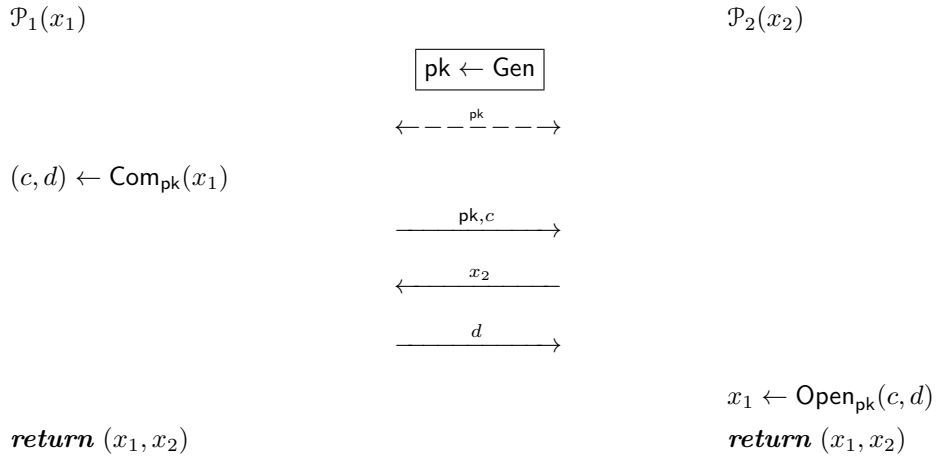
MTAT.07.003 Cryptology II
Spring 2012 / Exercise session ?? / Example Solution

Exercise (Security of simultaneous message exchange protocol). *Analyse security of the following simplistic protocol for simultaneous message exchange*

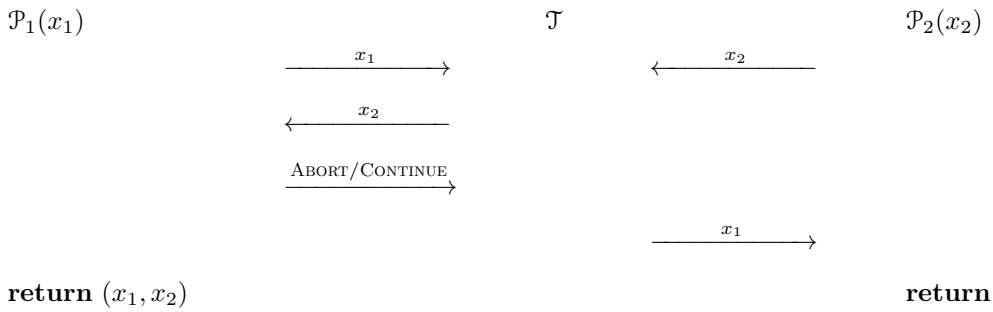


where bits x_1 and x_2 are private protocol inputs and a triple of algorithms $(\mathsf{Gen}, \mathsf{Com}, \mathsf{Open})$ is a commitment scheme $\mathcal{C}om$ with appropriate properties. The dashed line denotes sub-protocol for fixing the commitment parameters. Depending on the type of used commitment it can be specified as follows:

- *$\mathcal{P}_2$ will generate pk and send it over to $\mathcal{P}_1$ if $\mathcal{C}om$ is perfectly hiding for any acceptable value of pk,*
- *$\mathcal{P}_1$ will generate pk and send it over to $\mathcal{P}_2$ if $\mathcal{C}om$ is perfectly binding for any acceptable value of pk.*

It is also possible that the protocol has a trusted setup where parameter generation is done by a trusted third party. Consider security only against static malicious corruption.

Solution.
Right ideal implementation. As the first party $\mathcal{P}_1$ can refuse to open its input based on the opponents input x_2, we must consider the idealised functionality where the first party $\mathcal{P}_1$ is in the dominant position:



High-level descriptions for simulator constructions. Assume that the first party $\mathcal{P}_1$ is malicious. Then the corresponding simulator Sim must first provide an input $\hat{x}_1$ to the trusted third party $\mathcal{T}$ who replies x_2. After that it can still abort ideal computations by sending Abort signal. If the commitment parameters

are generated by $\mathcal{P}_1$, the corresponding simulator construction can be defined as follows

$\mathsf{Sim}(\phi_1, x_1)$
- $\omega_1 \xleftarrow{u} \Omega_1$
- $(\mathsf{pk}, c) \leftarrow \mathcal{P}_1^*(\phi_1, x_1; \omega_1)$
- $\hat{x}_1 \leftarrow \mathcal{K}(\phi_1, x_1, \omega_1)$
- Send $\hat{x}_1$ to $\mathcal{T}$ and store the reply as x_2.
- $d \leftarrow \mathcal{P}_1^*(x_2)$
- if $\mathsf{Open}_{\mathsf{pk}}(c, d) = \perp$ then
  - Send ABORT to $\mathcal{T}$
  - **return** $\mathcal{P}_2^*(\perp)$
- else if $\mathsf{Open}_{\mathsf{pk}}(c, d) = \hat{x}_1$ then
  - Send CONTINUE to $\mathcal{T}$
  - **return** $\mathcal{P}_2^*(x_2)$
- else **return** Fail

If commitment parameters are generated by $\mathcal{P}_2$ then the input extractor $\mathcal{K}$ must accept pk as an extra argument and the resulting simulator is somewhat different

$\mathsf{Sim}(\phi_1, x_1)$
- $\omega_1 \xleftarrow{u} \Omega_1$
- $\mathsf{pk} \leftarrow \mathsf{Gen}$
- $c \leftarrow \mathcal{P}_1^*(\phi_1, x_1, \mathsf{pk}; \omega_1)$
- $\hat{x}_1 \leftarrow \mathcal{K}(\phi_1, x_1, \mathsf{pk}, \omega_1)$
- Send $\hat{x}_1$ to $\mathcal{T}$ and store the reply as x_2.
- $d \leftarrow \mathcal{P}_1^*(x_2)$
- if $\mathsf{Open}_{\mathsf{pk}}(c, d) = \perp$ then
  - Send ABORT to $\mathcal{T}$
  - **return** $\mathcal{P}_2^*(\perp)$
- else if $\mathsf{Open}_{\mathsf{pk}}(c, d) = \hat{x}_1$ then
  - Send CONTINUE to $\mathcal{T}$
  - **return** $\mathcal{P}_2^*(x_2)$
- else **return** Fail

The interaction pattern is somewhat different if the second party $\mathcal{P}_2$ is malicious. Then the corresponding simulator Sim must first provide an input $\hat{x}_2$ to the trusted third party $\mathcal{T}$ who replies x_1. After that the simulator must make $\mathcal{P}_2^*$ to believe that honest party opened an input x_1 and the protocol outcome would be $(x_1, \hat{x}_2)$. As a result, the simulator consists of and input extraction followed by the output equivocation. If the commitment parameters are generated by $\mathcal{P}_1$, we get the the simulator

$\mathsf{Sim}(\phi_2, x_2)$
- $\mathsf{pk} \leftarrow \mathsf{Gen}$
- $\omega_2 \xleftarrow{u} \Omega_2$
- $\hat{x}_2 \leftarrow \mathcal{K}(\phi_2, x_2, \mathsf{pk}, \omega_2)$
- Send $\hat{x}_2$ to $\mathcal{T}$ and store the reply as x_1
- **return** $\mathcal{E}_\ell(\phi_2, x_1, x_2, \hat{x}_2, \mathsf{pk}, \omega_2)$

Note that the input extractor $\mathcal{K}$ and the output equivocator $\mathcal{E}_\ell$ must share inputs and randomness used by $\mathcal{P}_2^*$ or otherwise we cannot assure that the actions of $\mathcal{P}_2^*$ are consistent between both algorithms. The consistency is essential for getting a simulation with the right output distribution. If commitment parameters are generated by $\mathcal{P}_2$ then the plumbing between the simulator components changes

$$
\begin{array}{l}
\mathsf{Sim}(\phi_2, x_2) \\
\left[\begin{array}{l}
\omega_2 \xleftarrow{u} \Omega_2 \\
\mathsf{pk} \leftarrow \mathcal{P}_2^*(\phi_2, x_2; \omega_2) \\
\hat{x}_2 \leftarrow \mathcal{K}(\phi_2, x_2, \omega_2) \\
\text{Send } \hat{x}_2 \text{ to } \mathcal{T} \text{ and store the reply as } x_1 \\
\textbf{return } \mathcal{E}_\ell(\phi_2, x_1, x_2, \hat{x}_2, \mathsf{pk}, \omega_2)
\end{array}\right.
\end{array}
$$

but the overall scheme remains the same.

(A) Input extractor for $\mathcal{P}_1$. As the simulator must work universally well for all inputs (ϕ_1, x_1), the actual input $\hat{x}_1$ must be extracted form $\mathcal{P}_1$ in a black-box manner. Indeed, consider a specific adversary $\mathcal{P}_2^*$ that treats input ϕ_1 as the code and just interprets it to determine its actions. Depending on the precise computational model, such an interpretation is either linearly or quadratically slower than the dedicated attacker $\hat{\mathcal{P}}_2^*$ and we loose in efficiency. However, we gain universality – the input extractor $\mathcal{K}$ must works for all these attacks. Theoretically, the extractor $\mathcal{K}$ can use the code ϕ_1 to fine-tune its actions. However, so far nobody knows how to efficiently extract information from the program code and thus black-box execution with rewinding is the only known input extraction strategy.

The extractions itself depends who creates commitment parameters. If pk is generated inside $\mathcal{P}_1^*$ then initial inputs (ϕ_1, x_1, ω_1) together with x_2 completely fix the behaviour of $\mathcal{P}_1^*$. Consequently, we can consider the following extraction strategy

$$
\begin{array}{l}
\mathcal{K}(\phi_1, x_1, \omega_1) \\
\left[\begin{array}{l}
(\mathsf{pk}, c) \leftarrow \mathcal{P}_1^*(\phi_1, x_1, \omega_1) \\
\quad \text{For } x_2 \in \mathcal{X}_2 \text{ do} \\
\quad \left[\begin{array}{l}
d \leftarrow \mathcal{P}_1^*(x_2) \\
\hat{x}_1 \leftarrow \mathsf{Open}_{\mathsf{pk}}(c, d) \\
\mathsf{if}\ \hat{x}_1 \neq \perp\ \mathsf{then}\ \textbf{return}\ \hat{x}_1
\end{array}\right. \\
\textbf{return } \perp
\end{array}\right.
\end{array}
$$

where $\mathcal{X}_2$ is the set of all possible input values of the opponent $\mathcal{P}_2$. If pk is generated externally to $\mathcal{P}_1^*$ then we additionally need pk to completely fix the behaviour of $\mathcal{P}_1^*$. Thus, the plumbing between different components slightly changes

$$
\begin{array}{l}
\mathcal{K}(\phi_1, x_1, \mathsf{pk}, \omega_1) \\
\left[\begin{array}{l}
c \leftarrow \mathcal{P}_1^*(\phi_1, x_1, \mathsf{pk}, \omega_1) \\
\quad \text{For } x_2 \in \mathcal{X}_2 \text{ do} \\
\quad \left[\begin{array}{l}
d \leftarrow \mathcal{P}_1^*(x_2) \\
\hat{x}_1 \leftarrow \mathsf{Open}_{\mathsf{pk}}(c, d) \\
\mathsf{if}\ \hat{x}_1 \neq \perp\ \mathsf{then}\ \textbf{return}\ \hat{x}_1
\end{array}\right. \\
\textbf{return } \perp\ .
\end{array}\right.
\end{array}
$$

Prove the following facts

- If the commitment is perfectly binding then the protocol output y_2 of $\mathcal{P}_2$ is the same in the real and ideal world. Note that the output is completely determined by the values $(\phi_1, x_1, \omega_1, x_2)$ and thus can be considered as a deterministic function $y_2(\phi_1, x_1, \omega_1, x_2)$.

- Show that iteration over all possible values $\mathcal{X}_2$ is essential for black-box extraction. For that you may consider the following adversary that releases x_1 only for a specific input x_2:

$$
\begin{array}{l}
\mathcal{P}_2^*(\phi_1, x_1) \\
\left\lceil \begin{array}{l}
(\hat{x}_1, \hat{x}_2) \leftarrow \phi_1 \\
\mathsf{pk} \leftarrow \mathsf{Gen} \\
(c, d) \leftarrow \mathsf{Open}_{\mathsf{pk}}(\hat{x}_1) \\
\text{Release } \mathsf{pk} \text{ and } c. \text{ Store } x_2 \\
\mathsf{if}\ x_2 = \hat{x}_2\ \mathsf{then}\ \text{Release } d \\
\mathsf{else}\ \text{Abort}
\end{array} \right.
\end{array}
$$

  Show that if $\mathcal{K}$ does not use some $x_2 \in \mathcal{X}_2$ to extract input $\hat{x}_1$ then there exists inputs (ϕ_1, x_1) and (ϕ_2, x_2) such that the outcomes of $\mathcal{P}_2$ are completely different in the real and ideal world.

- What does the previous result mean in terms of size of the input domains $\mathcal{X}_2$.

- Show that the entire simulation construction is valid, i.e., the joint output distribution of $\mathcal{P}_1$ and $\mathcal{P}_2$ are identical if the commitment scheme is perfectly binding.

- Analyse what changes if we consider the setting with computationally binding commitments where pk is provided by $\mathcal{P}_2$.

(B) Input extractor for $\mathcal{P}_2$. First note that the input extractor for $\mathcal{P}_2^*$ must also work in a black-box manner. The reasoning is analogous to the reasoning given for malicious $\mathcal{P}_1^*$. As $\mathcal{P}_2$ releases its actual input $\hat{x}_2$ only after seeing $c_\circ$ we must provide some sort of commitment during extraction. However, differently form its opponent $\mathcal{P}_2^*$ reply may freely depend on $c_\circ$ and thus the semantics $\hat{x}_2$ is somewhat different – it is the input which can be later successfully combined with the revealed input x_1. This means that the quality of input extraction must be considered together with output equivocation. If commitment parameters pk are generated by the opponent $\mathcal{P}_1$ then the most natural input extraction strategy is following

$$
\begin{array}{l}
\mathcal{K}(\phi_2, x_2, \mathsf{pk}, \omega_2) \\
\left\lceil \begin{array}{l}
(c_\circ, d_\circ) \leftarrow \mathsf{Com}_{\mathsf{pk}}(0) \\
\hat{x}_2 \leftarrow \mathcal{P}_2^*(\phi_2, x_2, \mathsf{pk}; \omega_2) \\
\textbf{return } \hat{x}_2
\end{array} \right.
\end{array}
$$

where we set $\hat{x}_2$ to $\perp$ if $\mathcal{P}_2^*$ refuses to communicate after obtaining $c_\circ$. If commitment parameters pk are generated internally by $\mathcal{P}_2^*$ then the most natural input extraction strategy is following

$$
\begin{array}{l}
\mathcal{K}(\phi_2, x_2, \omega_2) \\
\left\lceil \begin{array}{l}
\mathsf{pk} \leftarrow \mathcal{P}_2^*(\phi_2, x_2; \omega_2) \\
(c_\circ, d_\circ) \leftarrow \mathsf{Com}_{\mathsf{pk}}(0) \\
\hat{x}_2 \leftarrow \mathcal{P}_2^*(x_2, \phi_2, \mathsf{pk}; \omega_2) \\
\textbf{return } \hat{x}_2
\end{array} \right.
\end{array}
$$

Prove the following facts

- If the commitment is perfectly hiding then the protocol output y_1 of $\mathcal{P}_1$ is the same in the real and ideal world. Note that the output is completely determined by the values $(\phi_2, x_2, \omega_2, x_1)$ and thus can be considered as a deterministic function $y_1(\phi_2, x_2, \omega_2, x_1)$.

- Analyse what changes if we consider the setting with computationally hiding commitments where pk is provided by $\mathcal{P}_1$. Show that corresponding distributions must be computationally indistinguishable. How is the corresponding time-bound connected to the running-time of the extractor.

- Show that if the number of possible input values $\mathcal{X}_2$ is small then the computational distance and statistical distance are equivalent, i.e., likelihood ratio test is efficient.

(C) Limited form of output equivocation for $\mathcal{P}_2$. Although the simulator using the input extractor $\mathcal{K}$ can perfectly match the output distribution of honest $\mathcal{P}_1$, we need to show closeness of the joint output distribution. This is straightforward for a limited class of malicious adversaries $\mathcal{P}_2^*$ that consist of two algorithms $\mathcal{A}$ and $\mathcal{B}$ with isolated states that are sequentially combined

$\mathcal{P}_1$		$\mathcal{P}_2^*(\phi_2, x_2)$
		$\omega_{\mathrm{a}} \leftarrow \Omega_2$
		$\mathsf{pk} \leftarrow \mathcal{A}(\phi_2, x_2; \omega_a)$
	$\xleftarrow{\mathsf{pk}}$	
...	$\xrightarrow{c}$	
		$\hat{x}_2 \leftarrow \mathcal{A}$
	$\xleftarrow{\hat{x_2}}$	
...	$\xrightarrow{d}$	
		$x_1 \leftarrow \mathsf{Open}_{\mathsf{pk}}(c, d)$
...		**return** $\mathcal{B}(\phi_2, x_2, x_1, \hat{x}_2, \omega_{\mathrm{a}})$

For such adversaries, the output equivocator is following

$\mathcal{E}(\phi_2, x_1, x_2, \hat{x}_2, \mathsf{pk}, \omega_2)$
- Split ω_2 into ω_{a} and ω_{b}
- **return** $\mathcal{B}(\phi_2, x_2, x_1, \hat{x}_2, \omega_{\mathrm{a}}; \omega_{\mathrm{b}})$

The randomness splitting is trivial if algorithms $\mathcal{A}$ and $\mathcal{B}$ have explicit description of the number f used random bits. If this is implicit, we can split the randomness by running $\mathcal{A}$ with ω_2 and set ω_{b} as the list of unused bits. Thus, the randomness splitting step is relatively efficient.

- Prove that if the commitment is perfectly hiding and the adversary has the structure described above then the joint output distributions in the real and ideal world are identical.
- Interpret the result. For which kind of security goals the malicious adversary $\mathcal{P}_2^*$ might obtain a significant gain is speed or in success. For that note that an isolated adversary $\mathcal{B}$ might completely restore the state of $\mathcal{A}$ is we additionally give him the randomness used to create the commitment decommitment pair (c, d). Are the class of neglected security goals relevant in the practice.

(D) Complete output equivocation for $\mathcal{P}_2$. To protect against all attack goals, we need equivocation algorithm works for malicious adversaries without structural restrictions. Let $\ell \in \mathbb{N}$ be a parameter determines a tradeoff between efficiency and quality of simulation. Then the following algorithm

$\mathcal{E}_\ell(\phi_2, x_1, x_2, \hat{x}_2, \mathsf{pk}, \omega_2)$
- For $i \in \{1, \ldots, \ell\}$ do
  - $(c, d) \leftarrow \mathsf{Com}_{\mathsf{pk}}(x_1)$
  - $x_2^* \leftarrow \mathcal{P}_2^*(x_2, \phi_2, \mathsf{pk}; \omega_2)$
  - if $x_2^* = \hat{x}_2$ then **return** $\mathcal{P}_2^*(d)$
- **return** Fail

performs rejection sampling over all possible protocol runs with the opponents input x_1 and $\mathcal{P}_2$ reply $\hat{x}_2$ and thus gets the desired output distribution when $\mathcal{E}$ does not fail.

Prove the following facts

- Assume that if the commitment is perfectly hiding and let $\hat{x}_2$ be the actual input fixed by the input extractor. Estimate the probability that $\mathcal{E}_\ell$ returns Fail as a function of probability $p(\hat{x}_2) = \Pr[\mathcal{K} = \hat{x}_2]$.
- Let $\mathcal{X}_2$ be the set of all potential inputs for $\mathcal{P}_2$ compute the maximal failure probability when $\hat{x}_2$ is sampled by $\mathcal{K}$.
- Prove that the statistical difference between joint output distributions is equal to the probability that $\Pr[\mathcal{E}_\ell = \mathsf{Fail}]$.
- Interpret results. How does the efficiency depend on desired statistical distance
- What changes if the commitment is only computationally hiding.

(E) Output equivocation based on trusted setup. We can use equivocal commitments to bypass problems in the output equivocation phase. But this leads to a setting with a trusted setup.

- Construct the corresponding simulator for malicious $\mathcal{P}_2$ by modifying the input extraction and output equivocation blocks
- Prove that the corresponding simulator achieves the desired goal. That is, the joint output distributions are identical in the real and ideal world.

(F) Input extraction based on trusted setup. The simulation efficiency for a malicious $\mathcal{P}_1^*$ depends on the size of $\mathcal{X}_2$ as the input extractor needs to iterate over all potential inputs x_2 to unlock the commitment. This problem can be bypassed if we use trusted setup with extractable commitment schemes.

- Construct the corresponding simulator for malicious $\mathcal{P}_1$ by modifying the input extraction block so that its efficiency does not depend on the size of $\mathcal{X}_2$.
- Prove that the corresponding simulator achieves the desired goal. That is, the joint output distributions are identical in the real and ideal world.