
MODULE *ShamirSecretSharing*

Specification for simple *Shamir* Secret Sharing. This is not a variable secret sharing scheme.

We specify that dealer first sends shares to all players, and once all players have received their shares they can eventually reconstruct the secret.

We do not deal with the communication protocol between players to send their shares to each other before reconstructing the secret.

EXTENDS *Integers, Sequences*

CONSTANT

<i>Dealer</i> ,	The dealer sharing the secret with the players
<i>Players</i> ,	Set of all players
<i>Coefficients</i>	The coefficient of the polynomial. These are provided by the model

VARIABLES

<i>shares</i> ,	Function mapping Player to computed shares
<i>shares_sent</i> ,	Function mapping Player to shares received
<i>shares_received</i> ,	Function mapping Player to received shares
<i>reconstructed</i>	Function mapping Player to flag if secret has been successfully constructed

vars \triangleq $\langle \textit{shares}, \textit{shares_sent}, \textit{shares_received}, \textit{reconstructed} \rangle$

NoValue \triangleq -1

Init \triangleq

Compute shares as $a + bx + cx^2$
 $\wedge \textit{shares} = [p \in \textit{Players} \mapsto \textit{Coefficients}[1] + \textit{Coefficients}[2] * p + \textit{Coefficients}[3] * p^2]$
 $\wedge \textit{shares_sent} = [p \in \textit{Players} \mapsto \textit{NoValue}]$
 $\wedge \textit{shares_received} = [p \in \textit{Players} \mapsto \textit{NoValue}]$
 $\wedge \textit{reconstructed} = [p \in \textit{Players} \mapsto \text{FALSE}]$

The type invariant for all variables.

TypeOK \triangleq

$\wedge \textit{shares} \in [\textit{Players} \rightarrow \textit{Int}]$
 $\wedge \textit{shares_sent} \in [\textit{Players} \rightarrow \textit{Int}]$
 $\wedge \textit{shares_received} \in [\textit{Players} \rightarrow \textit{Int}]$
 $\wedge \textit{reconstructed} \in [\textit{Players} \rightarrow \text{BOOLEAN}]$

Send the share to Player *p*.

SendShare(*p*) \triangleq

$\wedge \textit{shares_sent}[p] = \textit{NoValue}$
 Send a share that has not been sent to anyone
 $\wedge \textit{shares_sent}' = [\textit{shares_sent} \text{ EXCEPT } ![p] = \textit{shares}[p]]$
 $\wedge \text{UNCHANGED } \langle \textit{shares}, \textit{shares_received}, \textit{reconstructed} \rangle$

Receive the share at Player p . It should have been sent before.

$$\begin{aligned} \text{ReceiveShare}(p) &\triangleq \\ &\wedge \text{shares_received}[p] = \text{NoValue} \\ &\wedge \text{shares_sent}[p] \neq \text{NoValue} \\ &\wedge \text{shares_received}' = [\text{shares_received} \text{ EXCEPT } ![p] = \text{shares_sent}[p]] \\ &\wedge \text{UNCHANGED } \langle \text{shares}, \text{shares_sent}, \text{reconstructed} \rangle \end{aligned}$$

Reconstruct secret with *Players* p and q . The payers should have received share.

$$\begin{aligned} \text{Reconstruct}(p, q) &\triangleq \\ &\wedge p \neq q \\ &\wedge \text{shares_received}[p] \neq \text{NoValue} \\ &\wedge \text{shares_received}[q] \neq \text{NoValue} \\ &\text{We don't specify how the secret is reconstructed, just that it is} \\ &\text{reconstructed using shares of all two player combinations} \\ &\wedge \text{reconstructed}' = [\text{reconstructed} \text{ EXCEPT } ![p] = \text{TRUE}] \\ &\wedge \text{UNCHANGED } \langle \text{shares}, \text{shares_sent}, \text{shares_received} \rangle \end{aligned}$$

The next step either sends shares, receives them or reconstructs the secret.

$$\begin{aligned} \text{Next} &\triangleq \\ &\vee \exists p \in \text{Players} : \\ &\quad \text{SendShare}(p) \vee \text{ReceiveShare}(p) \\ &\vee \exists p \in \text{Players}, q \in \text{Players} : \text{Reconstruct}(p, q) \\ \text{Spec} &\triangleq \\ &\wedge \text{Init} \\ &\wedge \Box[\text{Next}]_{\text{vars}} \end{aligned}$$

Liveness states that eventually all players reconstruct the secret.

$$\text{Liveness} \triangleq \forall p \in \text{Players}, q \in \text{Players} : \text{WF}_{\text{vars}}(\text{Reconstruct}(p, q))$$

For a fair specification, we assure the spec takes next steps and liveness is guaranteed.

$$\text{FairSpec} \triangleq \text{Spec} \wedge \text{Liveness}$$

\ * Modification History
\ * Last modified Tue Jun 13 21:51:11 CEST 2023 by kulpreet
\ * Created Fri Jun 09 17:03:07 CEST 2023 by kulpreet