Sheherazade’s

*Tale of the Three Judges*

An example of stepwise development of security protocols

Annabelle McIver

Carroll Morgan
The three-judges protocol

Three judge ‘bots communicate over the internet to reach a verdict by majority: but no judge’s individual decision is to be revealed.

3JP

The three-judges protocol
The three-judges protocol

Three judge ‘bots communicate over the internet to reach a verdict by majority: but no judge’s individual decision is to be revealed.
The three-judges protocol

Three judge ‘bots communicate over the internet to reach a verdict by majority: but no judge’s individual decision is to be revealed.
The three-judges protocol

Three judge ‘bots communicate over the internet to reach a verdict by majority: but no judge’s individual decision is to be revealed.

The three-judges protocol
Two-party conjunction
Two-party disjunction
Oblivious transfer
Oblivious transfer
Oblivious transfer
Encryption lemma
Two-party conjunction

\[ \text{hid Bool } b, c \]
\[ \text{reveal } b \land c \]
Two-party conjunction

specification

hid Bool \( b, c \)

reveal \( b \land c \)

Agent \( B \) reveals a Boolean \( b_1 \) and Agent \( C \) reveals a Boolean \( c_0 \) such that the exclusive-or \( b_1 \oplus c_0 \) is the conjunction \( b \land c \).

But neither \( b \) nor \( c \) is itself revealed in the process.
Two-party conjunction

\texttt{hid} \texttt{Bool} \ b, \ c

\texttt{skip}

\texttt{reveal} \ b \land c
Two-party conjunction

\[ \text{hid} \ \text{Bool} \ b, c \]

\[ \| [ \ \text{hid} \ \text{Bool} \ b_1, b_2 \]

\[ b_1 \oplus b_2 := b \]

\[ \text{reveal} \ b_1 \]

\[ ] |  \| \]

\[ \text{reveal} \ b \land c \]

Encryption Lemma

A common component of many protocols of this sort, including the Dining Cryptographers (Chaum) and the Oblivious Transfer (Rabin/Rivest).
Two-party conjunction

\[
\begin{align*}
\| & \quad \text{hid} \ \text{Bool} \ b_1, b_2 \\
& \quad b_1 \oplus b_2 := b \\
& \quad \text{reveal} \ b_1 \\
\| & \quad \text{reveal} \ b \land c
\end{align*}
\]
Two-party conjunction

\[
\begin{align*}
\llbracket & \texttt{hid} \text{ Bool } b_1, b_2 \\
& b_1 \oplus b_2 := b \\
& \texttt{reveal} \ b_1 \\
& \texttt{reveal} \ b \land c \\
\rrbracket
\end{align*}
\]
Two-party conjunction

\[
|| \begin{align*}
\text{hid} & \quad \text{Bool} \quad b_1, b_2 \\
\quad & \\
\quad & b_1 \oplus b_2 := b \\
\quad & \text{reveal} \quad b_1 \\
\quad & \text{reveal} \quad b \triangleleft c \triangleright \text{false} \\
\end{align*}
\]

\begin{tabular}{c}
\text{IF} \\
\text{ELSE}
\end{tabular}
Two-party conjunction

\[
\begin{align*}
\| & \quad \text{hid Bool } b_1, b_2 \\
& \quad b_1 \oplus b_2 := b \\
& \quad \text{reveal } b_1 \\
& \quad \text{reveal } (b_1 \oplus b_2) \triangleleft c \triangleright (b_1 \oplus b_1)
\end{align*}
\]
Two-party conjunction

\[
\text{hid Bool } b_1, b_2 \\
b_1 \oplus b_2 := b \\
\text{reveal } b_1 \\
\text{reveal } b_1 \oplus (b_2 \lhd c \rhd b_1)
\]
Two-party conjunction

\[
\begin{align*}
\llbracket \text{hid Bool } b_1, b_2 \rrbracket \\
b_1 \oplus b_2 := b \\
\text{reveal } b_1 \\
\text{reveal } b_2 \triangleleft c \triangleright b_1
\end{align*}
\]
Two-party conjunction

\[
\begin{align*}
\parallel [ & \text{hid} \ \text{Bool} \ b_1, b_2 \\
& b_1 \oplus b_2 := b \\
& \text{reveal} \ b_1 \\
& \text{reveal} \ b_2 \triangleleft c \triangleright b_1 \\
\end{align*}
\]

replace by subprotocol
Two-party conjunction

\[
\begin{align*}
\| & \quad \text{hid} \text{ Bool } b_1, b_2 \\
& \quad b_1 \oplus b_2 := b \\
& \quad \text{reveal } b_1 \\
\end{align*}
\]

\[
c_0 := b_2 \triangleleft c \triangleright b_1 \quad \text{a subprotocol}
\]
Two-party conjunction

\[
\begin{align*}
\left[ \begin{array}{c}
\text{hid} \ Bool \ b_1, b_2 \\
\odot b_1 \oplus b_2 := b \\
\text{reveal} \ b_1 \\
\left[ \begin{array}{c}
\text{hid} \ Bool \ c_0 \\
\odot c_0 := b_2 \triangleleft c \triangleright b_1 \\
\text{reveal} \ c_0
\end{array} \right]
\end{array} \right]
\end{align*}
\]

a subprotocol
Two-party conjunction

\[ \parallel \begin{align*}
\text{hid} & \quad \text{Bool} & b_1, b_2 \\
\text{hid} & \quad \text{Bool} & c_0
\end{align*} \]

\[ b_1 \oplus b_2 := b \]

\[ \text{reveal} \ b_1 \]

\[ c_0 := b_2 \triangleleft c \triangleright b_1 \]

\[ \text{reveal} \ c_0 \]

\[ \parallel \]
Two-party conjunction

\[ \begin{align*}
\text{hid} & \text{ Bool } b_1, b_2 \\
\text{hid} & \text{ Bool } c_0 \\
\end{align*} \]

\[ b_1 \oplus b_2 := b \]

\[ \text{reveal } b_1 \]

\[ c_0 := b_2 \ll c \gg b_1 \]

\[ \text{reveal } c_0 \]

\[ B \text{ holds these} \]
Two-party conjunction

\[
\begin{align*}
\| & \text{hid Bool } b_1, b_2 & \quad B \text{ holds these} \\
& \text{hid Bool } c_0 & \quad C \text{ holds this} \\
& b_1 \oplus b_2 := b \\
& \text{reveal } b_1 \\
& c_0 := b_2 \triangleleft c \triangleright b_1 \\
& \text{reveal } c_0
\end{align*}
\]
Two-party conjunction

\[
\begin{align*}
&\text{hid Bool } b_1, b_2 \\
&\text{hid Bool } c_0 \\
&b_1 \oplus b_2 := b \\
&\text{reveal } b_1 \\
&c_0 := b_2 \triangleleft c \triangleright b_1 \\
&\text{reveal } c_0
\end{align*}
\]

\(B\) holds these

\(C\) holds this

\(B\) does these
Two-party conjunction

\[ \begin{align*}
\text{hid } \text{Bool } b_1, b_2 & \quad B \text{ holds these} \\
\text{hid } \text{Bool } c_0 & \quad C \text{ holds this} \\

b_1 \oplus b_2 & := b \\
\text{reveal } b_1 & \quad B \text{ does these} \\
c_0 & := b_2 \lhd c \rhd b_1 \\
\text{reveal } c_0 & \quad C \text{ does this} \\
\end{align*} \]
Two-party conjunction

\[\begin{align*}
\text{hid Bool } b_1, b_2 & \quad B \text{ holds these} \\
\text{hid Bool } c_0 & \quad C \text{ holds this} \\
\end{align*}\]

\[\begin{align*}
b_1 \oplus b_2 & : = b \\
\text{reveal } b_1 & \quad B \text{ does these} \\
c_0 & : = b_2 \triangleleft c \triangleright b_1 \\
\text{reveal } c_0 & \quad \text{Oblivious Transfer} \\
\end{align*}\]

\[\begin{align*}
\text{hid Bool } c_0 & \quad C \text{ does this} \\
\end{align*}\]

Two-party conjunction:

\[
\begin{align*}
\text{hid} & \text{ Bool } b_1, b_2 & B \text{ holds these} \\
\text{hid} & \text{ Bool } c_0 & C \text{ holds this} \\
\text{b}_1 \oplus b_2 & := b & \} & B \text{ does these} \\
\text{reveal} & \text{ b}_1 & \} & \text{Oblivious Transfer} \\
\text{c}_0 & := b_2 \triangleleft c \triangleright b_1 & C \text{ does this} \\
\text{reveal} & \text{ c}_0 & \\
\end{align*}
\]
Two-party conjunction: externally viewed

\[
\begin{align*}
\| & \quad \text{hid Bool } b_1, b_2 & \quad B \text{ holds these} \\
\text{hid Bool } c_0 & \quad C \text{ holds this} \\
\| & \quad b_1 \oplus b_2 := b & \quad B \text{ does these} \\
\text{reveal } b_1 & \quad \text{Oblivious Transfer} \\
\| & \quad c_0 := b_2 \triangleleft c \triangleright b_1 & \quad C \text{ does this} \\
\| & \quad \text{reveal } c_0
\end{align*}
\]
Two-party conjunction:
Two-party conjunction: as seen by $B$

Variables’ names usually indicate where they are located, i.e. which agent “owns” them. It’s an informal convention in this talk. It can of course be made precise with annotations, but we don’t bother.

On the other hand, we do bother for visibility attributes: an agent sometimes can see variables owned by others, and sometimes cannot.
Two-party conjunction: as seen by $B$

\[\text{vis Bool } b\]
\[\text{hid Bool } c\]

\[
\begin{align*}
\| & \text{vis Bool } b_1, b_2 \\
\text{hid Bool } c_0
\end{align*}
\]

\[
\begin{align*}
b_1 \oplus b_2 & := b; \quad \text{reveal } b_1 \\
c_0 & := b_2 \triangleleft c \triangleright b_1; \quad \text{reveal } c_0
\end{align*}
\]
Two-party conjunction:

\[ \text{hid Bool } b \]
\[ \text{vis Bool } c \]

\[ || \]
\[ \text{hid Bool } b_1, b_2 \]
\[ \text{vis Bool } c_0 \]

\[ b_1 \oplus b_2 := b; \quad \text{reveal } b_1 \]
\[ c_0 := b_2 \ll c \gg b_1; \quad \text{reveal } c_0 \]
\[ ]\]
Two-party conjunction: as seen by $C$

$$\textbf{hid} \ \text{Bool} \ b$$
$$\textbf{vis} \ \text{Bool} \ c$$

$$\| \quad \textbf{hid} \ \text{Bool} \ b_1, b_2$$
$$\textbf{vis} \ \text{Bool} \ c_0$$

$$b_1 \oplus b_2 := b; \quad \text{reveal} \ b_1$$
$$c_0 := b_2 \triangleleft c \triangleright b_1; \quad \text{reveal} \ c_0$$

$$\|$$
Two-party conjunction: multiple views

\[ \text{vis}_B \text{ Bool } b \]
\[ \text{vis}_C \text{ Bool } c \]

\[ \| \text{vis}_B \text{ Bool } b_1, b_2 \]
\[ \text{vis}_C \text{ Bool } c_0 \]

\[ b_1 \oplus b_2 := b; \quad \text{reveal } b_1 \]
\[ c_0 := b_2 \triangleleft c \triangleright b_1; \quad \text{reveal } c_0 \]
The tale of the Three Judges

*specification*

\[ \text{vis}_A \{0, 1\} \ a \]
\[ \text{vis}_B \{0, 1\} \ b \]
\[ \text{vis}_C \{0, 1\} \ c \]

reveal \((a + b + c \geq 2)\)

Reveal the majority verdict.

*secure refinement*

Do not reveal individual verdicts to anyone.
The tale of the Three Judges

reveal \((a + b + c \geq 2)\)
The tale of the Three Judges

reveal \((a + b + c \geq 2)\)

\[ a + b + c \geq 2 \equiv a \land (b \lor c) \lor b \land c \]
The tale of the Three Judges

reveal \((a + b + c \geq 2)\)

\[
a + b + c \geq 2 \\
\equiv a \land (b \lor c) \lor b \land c \\
\equiv b \lor c \triangleleft a \triangleright b \land c
\]
The tale of the Three Judges

\[ a + b + c \geq 2 \]

\[ \equiv a \land (b \lor c) \lor b \land c \]

\[ \equiv b \lor c \triangleleft a \triangleright b \land c \]

A shortcut.
The tale of the Three Judges: First attempt

\[
\text{if } a \\
\text{then reveal } (a + b + c \geq 2) \\
\text{else reveal } (a + b + c \geq 2) \\
\text{fi}
\]

But watch out... We will return to this.
The tale of the Three Judges: First attempt

\[
\text{if } a \\
\text{ then reveal } b \lor c \\
\text{ else reveal } b \land c \\
\text{fi}
\]
The tale of the Three Judges: First attempt

\[
\text{if } a \\text{ then } \text{reveal } b \lor c \quad \neg(\neg b \land \neg c) \\
\text{else } \text{reveal } b \land c \\
\text{fi}
\]
The tale of the Three Judges: First attempt

\[ \text{if } a \]
\[ \quad \text{then } \text{reveal } b \lor c \]
\[ \quad \text{else } \text{reveal } b \land c \]
\[ \text{fi} \]
The tale of the Three Judges: First attempt

\[
\text{if } a \\
\text{then reveal } b \lor c \\
\text{else}
\]

\text{fi}
The tale of the Three Judges: First attempt

if $a$

then reveal $b \lor c$

else $b_1 \oplus b_2 := b$

$c_0 := b_2 \triangleleft c \triangleright b_1$

fi
The tale of the Three Judges: First attempt

\[
\begin{align*}
&\text{if } a \\
&\quad \text{then reveal } b \lor c \\
&\quad \text{else } b_1 \oplus b_2 := b \\
&\quad \quad c_0 := b_2 \lhd c \rhd b_1 \\
&\quad \quad a_b := b_1 \\
&\quad \quad a_c := c_0
\end{align*}
\]

\text{fi}
The tale of the Three Judges: First attempt

\[
\text{if } a \\
\text{then reveal } b \lor c \\
\text{else } b_1 \oplus b_2 := b \\
\quad c_0 := b_2 \triangleleft c \triangleright b_1 \\
\quad a_b := b_1 \\
\quad a_c := c_0 \\
\text{reveal } a_b \oplus a_c \\
\text{fi}
\]

\[
\text{vis}_A \ a_b, \ a_c \\
\text{vis}_B \ b_1, \ b_2 \\
\text{vis}_C \ c_0
\]
The tale of the Three Judges: First attempt

\[
\text{if } a \\
\quad \text{then } b_1 \equiv b_2 := b \\
\quad c_0 := b_2 \triangleleft c \triangleright b_1 \\
\quad a_b := b_1 \\
\quad a_c := c_0 \\
\text{reveal } a_b \oplus a_c
\]
The tale of the Three Judges: First attempt

if $a$
    then $b_1 \equiv b_2 := b$
        $c_0 := b_2 \triangleleft c \triangleright b_1$
        $a_b := b_1$
        $a_c := c_0$
reveal $a_b \oplus a_c$

Oops! Agent $B$ learns $a$ by noting which protocol it is asked to follow.
The tale of the Three Judges: First attempt

\( prog \)

\( \neq \) \( \text{if } a \) \( \text{then } prog \) \( \text{else } prog \) \( \text{fi} \)

Because the two right-hand instances of \( prog \) can be manipulated independently, the Shadow semantics does not allow this equality.
The tale of the Three Judges: First attempt

reveal \((a + b + c \geq 2)\)

\[ a \neq \]

if \(a\)

then reveal \((a + b + c \geq 2)\)

else reveal \((a + b + c \geq 2)\)

fi

Not allowed.
The tale of the Three Judges:

\[ \text{reveal} \ (a + b + c \geq 2) \]

\[ ? \equiv \text{“Get both } b \lor c \text{ and } b \land c \” \]
\[ \text{reveal} \ (b \lor c) \triangleleft a \triangleright (b \land c) \]

This way, Agent B cannot tell which of the propositions Agent A actually wants.
The tale of the Three Judges: Second attempt

\[ \text{reveal } (a + b + c \geq 2) \]

\[ \Rightarrow \quad \text{“Get both } b \lor c \text{ and } b \land c” \]

\[ \text{reveal } (b \lor c) \vartriangleleft a \triangleright (b \land c) \]

This way, Agent B cannot tell which of the propositions Agent A actually wants.
The tale of the Three Judges: Second attempt

\[ \text{reveal } (a + b + c \geq 2) \]

\[ \neq \text{“Get both } b \lor c \text{ and } b \land c \text{”} \]

\[ \text{reveal } (b \lor c) \triangleleft a \triangleright (b \land c) \]

Oops! Agent A learns both \( b \lor c \) and \( b \land c \).
The tale of the Three Judges:

\[ \text{reveal} \ (a + b + c \geq 2) \]

\[
= \quad b \vee \bigoplus c \vee := b \lor c \\
\quad b \wedge \bigoplus c \wedge := b \land c \\
\quad a_b := (b \vee \triangleleft a \triangleright b \wedge) \\
\quad a_c := (c \vee \triangleleft a \triangleright c \wedge) \\
\text{reveal} \ a_b \bigoplus a_c
\]

\text{vis}_A \ a_b, \ a_c \\
\text{vis}_B \ b \wedge, \ b \vee \\
\text{vis}_C \ c \wedge, \ c \vee
The tale of the Three Judges: Correct refinement

reveal \((a + b + c \geq 2)\)

\[= \quad b_\lor \oplus c_\lor := b \lor c\]
\[b_\land \oplus c_\land := b \land c\]
\[a_b := (b_\lor \lhd a \rhd b_\land)\]
\[a_c := (c_\lor \lhd a \rhd c_\land)\]
\[reveal \quad a_b \oplus a_c\]

\(\text{vis}_A \ a_b, a_c\)
\(\text{vis}_B \ b_\land, b_\lor\)
\(\text{vis}_C \ c_\land, c_\lor\)

Variable’s name indicates its owner; its subscript indicates its purpose.
The tale of the Three Judges: Correct refinement

\[ \text{reveal } (a + b + c \geq 2) \]

\[
= b \lor \oplus c \lor := b \lor c \\
= b \land \oplus c \land := b \land c \\
a_b := (b \lor \triangleleft a \triangleright b \land) \\
a_c := (c \lor \triangleleft a \triangleright c \land) \\
\text{reveal } a_b \oplus a_c
\]

\[ \text{vis}_A a_b, a_c \]
\[ \text{vis}_B b \land, b \lor \]
\[ \text{vis}_C c \land, c \lor \]

None of Agents \(A, B, C\) learns anything about \(b \lor c\) from this.
The tale of the Three Judges: Correct refinement

\[ \text{reveal} \ (a + b + c \geq 2) \]

\[
\begin{align*}
= & \quad b \lor \bigoplus c \ := b \lor c \\
& \quad b \land \bigoplus c \ := b \land c \\
& \quad a_b \ := (b \lor \triangleleft a \triangleright b \land) \\
& \quad a_c \ := (c \lor \triangleleft a \triangleright c \land) \\
\text{reveal} \ a_b \bigoplus a_c
\end{align*}
\]

\[ \text{vis}_A \ a_b, a_c \]
\[ \text{vis}_B \ b \land, b \lor \]
\[ \text{vis}_C \ c \land, c \lor \]

None of Agents A,B,C learns anything about \( b \land c \) from this.
The tale of the Three Judges: Correct refinement

\[ \text{reveal} \ (a + b + c \geq 2) \]

\[ = \ b \lor \ominus c \ominus := b \lor c \]
\[ b \land \ominus c \ominus := b \land c \]
\[ a_b := (b \lor \lt a \gt b \land) \]
\[ a_c := (c \lor \lt a \gt c \land) \]
\[ \text{reveal} \ a_b \ominus a_c \]

\( \text{vis}_A \ a_b, a_c \)
\( \text{vis}_B \ b \land, b \lor \)
\( \text{vis}_C \ c \land, c \lor \)

Agent A learns nothing about “the other” \( b \); and Agent B learns nothing about \( a \).
The tale of the Three Judges: Correct refinement

\[ \text{reveal } (a + b + c \geq 2) \]

\[ = b \lor \oplus c \lor := b \lor c \]
\[ b \land \oplus c \land := b \land c \]
\[ a_b := (b \lor \lhd a \rhd b \land) \]
\[ a_c := (c \lor \lhd a \rhd c \land) \]
\[ \text{reveal } a_b \oplus a_c \]

\text{vis}_A a_b, a_c
\text{vis}_B b \land, b \lor
\text{vis}_C c \land, c \lor

Agent A learns nothing about “the other” \( c \); and Agent C learns nothing about \( a \).
The tale of the Three Judges: Correct refinement

\[
\text{reveal } (a + b + c \geq 2)
\]

\[
= b \lor \oplus c \lor := b \lor c
\]

\[
b \land \oplus c \land := b \land c
\]

\[
a_b := (b \lor \triangleleft a \triangleright b \land)
\]

\[
a_c := (c \lor \triangleleft a \triangleright c \land)
\]

\[
\text{reveal } a_b \oplus a_c
\]

In spite of all that, the verdict \((a+b+c \geq 2)\) is revealed at this point.
The tale of the Three Judges: Correct refinement

**reveal** \((a + b + c \geq 2)\)

\[
\begin{align*}
= & \quad b_\lor \oplus c_\lor := b \lor c \\
    & \quad b_\land \oplus c_\land := b \land c \\
    & \quad a_b := (b_\lor \lhd a \rhd b_\land) \\
    & \quad a_c := (c_\lor \lhd a \rhd c_\land) \\
\text{reveal} & \quad a_b \oplus a_c
\end{align*}
\]

\textbf{vis}_A \ a_b, \ a_c \\
\textbf{vis}_B \ b_\land, \ b_\lor \\
\textbf{vis}_C \ c_\land, \ c_\lor

In spite of all that, the verdict \((a+b+c \geq 2)\) is revealed at this point.
The tale of the Three Judges: Correct refinement

**reveal** \((a + b + c \geq 2)\)

\[
= \begin{align*}
    b_\lor \oplus c_\lor & := b \lor c \\
    b_\land \oplus c_\land & := b \land c \\
    a_b & := (b_\lor \triangleleft a \triangleright b_\land) \\
    a_c & := (c_\lor \triangleleft a \triangleright c_\land) \\
\end{align*}
\]

**vis**\(_A\) \(a_b, a_c\)
**vis**\(_B\) \(b_\land, b_\lor\)
**vis**\(_C\) \(c_\land, c_\lor\)

In spite of all that, the verdict \((a+b+c \geq 2)\) is revealed at this point.
The tale of the Three Judges: Correct refinement

\[ \text{reveal} \ (a + b + c \geq 2) \]

\[
= \begin{align*}
(b \lor \equiv b') & := b; \\
(c \lor) & := (b' \triangleleft c \triangleright b); \\
(b \land \oplus b') & := b; \\
(c \land) & := (b' \triangleleft c \triangleright b); \\
(a_b) & := (b \lor \triangleleft a \triangleright b); \\
(a_c) & := (c \lor \triangleleft a \triangleright c); \\
\text{reveal} & a_b \oplus a_c.
\end{align*}
\]
The tale of the Three Judges: Correct refinement

\[ \text{reveal } (a + b + c \geq 2) \]

\[ \begin{align*}
= & \quad (b_{\lor} \equiv b'_{\lor}) := b; \\
& \quad c_{\lor} := (b'_{\lor} \triangleleft c \triangleright b_{\lor}); \\
& \quad (b_{\land} \oplus b'_{\land}) := b; \\
& \quad c_{\land} := (b'_{\land} \triangleleft c \triangleright b_{\land}); \\
& \quad a_b := (b_{\lor} \triangleleft a \triangleright b_{\land}); \\
& \quad a_c := (c_{\lor} \triangleleft a \triangleright c_{\land}); \\
& \quad \text{reveal } a_b \oplus a_c
\end{align*} \]
The tale of the Three Judges: Correct refinement

\[ \text{reveal } (a + b + c \geq 2) \]

\[ = (b_\lor \equiv b'_\lor) := b; \]
\[ c_\lor := (b'_\lor < c > b_\lor); \]
\[ (b_\land \oplus b'_\land) := b; \]
\[ c_\land := (b'_\land < c > b_\land); \]
\[ a_b := (b_\lor < a > b_\land); \]
\[ a_c := (c_\lor < a > c_\land); \]
\[ \text{reveal } a_b \oplus a_c \]
The tale of the Three Judges: Correct refinement

reveal \((a + b + c \geq 2)\)

\[= (b \lor \equiv b') := b;\]
\[c \lor := \text{Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.}\]
\[= (b \land \oplus b') := b;\]
\[c \land := \text{Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.}\]
The tale of the Three Judges: Correct refinement

reveal \((a + b + c \geq 2)\)

== “Source-level proof”
The tale of the Three Judges: Correct refinement

\[ \text{reveal } (a + b + c \geq 2) \]

\[ = \text{ “} \text{This-} \text{-level proof”} \]

\[ b \lor \oplus c \lor := b \lor c \]
\[ b \land \oplus c \land := b \land c \]
\[ a_b := (b \lor \triangleleft a \triangleright b \land) \]
\[ a_c := (c \lor \triangleleft a \triangleright c \land) \]
\[ \text{reveal } a_b \oplus a_c \]

The subprotocols’ proofs are (will be one day) off-the-shelf, and need not be repeated for specific applications.
The tale of the Three Judges
The tale of the Three Judges

Judge A

Judge B

Judge C

Private communications.

initialisation
The tale of the Three Judges

initialisation

Judge A

Judge B

Judge C
The tale of the Three Judges

Judge A

Judge B

Judge C
The tale of the Three Judges

Judge A

Judge B

Judge C
The tale of the Three Judges

Judge A

Judge B

Judge C
The tale of the Three Judges

Judge A

Judge B

Judge C
The tale of the Three Judges

Judge A

Judge B

Judge C
The tale of the Three Judges

Judge A

Judge B

Judge C
The tale of the Three Judges

Judge A

Judge B

Judge C
The tale of the Three Judges

Judge A

Judge B

Judge C

ready to judge
The tale of the Three Judges

Judge A

Judge B

Judge C
The tale of the Three Judges

Judge A

assemble the verdict

Judge B

Judge C
The tale of the Three Judges

Judge A

Judge B

Judge C

assemble the verdict
The tale of the Three Judges

Judge A

Judge B

Judge C

assemble the verdict

Public communications.
The tale of the Three Judges

Judge A

Public communications.

assemble the verdict

Judge B

Judge C
The tale of the Three Judges

Judge A

Public communications.

assemble the verdict

Judge B

Judge C
The tale of the Three Judges

Judge A

assemble the verdict

Judge B

 Judge C
The tale of the Three Judges

Judge A

Judge B

Judge C

assemble the verdict
The tale of the Three Judges

Judge A

Judge B

Judge C
The tale of the Three Judges

Judge A

Judge B

Judge C
The tale of the Three Judges

Judge A: $a$

Judge B: $b \lor b \land b$

Judge C: $c \lor c \land c$

Judge: $a$
The tale of the Three Judges

Judge A

Judge B

Judge C

$\forall b$, $\forall c$

$\forall a$, $\forall b$

$\forall c$
The tale of the Three Judges

Judge A

Judge B

Judge C
The tale of the Three Judges

Judge A

\[ a, a_b, a_c \]

Judge B

\[ b \lor b, b, b \land c \]

Judge C

\[ c \lor c, c \land b \]
The tale of Judge A, Judge B, and Judge C.

Judge A now knows the majority verdict; before this point, no-one* knew more than their own judgement even though all green messages were public.

*except Sheherazade.
The tale of the Three Judges

Judge A

Judge B

Judge C

announce the outcome:

the defendant will rise...
The tale of the Three Judges

Judge A

$ab \oplus ac$

Judge B

$b \lor b \land b$

Judge C

$c \lor c \land c$

58
The tale of the Three Judges

Judge A

Judge B

Judge C

\[ a \]

\[ b \lor b \land b \]

\[ c \lor c \land c \]

\[ a b \oplus a c \]
The tale of the Three Judges

Judge A

Judge B

Judge C
The tale of the Three Judges

Judge A

(a + b + c) ≥ 2

Judge B

Judge C
The tale of the Three Judges

Judge A

Judge B

Judge C

\[(a+b+c) \geq 2\]
Appendix

- Derivation of the Encryption Lemma
- Derivation of The Three Judges
- Derivation of Oblivious Transfer
The Encryption Lemma derivation

\[
\text{var} \text{ Bool } b
\]

\[
\parallel \text{ hid Bool } b_1, b_2
\]

\[
b_1 \oplus b_2 := b
\]

\[
\text{reveal } b_1
\]

\[
\parallel
\]

\[
\parallel
\]

63
The Encryption Lemma derivation

\[
\text{var } \text{Bool } b \\
\| [ \text{hid} \text{ Bool } b_1, b_2 \\
\quad b_1 \oplus b_2 := b \\
\quad \text{reveal } b_1 ] ||
\]
The Encryption Lemma derivation

\textbf{var} \ Bool \ b

\begin{align*}
\| & \mid \text{hid} \ Bool \ b_1, b_2 \\
& b_2 \in \{0, 1\} \\
& b_1 := b \oplus b_2 \\
& \text{reveal} \ b_1 \\
\| & \end{align*}
The Encryption Lemma derivation

\[ \text{var } \text{Bool } b \]

\[ \begin{array}{l}
|| \ 
\text{hid } \text{Bool } b_1, b_2 \\
\quad b_1 : \in \{0, 1\} \\
\quad b_2 := b \oplus b_1 \\
\quad \text{reveal } b_1 \\
\end{array} \]

\[ \]
The Encryption Lemma derivation

\[ \begin{align*}
\textbf{var} & \ \text{Bool} \ b \\
\| & \textbf{hid} \ \text{Bool} \ b_1, b_2 \\
& b_1 : \in \ \{0, 1\} \\
& b_2 := b \oplus b_1 \\
\textbf{reveal} & \ b_1 \\
\| 
\end{align*} \]
The Encryption Lemma derivation

\textbf{var} \Bool b

\begin{align*}
\| & \begin{array}{l}
\text{hid } \Bool b_1 \\
b_1 \in \{0, 1\} \\
\text{reveal } b_1
\end{array} \\
\| & \end{align*}
The Encryption Lemma derivation

\texttt{var \ Bool \ b}

\texttt{skip}
Appendix

Derivation of the Encryption Lemma •
Derivation of The Three Judges •
Derivation of Oblivious Transfer •
The Three-Judges derivation

\[ \text{vis}_A \ a \]
\[ \text{vis}_B \ b \]
\[ \text{vis}_C \ c \]

reveal \( (a+b+c \geq 2) \)
The Three-Judges derivation

\[
\text{reveal } (a + b + c \geq 2)
\]
The Three-Judges derivation

\[\begin{array}{c}
\text{Lovers' Protocol} \\
\end{array}\]

\[\begin{array}{c}
\text{reveal} \ (a+b+c \geq 2) \\
\end{array}\]

A: trivial.
B,C: EL.
The Three-Judges derivation

$b^\wedge \oplus c^\wedge := b \land c$

reveal \((a + b + c \geq 2)\)
The Three-Judges derivation

\[ b \land \oplus c \land := b \land c \]
\[ b \lor \oplus c \lor := b \lor c \]

reveal \((a+b+c \geq 2)\)

**A:** trivial.

**B,C:** EL.
The Three-Judges derivation

\[ b\land \oplus c\land := b \land c \]
\[ b\lor \oplus c\lor := b \lor c \]

\[ a_b \oplus a_c := (a+b+c \geq 2) \]

\textbf{reveal} \( a_b \oplus a_c \)

\textit{A:} \ reveal.

\textit{B,C:} trivial.
The Three-Judges derivation

\[ b_\wedge \oplus c_\wedge := b \wedge c \]

\[ b_\lor \oplus c_\lor := b \lor c \]

\[ a_b \oplus a_c := (b_\lor \oplus c_\lor \triangleleft a \triangleright b_\wedge \oplus c_\wedge) \]

reveal \[ a_b \oplus a_c \]

\[ \text{prop calc} \]
\[ \text{prog alg} \]
The Three-Judges derivation

\[ b \wedge \oplus c \wedge := b \wedge c \]

\[ b \vee \oplus c \vee := b \vee c \]

\[ a_b \oplus a_c := (b \vee \triangleleft a \triangleright b \wedge) \oplus (c \vee \triangleleft a \triangleright c \wedge) \]

\textbf{reveal} \hspace{1em} a_b \oplus a_c

\textbf{prop calc}
The Three-Judges derivation

\[ b^\land \oplus c^\land := b \land c \]
\[ b^\lor \oplus c^\lor := b \lor c \]

\[ a_b, a_c := (b^\lor \triangleleft a \triangleright b^\land), (c^\lor \triangleleft a \triangleright c^\land) \]

reveal \( a_b \oplus a_c \)
The Three-Judges derivation

This resolution of nondeterminism is not valid on its own.

\[
\begin{align*}
 b_\wedge \oplus c_\wedge & := b \land c \\
 b_\lor \oplus c_\lor & := b \lor c 
\end{align*}
\]

\[ a_b, a_c := (b_\lor \triangleleft a \triangleright b_\wedge), (c_\lor \triangleleft a \triangleright c_\wedge) \]

**reveal** \( a_b \oplus a_c \)

\[
\begin{align*}
 \text{vis}_A & \ a \\
 \text{vis}_B & \ b \\
 \text{vis}_C & \ c \\
\| & \ \text{vis}_B \ b_\wedge; \text{vis}_C \ c_\wedge \\
 & \ \text{vis}_B \ b_\lor; \text{vis}_C \ c_\lor \\
 & \ \text{vis}_A \ a_b, a_c
\end{align*}
\]
The Three-Judges derivation

This resolution of nondeterminism is not valid on its own.

\[ b^\land \oplus c^\land := b \land c \]
\[ b^\lor \oplus c^\lor := b \lor c \]
\[ a_b \oplus a_c := (b^\lor \triangleleft a \triangleright b^\land) \oplus (c^\lor \triangleleft a \triangleright c^\land) \]
\[ a_b, a_c := (b^\lor \triangleleft a \triangleright b^\land), (c^\lor \triangleleft a \triangleright c^\land) \]
\[ \text{reveal} \ a_b \oplus a_c \]

\[ \begin{align*}
\text{vis}_A & \ a \\
\text{vis}_B & \ b \\
\text{vis}_C & \ c \\
[C & | [ \text{vis}_B \ b^\land; \text{vis}_C \ c^\land \\
\text{vis}_B & \ b^\lor; \text{vis}_C \ c^\lor \\
\text{vis}_A & \ a_b, a_c]
\end{align*} \]
The Three-Judges derivation

This resolution of nondeterminism is *not valid* on its own.

For $B,C$ it is invalid because it resolves hidden nondeterminism in the $A$-variables.

\[
\begin{align*}
    b_\land \oplus c_\land & := b \land c \\
    b_\lor \oplus c_\lor & := b \lor c \\
    a_b \oplus a_c & := (b_\lor \triangleleft a \triangleright b_\land) \oplus (c_\lor \triangleleft a \triangleright c_\land) \\
    a_b, a_c & := (b_\lor \triangleleft a \triangleright b_\land), (c_\lor \triangleleft a \triangleright c_\land) \\
    \text{reveal} a_b \oplus a_c
\end{align*}
\]
The Three-Judges derivation

This resolution of nondeterminism is *not valid* on its own.

\[
b\land \oplus c\land := b \land c
\]
\[
b\lor \oplus c\lor := b \lor c
\]

For \(B,C\) it is invalid because it resolves hidden nondeterminism in the \(A\)-variables.

For \(A\) it is invalid because it reveals information about the separate halves of the exclusive-or.

\[a_b, a_c := (b\lor \triangleleft a \triangleright b\land), (c\lor \triangleleft a \triangleright c\land)\]

reveal \(a_b \oplus a_c\)
The Three-Judges derivation

\[ B, C's \text{ point of view.} \]

\[
\begin{align*}
    b_\wedge \oplus c_\wedge & := b \wedge c \\
    b_\vee \oplus c_\vee & := b \vee c \\
    a_b \oplus a_c & := (b_\vee \triangleleft a \triangleright b_\wedge) \oplus (c_\vee \triangleleft a \triangleright c_\wedge)
\end{align*}
\]

reveal \[ a_b \oplus a_c \]
The Three-Judges derivation

\[ b_\wedge \oplus c_\wedge := b \wedge c \]
\[ b_\vee \oplus c_\vee := b \vee c \]

\[ a_b \oplus a_c := (b_\vee \triangleleft a \triangleright b_\wedge) \oplus (c_\vee \triangleleft a \triangleright c_\wedge) \]

\[ a_b \oplus a_c := a_b \oplus a_c \]

reveal \[ a_b \oplus a_c \]
The Three-Judges derivation

*B, C’s point of view.*

\[
b_{\land} \oplus c_{\land} := b \land c
\]

\[
b_{\lor} \oplus c_{\lor} := b \lor c
\]

\[
a_b, a_c := (b_{\lor} \triangleleft a \triangleright b_{\land}), (c_{\lor} \triangleleft a \triangleright c_{\land})
\]

\[
a_b \oplus a_c := a_b \oplus a_c
\]

**reveal** \(a_b \oplus a_c\)
The Three-Judges derivation

$B, C$'s point of view.

\[
\begin{align*}
    b_\wedge \oplus c_\wedge & := b \wedge c \\
    b_\vee \oplus c_\vee & := b \vee c
\end{align*}
\]

\[
\begin{align*}
    a_b, a_c & := (b_\vee \triangleleft a \triangleright b_\wedge), (c_\vee \triangleleft a \triangleright c_\wedge) \\
    \text{reveal} \ a_b \oplus a_c \\
    a_b \oplus a_c & := a_b \oplus a_c
\end{align*}
\]
The Three-Judges derivation

*B,C’s point of view.*

\[
\begin{align*}
b \land \oplus c \land & := b \land c \\
b \lor \oplus c \lor & := b \lor c
\end{align*}
\]

\[
\begin{align*}
a_b, a_c & := (b \lor \triangleleft a \triangleright b \land), (c \lor \triangleleft a \triangleright c \land) \\
\text{reveal} & \ a_b \oplus a_c
\end{align*}
\]

\[
\begin{align*}
\text{vis}_A & \ a \\
\text{vis}_B & \ b \\
\text{vis}_C & \ c \\
\mid \begin{align*}
\text{vis}_B & \ b \land; \text{vis}_C \ c \land \\
\text{vis}_B & \ b \lor; \text{vis}_C \ c \lor \\
\text{vis}_A & \ a_b, a_c
\end{align*}
\]

85
The Three-Judges derivation

A’s point of view, when $a$ is true.

\[
\begin{align*}
 b_{\wedge} \oplus c_{\wedge} & := b \wedge c \\
 b_{\vee} \oplus c_{\vee} & := b \vee c
\end{align*}
\]

\[
 a_b \oplus a_c := (b_{\vee} \triangleleft a \triangleright b_{\wedge}) \oplus (c_{\vee} \triangleleft a \triangleright c_{\wedge})
\]

reveal $a_b \oplus a_c$

$A$: reveal not “trivial”
The Three-Judges derivation

A’s point of view, when $a$ is true.

\[
\begin{align*}
b \land \oplus c \land & := b \land c \\
b \lor \oplus c \lor & := b \lor c \\
\end{align*}
\]

\[
\begin{align*}
a_b, a_c & := (b \lor \triangleleft a \triangleright b \land), (c \lor \triangleleft a \triangleright c \land) \\
\text{reveal } a_b & \oplus a_c
\end{align*}
\]

\[
\begin{align*}
\text{vis}_A & a \\
\text{vis}_B & b \\
\text{vis}_C & c \\
\mathrel{\parallel} \text{vis}_B b \land; \text{vis}_C c \land \\
\text{vis}_{AB} & b \lor; \text{vis}_{AC} c \lor \\
\text{vis}_A & a_b, a_c
\end{align*}
\]
The Three-Judges derivation

![A’s point of view, when a is true.](image)

\[ b_\land \oplus c_\land := b \land c \]
\[ b_\lor \oplus c_\lor := b \lor c \]
\[ a_b \oplus a_c := (b_\lor \lhd a \rhd b_\land) \oplus (c_\lor \lhd a \rhd c_\land) \]
\[ a_b, a_c := (b_\lor \lhd a \rhd b_\land), (c_\lor \lhd a \rhd c_\land) \]
\[ \text{reveal } a_b \oplus a_c \]

\[ \text{vis}_A a \]
\[ \text{vis}_B b \]
\[ \text{vis}_C c \]
\[ || \text{vis}_B b_\land; \text{vis}_C c_\land \]
\[ \text{vis}_{AB} b_\lor; \text{vis}_{AC} c_\lor \]
\[ \text{vis}_A a_b, a_c \]

A: standard refinement
The Three-Judges derivation

A’s point of view, when \( a \) is true.

\[
\begin{align*}
    b \land \oplus c \land & := b \land c \\
    b \lor \oplus c \lor & := b \lor c
\end{align*}
\]

\( a_b, a_c := (b \lor \triangleleft a \triangleright b \land), (c \lor \triangleleft a \triangleright c \land) \)

\( \text{reveal } a_b \oplus a_c \)

\( \forall \text{vis}_A a \), \( \text{vis}_B b \), \( \text{vis}_C c \)

\( \parallel \text{vis}_B b \land; \text{vis}_C c \land \text{vis}_B b \lor; \text{vis}_C c \lor \text{vis}_A a_b, a_c \)

A: reduce visibility
The Three-Judges derivation

A’s point of view, when \( a \) is true and, by symmetry, false too.

\[
\begin{align*}
\land & \oplus \land := b \land c \\
\lor & \oplus \lor := b \lor c \\
abla a, \nabla c & := (b \triangleleft a \triangleright b \land), (c \triangleleft a \triangleright c \land) \\
\text{reveal} \quad a_b \oplus a_c
\end{align*}
\]

\textbf{A:} symmetry
The Three-Judges derivation

An alternative is to consider the three statements together.

\[ b \land \oplus c \land := b \land c \]

\[ b \land \lor c \land := b \lor c \]

\[ a_b \lor a_c := (b \land \triangleleft a \triangleright b \land) \oplus (c \land \triangleleft a \triangleright c \land) \]

reveal \( a_b \lor a_c \)

\[ \text{vis}_A a \]
\[ \text{vis}_B b \]
\[ \text{vis}_C c \]

\[ \parallel \text{vis}_B b \land; \text{vis}_C c \land \]
\[ \text{vis}_B b \lor; \text{vis}_C c \lor \]
\[ \text{vis}_A a_b, a_c \]
The Three-Judges derivation

The case-analysis is avoided; but it becomes less algebraic.

\[
\begin{align*}
&b \land \oplus c \land := b \land c \\
&b \lor \oplus c \lor := b \lor c
\end{align*}
\]

\[
\begin{align*}
&\text{vis}_A \ a, \ a_b, \ a_c := (b \lor \lhd a \rhd b \land), (c \lor \lhd a \rhd c \land) \\
&\text{reveal} \ a_b \oplus a_c
\end{align*}
\]

macro-atomicity
The Three-Judges derivation

\[ b_\land \oplus c_\land := b \land c \]
\[ b_\lor \oplus c_\lor := b \lor c \]

\[ a_b, a_c := (b_\lor \triangleleft a \triangleright b_\land), (c_\lor \triangleleft a \triangleright c_\land) \]

**reveal** \[ a_b \oplus a_c \]

\[ \text{vis}_A \ a \]
\[ \text{vis}_B \ b \]
\[ \text{vis}_C \ c \]
\[ | [ \text{vis}_B \ b_\land; \text{vis}_C \ c_\land \]
\[ \text{vis}_B \ b_\lor; \text{vis}_C \ c_\lor \]
\[ \text{vis}_A \ a_b, a_c \]
The Three-Judges derivation

\[ b_\wedge \oplus c_\wedge := b \wedge c \]
\[ b_\lor \oplus c_\lor := b \lor c \]

\[ a_b := (b_\lor \triangleleft a \triangleright b_\wedge) \]
\[ a_c := (c_\lor \triangleleft a \triangleright c_\wedge) \]

\text{reveal} \ a_b \oplus a_c

\text{vis}_A \ a
\text{vis}_B \ b
\text{vis}_C \ c
\| \text{vis}_B \ b_\wedge ; \text{vis}_C \ c_\wedge
\text{vis}_B \ b_\lor ; \text{vis}_C \ c_\lor
\text{vis}_A \ a_b , a_c
Macro-atomicity

Let «$P$» mean “program fragment $P$ executed atomically: ephemeral visibles not seen, nor interior control flow.”
Macro-atomicity

Let «P» mean “program fragment P executed atomically: ephemeral visibles not seen, nor interior control flow.”

1. For classical primitive statement S we have «S» = S. This is by definition of primitive statements’ semantics.
Macro-atomicity

Let «P» mean “program fragment P executed atomically: ephemeral visibles not seen, nor interior control flow.”

1. For classical primitive statement S we have «S» = S. This is by definition of primitive statements’ semantics.

2. For fragments P,Q we have «P»; «Q» ⊑ «P;Q». This is by definition of «•». Informally, it is because the ephemerals are visible at left but hidden at right.
Macro-atomicity

Let «$P$» mean “program fragment $P$ executed atomically: ephemeral visibles not seen, nor interior control flow.”

1. For classical primitive statement $S$ we have «$S$» = $S$.
2. For fragments $P,Q$ we have «$P$»;«$Q$» ⊑ «$P;Q$».
3. For fragments $P,Q$ we have «$P$»;«$Q$» = «$P;Q$» provided the intermediate visibles can be inferred from the external visibles.
   
   Informally, it is because the intermediates’ being inferred means that hiding them has no effect.
Macro-atomicity

Let «$P$» mean “program fragment $P$ executed atomically: ephemeral visibles not seen, nor interior control flow.”

1. For classical primitive statement $S$ we have «$S$» = $S$.
2. For fragments $P,Q$ we have «$P$»;«$Q$» $\sqsubseteq$ «$P;Q$».
3. For fragments $P,Q$ we have «$P$»;«$Q$» = «$P;Q$» provided the intermediate visibles can be inferred from the external visibles.
4. For fragments $P,Q$ we have $P = Q$ implies «$P$» = «$Q$».
   Trivial.
Macro-atomicity

Let «P» mean “program fragment P executed atomically: ephemeral visibles not seen, nor interior control flow.”

1. For classical primitive statement S we have «S» = S.
2. For fragments P,Q we have «P»;«Q» ⊑ «P;Q».
3. For fragments P,Q we have «P»;«Q» = «P;Q» provided the intermediate visibles can be inferred from the external visibles.
4. For fragments P,Q we have P = Q implies «P» = «Q».

If two straight-line primitive sequences S₁;S₂;...;Sₘ and T₁;T₂;...;Tₙ are equal classically, and no visible is assigned-to twice at right, then the sequences are security-equal as well.
Appendix

- Derivation of the Encryption Lemma
- Derivation of The Three Judges
- Derivation of Oblivious Transfer
Oblivious transfer

\[ \text{vis}_A \ a, a_b \]
\[ \text{vis}_B \ b_v, b_\wedge \]
\[ a_b := (b_v \triangleleft a \triangleright b_\wedge) \]
Oblivious transfer

\[ a_b := (b_\lor \triangleleft a \triangleright b_\land) \]

\[ \text{vis}_A \ a, a_b \]

\[ \text{vis}_B \ b_\lor, b_\land \]
Oblivious transfer

\[ a_b := (b \triangleleft a \triangleright b \land) \]
Oblivious transfer

\[ a_b := (b \lor \triangleleft a \triangleright b \land) \]

**Encryption Lemma**

\[
\langle \text{vis } x; \text{vis}_A a' \\
\quad a' : \in \{0, 1\} \\
\quad x := a \oplus a' \\
\rangle \]

\[
\text{vis}_A a, a_b \\
\text{vis}_B b \lor, b \land
\]
Oblivious transfer

\[ a \leftarrow b \lor \neg a \land b \lor b \land (\mathsf{vis}_B b, b \land), b \land \mathsf{vis}_A a, a \land b \mid \mathsf{vis}_A a, a \land b \mid \mathsf{vis}_x x ; \mathsf{vis}_A a \leftarrow a \oplus a \oplus a' \]

\[ a_b := (b \lor a \land b \land) \]

\textbf{Encryption Lemma}
Oblivious transfer

\[ a' : \in \{0, 1\} \]
\[ x := a \oplus a' \]

\[ a_b := (b \lor \triangleleft a \triangleright b \land) \]

\[ \text{vis}_A a, a_b \]
\[ \text{vis}_B b \lor, b \land \]
\[ || \text{vis} x; \text{vis}_A a' \]
Oblivious transfer

\[ a': \in \{0, 1\} \]
\[ x := a \oplus a' \]
\[ \| [ \text{vis}_A a, a_b ] \]
\[ \text{vis}_B b'_0, b'_1 \]
\[ b'_0 : \in \{0, 1\}; b'_1 : \in \{0, 1\} \]
\[ y_\triangledown := b_\triangledown \oplus b'_x \]
\[ y_\wedge := b_\wedge \oplus b'_x \]
\[ a_b := (b_\triangledown \triangleleft a \triangleright b_\wedge) \]

Encryption Lemma twice
Oblivious transfer

\[ a' \in \{0, 1\} \]
\[ x := a \oplus a' \]
\[ \langle \text{vis } y_\lor, y_\land \rangle \]
\[ \text{vis}_B b'_0, b'_1 \]
\[ b'_0 \in \{0, 1\}; b'_1 \in \{0, 1\} \]
\[ y_\lor := b_\lor \oplus b'_x \]
\[ y_\land := b_\land \oplus b'_x \]
\[ a_b := (b_\lor \triangleleft a \triangleright b_\land) \]
Oblivious transfer

\[ a' : \in \{0, 1\} \]
\[ x := a \oplus a' \]

\[ b'_0 : \in \{0, 1\}; b'_1 : \in \{0, 1\} \]
\[ y_\lor := b_\lor \oplus b'_\neg x \]
\[ y_\land := b_\land \oplus b'_x \]

\[ a_b := (b_\lor \triangleleft a \triangleright b_\land) \]
Oblivious transfer

$$a' \in \{0, 1\}$$

$$x \leftarrow a \oplus a'$$

$$b'_0 \in \{0, 1\}; b'_1 \in \{0, 1\}$$

$$y_\land \leftarrow b_\land \oplus b'_x$$

$$y_\lor \leftarrow b_\lor \oplus b'_x$$

$$a_b := (b_\lor \triangleleft a \triangleright b_\land)$$
Oblivious transfer

\[
x := a \oplus a'
\]

\[
y \lor := b \lor \oplus b'_{\neg x}
\]

\[
y \land := b \land \oplus b'_x
\]

\[
a_b := (b \lor \triangleleft a \triangleright b \land)
\]

\[
\text{vis}_A a, a_b
\]
\[
\text{vis}_B b \lor, b \land
\]
\[
\| \text{vis} x; \text{vis}_A a'
\]
\[
\text{vis} y \lor, y \land
\]
\[
\text{vis}_B b'_0, b'_1
\]
\[
a' : \in \{0, 1\}
\]
\[
b'_0 : \in \{0, 1\}; b'_1 : \in \{0, 1\}
\]
Oblivious transfer

\[ x := a \oplus a' \]
\[ y_\lor := b_\lor \oplus b_x' \]
\[ y_\land := b_\land \oplus b_x' \]

\[ a_b := (b_\lor \triangleleft a \triangleright b_\land) \]

All three variables are visible to everyone; but they learn nothing at all from them.

(vis \_A \ a, a_b)
(vis \_B \ b_\lor, b_\land)
|| (vis \_ \ x; vis \_A \ a'
(vis \_ \ y_\lor, y_\land
(vis \_B \ b'_0, b'_1
a' \in \{0, 1\}
b'_0 \in \{0, 1\}; b'_1 \in \{0, 1\}
Oblivious transfer

\[ x := a \oplus a' \]
\[ y_\lor := b_\lor \oplus b'_x \]
\[ y_\land := b_\land \oplus b'_x \]
\[ a_b := (b_\lor \triangleleft a \triangleright b_\land) \]

All three variables are visible to everyone; but they learn nothing at all from them.

And yet...
Oblivious transfer

\[ x := a \oplus a' \]
\[ y_\lor := b_\lor \oplus b'_\neg x \]
\[ y_\land := b_\land \oplus b'_x \]

\[ a_b := (b_\lor \triangleleft a \triangleright b_\land) \]

\[ \text{vis}_A \ a, a_b \]
\[ \text{vis}_B \ b_\lor, b_\land \]
\[ \| \text{vis} \ x; \text{vis}_A \ a' \]
\[ \text{vis} \ y_\lor, y_\land \]
\[ \text{vis}_B \ b'_0, b'_1 \]
\[ a' : \in \{0, 1\} \]
\[ b'_0 : \in \{0, 1\}; b'_1 : \in \{0, 1\} \]
Oblivious transfer

\[ x := a \oplus a' \]
\[ y_\lor := b_\lor \oplus b'_{\neg x} \]
\[ y_\land := b_\land \oplus b'_x \]

\[ a_b := (y_\lor \oplus b'_{\neg x} \triangleleft a \triangleright y_\land \oplus b'_x) \]

**vis** \( A \) \( a, a_b \)
**vis** \( B \) \( b_\lor, b_\land \)
\( || \) \( \text{vis} \ x; \text{vis}_A \ a' \)
\( \text{vis} \ y_\lor, y_\land \)
\( \text{vis}_B \ b'_0, b'_1 \)
\( a' : \in \{0, 1\} \)
\( b'_0 : \in \{0, 1\}; b'_1 : \in \{0, 1\} \)
Oblivious transfer

\[ x := a \oplus a' \]
\[ y_\vee := b_\vee \oplus b'_\neg x \]
\[ y_\wedge := b_\wedge \oplus b'_x \]

\[ a_b := (y_\vee \ominus b'^{a'}_a \triangleleft a \triangleright y_\wedge \ominus b'^{a'}_a, ) \]
Oblivious transfer

\[ x := a \oplus a' \]
\[ y_\lor := b_\lor \oplus b'_x \]
\[ y_\land := b_\land \oplus b'_x \]

\[ a_b := (y_\lor \triangleleft a \triangleright y_\land) \oplus b'_{a'} \]
Oblivious transfer

\[ x \equiv a \oplus a' \]
\[ y_\lor \equiv b_\lor \oplus b'_x \]
\[ y_\land \equiv b_\land \oplus b'_x \]

\[ a_b \equiv (y_\lor \triangleleft a \triangleright y_\land) \oplus b'_{a'} \]

Both visible to \( A \)
Oblivious transfer

\[ x := a \oplus a' \]
\[ y_\vee := b_\vee \oplus b'_x \]
\[ y_\wedge := b_\wedge \oplus b'_x \]
\[ a'' := b'_{a'} \]
\[ a_b := (y_\vee \triangleleft a \triangleright y_\wedge) \oplus a'' \]
Oblivious transfer

\[ x := a \oplus a' \]

\[ y_\lor := b_\lor \oplus b'_x \]

\[ y_\land := b_\land \oplus b'_x \]

\[ a'' := (b'_1 \triangleleft a' \triangleright b'_0) \]

\[ a_b := (y_\lor \triangleleft a \triangleright y_\land) \oplus a'' \]
Oblivious transfer

\[ a'' := (b'_1 \triangleleft a' \triangleright b'_0) \]

\[ x := a \oplus a' \]

\[ y_\lor := b_\lor \oplus b'_{\neg x} \]

\[ y_\land := b_\land \oplus b'_x \]

\[ a_b := (y_\lor \triangleleft a \triangleright y_\land) \oplus a'' \]

\[ \textbf{vis}_A a, a_b \]

\[ \textbf{vis}_B b_\lor, b_\land \]

\[ \| [ \textbf{vis} x; \textbf{vis}_A a' \]

\[ \textbf{vis} y_\lor, y_\land \]

\[ \textbf{vis}_B b'_0, b'_1 \]

\[ a':\in \{0, 1\} \]

\[ b'_0:\in \{0, 1\} ; b'_1:\in \{0, 1\} \]

\[ \textbf{vis}_A a'' \]
Oblivious transfer

\( a' \in \{0, 1\} \)

\( b'_0 \in \{0, 1\}; b'_1 \in \{0, 1\} \)

\( a'' := (b'_1 \triangleleft a' \triangleright b'_0) \)

\( x := a \oplus a' \)

\( y_{\lor} := b_{\lor} \oplus b'_{\neg x} \)

\( y_{\land} := b_{\land} \oplus b'_x \)

\( a_b := (y_{\lor} \triangleleft a \triangleright y_{\land}) \oplus a'' \)

\( \text{vis}_A a, a_b \)

\( \text{vis}_B b_{\lor}, b_{\land} \)

\( \| [ \text{vis} x; \text{vis}_A a' \text{vis} y_{\lor}, y_{\land} \text{vis}_B b'_0, b'_1 \text{vis}_A a'' \)
Oblivious transfer

\[ a' \in \{0, 1\} \]
\[ b'_0 \in \{0, 1\}; \; b'_1 \in \{0, 1\} \]
\[ a'' := (b'_1 \triangleleft a' \triangleright b'_0) \]

\[ x := a \oplus a' \]
\[ y \lor := b_\lor \oplus b'_x \]
\[ y \land := b_\land \oplus b'_x \]
\[ a_b := (y \lor \triangleleft a \triangleright y \land) \oplus a'' \]

Done in advance, by a trusted third party
At first, prepare...

\[ a' : \in \{0, 1\} \]
\[ b'_0 : \in \{0, 1\}; b'_1 : \in \{0, 1\} \]
\[ a'' := (b'_1 \triangleleft a' \triangleright b'_0) \]

\[ x := a \oplus a' \]
\[ y_\lor := b_\lor \oplus b'_\neg x \]
\[ y_\land := b_\land \oplus b'_x \]
\[ a_b := (y_\lor \triangleleft a \triangleright y_\land) \oplus a'' \]
The actual transfer

\[ a' \in \{0, 1\} \]
\[ b_0' \in \{0, 1\}; b_1' \in \{0, 1\} \]
\[ a'' := (b_1' \triangleleft a' \triangleright b_0') \]

Later...

\[ x := a \oplus a' \]
\[ y_\lor := b_\lor \oplus b'_{x} \]
\[ y_\land := b_\land \oplus b'_{x} \]
\[ a_{b} := (y_\lor \triangleleft a \triangleright y_\land) \oplus a'' \]

\[ \text{vis}_A \ a, a_b \]
\[ \text{vis}_B \ b_\lor, b_\land \]
\[ \models \ [ \text{vis}_A \ a', a'' \]
\[ \text{vis}_B \ b_0', b_1' \]
\[ \text{vis} \ x, y_\lor, y_\land \]
Appendix

- Derivation of the Encryption Lemma
- Derivation of The Three Judges
- Derivation of Oblivious Transfer