathcal{P}(\alpha) : \mathcal{V}_*(\gamma) = \phi\right] \ .$$

After that the reasoning about the statistical distance is identical to the approach used in the first proof.