

# Contents

<b>1</b>	<b>OMNIType Theory</b>	<b>3</b>
1.1	Datatypes . . . . .	3
1.2	Theorems . . . . .	3
<b>2</b>	<b>ssm11 Theory</b>	<b>4</b>
2.1	Datatypes . . . . .	4
2.2	Definitions . . . . .	4
2.3	Theorems . . . . .	5
<b>3</b>	<b>ssm Theory</b>	<b>11</b>
3.1	Datatypes . . . . .	11
3.2	Definitions . . . . .	12
3.3	Theorems . . . . .	13
<b>4</b>	<b>satList Theory</b>	<b>21</b>
4.1	Definitions . . . . .	21
4.2	Theorems . . . . .	21
<b>5</b>	<b>PBTypeIntegrated Theory</b>	<b>21</b>
5.1	Datatypes . . . . .	21
5.2	Theorems . . . . .	22
<b>6</b>	<b>PBIntegratedDef Theory</b>	<b>23</b>
6.1	Definitions . . . . .	23
6.2	Theorems . . . . .	24
<b>7</b>	<b>ssmConductORP Theory</b>	<b>28</b>
7.1	Definitions . . . . .	28
7.2	Theorems . . . . .	29
<b>8</b>	<b>ConductORPType Theory</b>	<b>33</b>
8.1	Datatypes . . . . .	33
8.2	Theorems . . . . .	34
<b>9</b>	<b>ssmConductPB Theory</b>	<b>35</b>
9.1	Definitions . . . . .	35
9.2	Theorems . . . . .	35
<b>10</b>	<b>ConductPBType Theory</b>	<b>40</b>
10.1	Datatypes . . . . .	40
10.2	Theorems . . . . .	40

<b>11</b>	<b>ssmMoveToORP Theory</b>	<b>41</b>
11.1	Definitions . . . . .	41
11.2	Theorems . . . . .	42
<b>12</b>	<b>MoveToORPType Theory</b>	<b>46</b>
12.1	Datatypes . . . . .	46
12.2	Theorems . . . . .	46
<b>13</b>	<b>ssmMoveToPB Theory</b>	<b>47</b>
13.1	Definitions . . . . .	47
13.2	Theorems . . . . .	47
<b>14</b>	<b>MoveToPBType Theory</b>	<b>51</b>
14.1	Datatypes . . . . .	51
14.2	Theorems . . . . .	51
<b>15</b>	<b>ssmPlanPB Theory</b>	<b>52</b>
15.1	Theorems . . . . .	52
<b>16</b>	<b>PlanPBType Theory</b>	<b>62</b>
16.1	Datatypes . . . . .	62
16.2	Theorems . . . . .	63
<b>17</b>	<b>PlanPBDef Theory</b>	<b>66</b>
17.1	Definitions . . . . .	66
17.2	Theorems . . . . .	67

# 1 OMNITYPE Theory

**Built:** 10 June 2018

**Parent Theories:** indexedLists, patternMatches

## 1.1 Datatypes

```
command = ESCc escCommand | SLc 'slCommand

escCommand = returnToBase | changeMission | resupply
             | reactToContact

escOutput = ReturnToBase | ChangeMission | Resupply
           | ReactToContact

escState = RTB | CM | RESUPPLY | RTC

output = ESCo escOutput | SLo 'slOutput

principal = SR 'stateRole

state = ESCs escState | SLs 'slState
```

## 1.2 Theorems

[command\_distinct\_clauses]

$\vdash \forall a' a. \text{ESCc } a \neq \text{SLc } a'$

[command\_one\_one]

$\vdash (\forall a a'. (\text{ESCc } a = \text{ESCc } a') \iff (a = a')) \wedge$   
 $\quad \forall a a'. (\text{SLc } a = \text{SLc } a') \iff (a = a')$

[escCommand\_distinct\_clauses]

$\vdash \text{returnToBase} \neq \text{changeMission} \wedge \text{returnToBase} \neq \text{resupply} \wedge$   
 $\quad \text{returnToBase} \neq \text{reactToContact} \wedge \text{changeMission} \neq \text{resupply} \wedge$   
 $\quad \text{changeMission} \neq \text{reactToContact} \wedge \text{resupply} \neq \text{reactToContact}$

[escOutput\_distinct\_clauses]

$\vdash \text{ReturnToBase} \neq \text{ChangeMission} \wedge \text{ReturnToBase} \neq \text{Resupply} \wedge$   
 $\quad \text{ReturnToBase} \neq \text{ReactToContact} \wedge \text{ChangeMission} \neq \text{Resupply} \wedge$   
 $\quad \text{ChangeMission} \neq \text{ReactToContact} \wedge \text{Resupply} \neq \text{ReactToContact}$

[escState\_distinct\_clauses]

$\vdash \text{RTB} \neq \text{CM} \wedge \text{RTB} \neq \text{RESUPPLY} \wedge \text{RTB} \neq \text{RTC} \wedge \text{CM} \neq \text{RESUPPLY} \wedge$   
 $\quad \text{CM} \neq \text{RTC} \wedge \text{RESUPPLY} \neq \text{RTC}$

[output\_distinct\_clauses]

$\vdash \forall a' a. \text{ESCo } a \neq \text{SLo } a'$

[output\_one\_one]

$\vdash (\forall a a'. (\text{ESCo } a = \text{ESCo } a') \iff (a = a')) \wedge$   
 $\quad \forall a a'. (\text{SLo } a = \text{SLo } a') \iff (a = a')$

[principal\_one\_one]

$\vdash \forall a a'. (\text{SR } a = \text{SR } a') \iff (a = a')$

[state\_distinct\_clauses]

$\vdash \forall a' a. \text{ESCs } a \neq \text{SLs } a'$

[state\_one\_one]

$\vdash (\forall a a'. (\text{ESCs } a = \text{ESCs } a') \iff (a = a')) \wedge$   
 $\quad \forall a a'. (\text{SLs } a = \text{SLs } a') \iff (a = a')$

## 2 ssm11 Theory

**Built:** 10 June 2018

**Parent Theories:** satList

### 2.1 Datatypes

```
configuration =
  CFG (('command order, 'principal, 'd, 'e) Form -> bool)
      (('state -> ('command order, 'principal, 'd, 'e) Form)
      (('command order, 'principal, 'd, 'e) Form list)
      (('command order, 'principal, 'd, 'e) Form list) 'state
      ('output list)

order = SOME 'command | NONE

trType = discard 'command | trap 'command | exec 'command
```

### 2.2 Definitions

[TR\_def]

$\vdash \text{TR} =$   
 $\quad (\lambda a_0 a_1 a_2 a_3.$   
 $\quad \quad \forall TR'.$   
 $\quad \quad (\forall a_0 a_1 a_2 a_3.$   
 $\quad \quad \quad (\exists \text{authenticationTest } P \text{ NS } M \text{ Oi } Os \text{ Out } s$   
 $\quad \quad \quad \quad \text{securityContext stateInterp cmd ins outs.}$   
 $\quad \quad \quad (a_0 = (M, Oi, Os)) \wedge (a_1 = \text{exec cmd}) \wedge$   
 $\quad \quad \quad (a_2 =$

```

CFG authenticationTest stateInterp
  securityContext (P says prop (SOME cmd)::ins) s
  outs) ∧
(a3 =
  CFG authenticationTest stateInterp
    securityContext ins (NS s (exec cmd))
    (Out s (exec cmd)::outs)) ∧
authenticationTest (P says prop (SOME cmd)) ∧
CFGInterpret (M, Oi, Os)
  (CFG authenticationTest stateInterp
    securityContext (P says prop (SOME cmd)::ins)
    s outs)) ∨
(∃ authenticationTest P NS M Oi Os Out s
  securityContext stateInterp cmd ins outs.
  (a0 = (M, Oi, Os)) ∧ (a1 = trap cmd) ∧
  (a2 =
    CFG authenticationTest stateInterp
      securityContext (P says prop (SOME cmd)::ins) s
      outs) ∧
  (a3 =
    CFG authenticationTest stateInterp
      securityContext ins (NS s (trap cmd))
      (Out s (trap cmd)::outs)) ∧
  authenticationTest (P says prop (SOME cmd)) ∧
  CFGInterpret (M, Oi, Os)
    (CFG authenticationTest stateInterp
      securityContext (P says prop (SOME cmd)::ins)
      s outs)) ∨
(∃ authenticationTest NS M Oi Os Out s securityContext
  stateInterp cmd x ins outs.
  (a0 = (M, Oi, Os)) ∧ (a1 = discard cmd) ∧
  (a2 =
    CFG authenticationTest stateInterp
      securityContext (x::ins) s outs) ∧
  (a3 =
    CFG authenticationTest stateInterp
      securityContext ins (NS s (discard cmd))
      (Out s (discard cmd)::outs)) ∧
  ¬authenticationTest x) ⇒
  TR' a0 a1 a2 a3) ⇒
  TR' a0 a1 a2 a3)

```

## 2.3 Theorems

[CFGInterpret\_def]

```

⊢ CFGInterpret (M, Oi, Os)
  (CFG authenticationTest stateInterp securityContext
    (input::ins) state outputStream) ⇔

```

$$(M, Oi, Os) \text{ satList } securityContext \wedge (M, Oi, Os) \text{ sat } input \wedge \\ (M, Oi, Os) \text{ sat } stateInterp \text{ state}$$

[CFGInterpret\_ind]

$$\vdash \forall P. \\ (\forall M \ Oi \ Os \ authenticationTest \ stateInterp \ securityContext \\ input \ ins \ state \ outputStream. \\ P \ (M, Oi, Os) \\ (CFG \ authenticationTest \ stateInterp \ securityContext \\ (input :: ins) \ state \ outputStream)) \wedge \\ (\forall v_{15} \ v_{10} \ v_{11} \ v_{12} \ v_{13} \ v_{14}. \\ P \ v_{15} \ (CFG \ v_{10} \ v_{11} \ v_{12} \ [] \ v_{13} \ v_{14})) \Rightarrow \\ \forall v \ v_1 \ v_2 \ v_3. \ P \ (v, v_1, v_2) \ v_3$$

[configuration\_one\_one]

$$\vdash \forall a_0 \ a_1 \ a_2 \ a_3 \ a_4 \ a_5 \ a'_0 \ a'_1 \ a'_2 \ a'_3 \ a'_4 \ a'_5. \\ (CFG \ a_0 \ a_1 \ a_2 \ a_3 \ a_4 \ a_5 = CFG \ a'_0 \ a'_1 \ a'_2 \ a'_3 \ a'_4 \ a'_5) \iff \\ (a_0 = a'_0) \wedge (a_1 = a'_1) \wedge (a_2 = a'_2) \wedge (a_3 = a'_3) \wedge \\ (a_4 = a'_4) \wedge (a_5 = a'_5)$$

[order\_distinct\_clauses]

$$\vdash \forall a. \text{ SOME } a \neq \text{ NONE}$$

[order\_one\_one]

$$\vdash \forall a \ a'. \ (\text{SOME } a = \text{SOME } a') \iff (a = a')$$

[TR\_cases]

$$\vdash \forall a_0 \ a_1 \ a_2 \ a_3. \\ \text{TR } a_0 \ a_1 \ a_2 \ a_3 \iff \\ (\exists authenticationTest \ P \ NS \ M \ Oi \ Os \ Out \ s \ securityContext \\ stateInterp \ cmd \ ins \ outs. \\ (a_0 = (M, Oi, Os)) \wedge (a_1 = \text{exec } cmd) \wedge \\ (a_2 = \\ CFG \ authenticationTest \ stateInterp \ securityContext \\ (P \text{ says prop } (\text{SOME } cmd) :: ins) \ s \ outs) \wedge \\ (a_3 = \\ CFG \ authenticationTest \ stateInterp \ securityContext \ ins \\ (NS \ s \ (\text{exec } cmd)) \ (Out \ s \ (\text{exec } cmd) :: outs)) \wedge \\ authenticationTest \ (P \text{ says prop } (\text{SOME } cmd)) \wedge \\ CFGInterpret \ (M, Oi, Os) \\ (CFG \ authenticationTest \ stateInterp \ securityContext \\ (P \text{ says prop } (\text{SOME } cmd) :: ins) \ s \ outs)) \vee \\ (\exists authenticationTest \ P \ NS \ M \ Oi \ Os \ Out \ s \ securityContext \\ stateInterp \ cmd \ ins \ outs. \\ (a_0 = (M, Oi, Os)) \wedge (a_1 = \text{trap } cmd) \wedge \\ (a_2 = \\ CFG \ authenticationTest \ stateInterp \ securityContext \\ (P \text{ says prop } (\text{SOME } cmd) :: ins) \ s \ outs) \wedge$$

$$\begin{aligned}
& (a_3 = \\
& \quad \text{CFG authenticationTest stateInterp securityContext ins} \\
& \quad \quad (\text{NS } s \text{ (trap cmd)}) (\text{Out } s \text{ (trap cmd)::outs})) \wedge \\
& \quad \text{authenticationTest } (P \text{ says prop (SOME cmd)}) \wedge \\
& \quad \text{CFGInterpret } (M, Oi, Os) \\
& \quad \quad (\text{CFG authenticationTest stateInterp securityContext} \\
& \quad \quad \quad (P \text{ says prop (SOME cmd)::ins) } s \text{ outs})) \vee \\
& \exists \text{ authenticationTest NS } M \text{ } Oi \text{ } Os \text{ } \text{Out } s \text{ securityContext} \\
& \quad \text{stateInterp cmd } x \text{ ins outs.} \\
& (a_0 = (M, Oi, Os)) \wedge (a_1 = \text{discard cmd}) \wedge \\
& (a_2 = \\
& \quad \text{CFG authenticationTest stateInterp securityContext} \\
& \quad \quad (x::ins) s \text{ outs}) \wedge \\
& (a_3 = \\
& \quad \text{CFG authenticationTest stateInterp securityContext ins} \\
& \quad \quad (\text{NS } s \text{ (discard cmd)}) (\text{Out } s \text{ (discard cmd)::outs})) \wedge \\
& \neg \text{authenticationTest } x
\end{aligned}$$

[TR\_discard\_cmd\_rule]

$$\begin{aligned}
& \vdash \text{TR } (M, Oi, Os) \text{ (discard cmd)} \\
& \quad (\text{CFG authenticationTest stateInterp securityContext} \\
& \quad \quad (x::ins) s \text{ outs}) \\
& \quad (\text{CFG authenticationTest stateInterp securityContext ins} \\
& \quad \quad \quad (\text{NS } s \text{ (discard cmd)}) (\text{Out } s \text{ (discard cmd)::outs})) \iff \\
& \neg \text{authenticationTest } x
\end{aligned}$$

[TR\_EQ\_rules\_thm]

$$\begin{aligned}
& \vdash (\text{TR } (M, Oi, Os) \text{ (exec cmd)}) \\
& \quad (\text{CFG authenticationTest stateInterp securityContext} \\
& \quad \quad (P \text{ says prop (SOME cmd)::ins) } s \text{ outs}) \\
& \quad (\text{CFG authenticationTest stateInterp securityContext ins} \\
& \quad \quad \quad (\text{NS } s \text{ (exec cmd)}) (\text{Out } s \text{ (exec cmd)::outs})) \iff \\
& \text{authenticationTest } (P \text{ says prop (SOME cmd)}) \wedge \\
& \text{CFGInterpret } (M, Oi, Os) \\
& \quad (\text{CFG authenticationTest stateInterp securityContext} \\
& \quad \quad (P \text{ says prop (SOME cmd)::ins) } s \text{ outs})) \wedge \\
& (\text{TR } (M, Oi, Os) \text{ (trap cmd)}) \\
& \quad (\text{CFG authenticationTest stateInterp securityContext} \\
& \quad \quad (P \text{ says prop (SOME cmd)::ins) } s \text{ outs}) \\
& \quad (\text{CFG authenticationTest stateInterp securityContext ins} \\
& \quad \quad \quad (\text{NS } s \text{ (trap cmd)}) (\text{Out } s \text{ (trap cmd)::outs})) \iff \\
& \text{authenticationTest } (P \text{ says prop (SOME cmd)}) \wedge \\
& \text{CFGInterpret } (M, Oi, Os) \\
& \quad (\text{CFG authenticationTest stateInterp securityContext} \\
& \quad \quad (P \text{ says prop (SOME cmd)::ins) } s \text{ outs})) \wedge \\
& (\text{TR } (M, Oi, Os) \text{ (discard cmd)}) \\
& \quad (\text{CFG authenticationTest stateInterp securityContext} \\
& \quad \quad (x::ins) s \text{ outs}) \\
& \quad (\text{CFG authenticationTest stateInterp securityContext ins}
\end{aligned}$$

$$(NS\ s\ (\text{discard}\ cmd))\ (Out\ s\ (\text{discard}\ cmd)::outs)) \iff \neg authenticationTest\ x)$$

[TR\_exec\_cmd\_rule]

$$\begin{aligned} &\vdash \forall authenticationTest\ securityContext\ stateInterp\ P\ cmd\ ins\ s\ outs. \\ &\quad (\forall M\ Oi\ Os. \\ &\quad \quad CFGInterpret\ (M, Oi, Os) \\ &\quad \quad \quad (CFG\ authenticationTest\ stateInterp\ securityContext \\ &\quad \quad \quad \quad (P\ \text{says}\ \text{prop}\ (SOME\ cmd)::ins)\ s\ outs) \Rightarrow \\ &\quad \quad \quad (M, Oi, Os)\ \text{sat}\ \text{prop}\ (SOME\ cmd)) \Rightarrow \\ &\quad \forall NS\ Out\ M\ Oi\ Os. \\ &\quad TR\ (M, Oi, Os)\ (\text{exec}\ cmd) \\ &\quad \quad (CFG\ authenticationTest\ stateInterp\ securityContext \\ &\quad \quad \quad (P\ \text{says}\ \text{prop}\ (SOME\ cmd)::ins)\ s\ outs) \\ &\quad \quad (CFG\ authenticationTest\ stateInterp\ securityContext\ ins \\ &\quad \quad \quad (NS\ s\ (\text{exec}\ cmd))\ (Out\ s\ (\text{exec}\ cmd)::outs)) \iff \\ &\quad authenticationTest\ (P\ \text{says}\ \text{prop}\ (SOME\ cmd)) \wedge \\ &\quad CFGInterpret\ (M, Oi, Os) \\ &\quad \quad (CFG\ authenticationTest\ stateInterp\ securityContext \\ &\quad \quad \quad (P\ \text{says}\ \text{prop}\ (SOME\ cmd)::ins)\ s\ outs) \wedge \\ &\quad (M, Oi, Os)\ \text{sat}\ \text{prop}\ (SOME\ cmd)) \end{aligned}$$

[TR\_ind]

$$\begin{aligned} &\vdash \forall TR'. \\ &\quad (\forall authenticationTest\ P\ NS\ M\ Oi\ Os\ Out\ s\ securityContext \\ &\quad \quad stateInterp\ cmd\ ins\ outs. \\ &\quad \quad authenticationTest\ (P\ \text{says}\ \text{prop}\ (SOME\ cmd)) \wedge \\ &\quad \quad CFGInterpret\ (M, Oi, Os) \\ &\quad \quad \quad (CFG\ authenticationTest\ stateInterp\ securityContext \\ &\quad \quad \quad \quad (P\ \text{says}\ \text{prop}\ (SOME\ cmd)::ins)\ s\ outs) \Rightarrow \\ &\quad \quad TR'\ (M, Oi, Os)\ (\text{exec}\ cmd) \\ &\quad \quad \quad (CFG\ authenticationTest\ stateInterp\ securityContext \\ &\quad \quad \quad \quad (P\ \text{says}\ \text{prop}\ (SOME\ cmd)::ins)\ s\ outs) \\ &\quad \quad \quad (CFG\ authenticationTest\ stateInterp\ securityContext\ ins \\ &\quad \quad \quad \quad (NS\ s\ (\text{exec}\ cmd))\ (Out\ s\ (\text{exec}\ cmd)::outs))) \wedge \\ &\quad (\forall authenticationTest\ P\ NS\ M\ Oi\ Os\ Out\ s\ securityContext \\ &\quad \quad stateInterp\ cmd\ ins\ outs. \\ &\quad \quad authenticationTest\ (P\ \text{says}\ \text{prop}\ (SOME\ cmd)) \wedge \\ &\quad \quad CFGInterpret\ (M, Oi, Os) \\ &\quad \quad \quad (CFG\ authenticationTest\ stateInterp\ securityContext \\ &\quad \quad \quad \quad (P\ \text{says}\ \text{prop}\ (SOME\ cmd)::ins)\ s\ outs) \Rightarrow \\ &\quad \quad TR'\ (M, Oi, Os)\ (\text{trap}\ cmd) \\ &\quad \quad \quad (CFG\ authenticationTest\ stateInterp\ securityContext \\ &\quad \quad \quad \quad (P\ \text{says}\ \text{prop}\ (SOME\ cmd)::ins)\ s\ outs) \\ &\quad \quad \quad (CFG\ authenticationTest\ stateInterp\ securityContext\ ins \\ &\quad \quad \quad \quad (NS\ s\ (\text{trap}\ cmd))\ (Out\ s\ (\text{trap}\ cmd)::outs))) \wedge \\ &\quad (\forall authenticationTest\ NS\ M\ Oi\ Os\ Out\ s\ securityContext \\ &\quad \quad stateInterp\ cmd\ x\ ins\ outs. \end{aligned}$$



$$\begin{aligned}
& \neg \text{authenticationTest } x \Rightarrow \\
& \text{TR}' (M, Oi, Os) (\text{discard } cmd) \\
& \quad (\text{CFG authenticationTest stateInterp securityContext} \\
& \quad \quad (x :: ins) s outs) \\
& \quad (\text{CFG authenticationTest stateInterp securityContext} \\
& \quad \quad ins (NS s (\text{discard } cmd))) \\
& \quad (\text{Out } s (\text{discard } cmd) :: outs))) \Rightarrow \\
& \forall a_0 a_1 a_2 a_3. \text{TR } a_0 a_1 a_2 a_3 \Rightarrow \text{TR}' a_0 a_1 a_2 a_3
\end{aligned}$$

## [TR\_rules]

$$\begin{aligned}
& \vdash (\forall \text{authenticationTest } P \text{ NS } M \text{ Oi } Os \text{ Out } s \text{ securityContext} \\
& \quad \text{stateInterp } cmd \text{ ins } outs. \\
& \quad \text{authenticationTest } (P \text{ says prop (SOME } cmd)) \wedge \\
& \quad \text{CFGInterpret } (M, Oi, Os) \\
& \quad \quad (\text{CFG authenticationTest stateInterp securityContext} \\
& \quad \quad \quad (P \text{ says prop (SOME } cmd) :: ins) s outs) \Rightarrow \\
& \quad \text{TR } (M, Oi, Os) (\text{exec } cmd) \\
& \quad \quad (\text{CFG authenticationTest stateInterp securityContext} \\
& \quad \quad \quad (P \text{ says prop (SOME } cmd) :: ins) s outs) \\
& \quad \quad (\text{CFG authenticationTest stateInterp securityContext ins} \\
& \quad \quad \quad (NS s (\text{exec } cmd)) (\text{Out } s (\text{exec } cmd) :: outs))) \wedge \\
& (\forall \text{authenticationTest } P \text{ NS } M \text{ Oi } Os \text{ Out } s \text{ securityContext} \\
& \quad \text{stateInterp } cmd \text{ ins } outs. \\
& \quad \text{authenticationTest } (P \text{ says prop (SOME } cmd)) \wedge \\
& \quad \text{CFGInterpret } (M, Oi, Os) \\
& \quad \quad (\text{CFG authenticationTest stateInterp securityContext} \\
& \quad \quad \quad (P \text{ says prop (SOME } cmd) :: ins) s outs) \Rightarrow \\
& \quad \text{TR } (M, Oi, Os) (\text{trap } cmd) \\
& \quad \quad (\text{CFG authenticationTest stateInterp securityContext} \\
& \quad \quad \quad (P \text{ says prop (SOME } cmd) :: ins) s outs) \\
& \quad \quad (\text{CFG authenticationTest stateInterp securityContext ins} \\
& \quad \quad \quad (NS s (\text{trap } cmd)) (\text{Out } s (\text{trap } cmd) :: outs))) \wedge \\
& \forall \text{authenticationTest } NS \text{ M } Oi \text{ Os } Out \text{ s securityContext} \\
& \quad \text{stateInterp } cmd \text{ x ins } outs. \\
& \neg \text{authenticationTest } x \Rightarrow \\
& \text{TR } (M, Oi, Os) (\text{discard } cmd) \\
& \quad (\text{CFG authenticationTest stateInterp securityContext} \\
& \quad \quad (x :: ins) s outs) \\
& \quad (\text{CFG authenticationTest stateInterp securityContext ins} \\
& \quad \quad (NS s (\text{discard } cmd)) (\text{Out } s (\text{discard } cmd) :: outs)))
\end{aligned}$$

## [TR\_strongind]

$$\begin{aligned}
& \vdash \forall \text{TR}'. \\
& \quad (\forall \text{authenticationTest } P \text{ NS } M \text{ Oi } Os \text{ Out } s \text{ securityContext} \\
& \quad \quad \text{stateInterp } cmd \text{ ins } outs. \\
& \quad \quad \text{authenticationTest } (P \text{ says prop (SOME } cmd)) \wedge \\
& \quad \quad \text{CFGInterpret } (M, Oi, Os) \\
& \quad \quad \quad (\text{CFG authenticationTest stateInterp securityContext} \\
& \quad \quad \quad \quad (P \text{ says prop (SOME } cmd) :: ins) s outs) \Rightarrow
\end{aligned}$$

$$\begin{aligned}
& TR' (M, Oi, Os) (\text{exec } cmd) \\
& \quad (CFG \text{ authenticationTest } stateInterp \text{ securityContext} \\
& \quad \quad (P \text{ says prop } (SOME \text{ cmd}) :: ins) \text{ s } outs) \\
& \quad (CFG \text{ authenticationTest } stateInterp \text{ securityContext} \\
& \quad \quad \quad ins (NS \text{ s } (\text{exec } cmd)) (Out \text{ s } (\text{exec } cmd) :: outs))) \wedge \\
& (\forall \text{ authenticationTest } P \text{ NS } M \text{ Oi } Os \text{ Out } s \text{ securityContext} \\
& \quad \text{stateInterp } cmd \text{ ins } outs. \\
& \text{authenticationTest } (P \text{ says prop } (SOME \text{ cmd})) \wedge \\
& \text{CFGInterpret } (M, Oi, Os) \\
& \quad (CFG \text{ authenticationTest } stateInterp \text{ securityContext} \\
& \quad \quad (P \text{ says prop } (SOME \text{ cmd}) :: ins) \text{ s } outs) \Rightarrow \\
& TR' (M, Oi, Os) (\text{trap } cmd) \\
& \quad (CFG \text{ authenticationTest } stateInterp \text{ securityContext} \\
& \quad \quad (P \text{ says prop } (SOME \text{ cmd}) :: ins) \text{ s } outs) \\
& \quad (CFG \text{ authenticationTest } stateInterp \text{ securityContext} \\
& \quad \quad \quad ins (NS \text{ s } (\text{trap } cmd)) (Out \text{ s } (\text{trap } cmd) :: outs))) \wedge \\
& (\forall \text{ authenticationTest } NS \text{ M } Oi \text{ Os } Out \text{ s } \text{ securityContext} \\
& \quad \text{stateInterp } cmd \text{ x } ins \text{ outs}. \\
& \neg \text{authenticationTest } x \Rightarrow \\
& TR' (M, Oi, Os) (\text{discard } cmd) \\
& \quad (CFG \text{ authenticationTest } stateInterp \text{ securityContext} \\
& \quad \quad (x :: ins) \text{ s } outs) \\
& \quad (CFG \text{ authenticationTest } stateInterp \text{ securityContext} \\
& \quad \quad \quad ins (NS \text{ s } (\text{discard } cmd)) \\
& \quad \quad \quad \quad (Out \text{ s } (\text{discard } cmd) :: outs))) \Rightarrow \\
& \forall a_0 \ a_1 \ a_2 \ a_3. TR \ a_0 \ a_1 \ a_2 \ a_3 \Rightarrow TR' \ a_0 \ a_1 \ a_2 \ a_3
\end{aligned}$$

[TR\_trap\_cmd\_rule]

$$\begin{aligned}
& \vdash \forall \text{ authenticationTest } stateInterp \text{ securityContext } P \text{ cmd } ins \text{ s } \\
& \quad \text{outs}. \\
& (\forall M \text{ Oi } Os. \\
& \quad \text{CFGInterpret } (M, Oi, Os) \\
& \quad \quad (CFG \text{ authenticationTest } stateInterp \text{ securityContext} \\
& \quad \quad \quad (P \text{ says prop } (SOME \text{ cmd}) :: ins) \text{ s } outs) \Rightarrow \\
& \quad \quad (M, Oi, Os) \text{ sat prop NONE}) \Rightarrow \\
& \forall NS \text{ Out } M \text{ Oi } Os. \\
& \quad TR (M, Oi, Os) (\text{trap } cmd) \\
& \quad \quad (CFG \text{ authenticationTest } stateInterp \text{ securityContext} \\
& \quad \quad \quad (P \text{ says prop } (SOME \text{ cmd}) :: ins) \text{ s } outs) \\
& \quad \quad (CFG \text{ authenticationTest } stateInterp \text{ securityContext } ins \\
& \quad \quad \quad \quad (NS \text{ s } (\text{trap } cmd)) (Out \text{ s } (\text{trap } cmd) :: outs)) \iff \\
& \quad \text{authenticationTest } (P \text{ says prop } (SOME \text{ cmd})) \wedge \\
& \quad \text{CFGInterpret } (M, Oi, Os) \\
& \quad \quad (CFG \text{ authenticationTest } stateInterp \text{ securityContext} \\
& \quad \quad \quad (P \text{ says prop } (SOME \text{ cmd}) :: ins) \text{ s } outs) \wedge \\
& \quad \quad (M, Oi, Os) \text{ sat prop NONE}
\end{aligned}$$

[TRrule0]

$$\begin{aligned}
& \vdash TR (M, Oi, Os) (\text{exec } cmd) \\
& \quad (CFG \text{ authenticationTest } stateInterp \text{ securityContext}
\end{aligned}$$

---

```

(P says prop (SOME cmd)::ins) s outs)
(CFG authenticationTest stateInterp securityContext ins
 (NS s (exec cmd)) (Out s (exec cmd)::outs))  $\iff$ 
authenticationTest (P says prop (SOME cmd))  $\wedge$ 
CFGInterpret (M, Oi, Os)
 (CFG authenticationTest stateInterp securityContext
  (P says prop (SOME cmd)::ins) s outs)

```

[TRrule1]

```

 $\vdash$  TR (M, Oi, Os) (trap cmd)
  (CFG authenticationTest stateInterp securityContext
   (P says prop (SOME cmd)::ins) s outs)
  (CFG authenticationTest stateInterp securityContext ins
   (NS s (trap cmd)) (Out s (trap cmd)::outs))  $\iff$ 
authenticationTest (P says prop (SOME cmd))  $\wedge$ 
CFGInterpret (M, Oi, Os)
  (CFG authenticationTest stateInterp securityContext
   (P says prop (SOME cmd)::ins) s outs)

```

[trType\_distinct\_clauses]

```

 $\vdash (\forall a' a. \text{discard } a \neq \text{trap } a') \wedge (\forall a' a. \text{discard } a \neq \text{exec } a') \wedge$ 
 $\forall a' a. \text{trap } a \neq \text{exec } a'$ 

```

[trType\_one\_one]

```

 $\vdash (\forall a a'. (\text{discard } a = \text{discard } a') \iff (a = a')) \wedge$ 
 $(\forall a a'. (\text{trap } a = \text{trap } a') \iff (a = a')) \wedge$ 
 $\forall a a'. (\text{exec } a = \text{exec } a') \iff (a = a')$ 

```

### 3 ssm Theory

**Built:** 10 June 2018

**Parent Theories:** satList

#### 3.1 Datatypes

```

configuration =
  CFG (('command option, 'principal, 'd, 'e) Form -> bool)
    ('state ->
      ('command option, 'principal, 'd, 'e) Form list ->
        ('command option, 'principal, 'd, 'e) Form list)
      (('command option, 'principal, 'd, 'e) Form list ->
        ('command option, 'principal, 'd, 'e) Form list)
      (('command option, 'principal, 'd, 'e) Form list list)
      'state ('output list)

trType = discard 'cmdlist | trap 'cmdlist | exec 'cmdlist

```

### 3.2 Definitions

[authenticationTest\_def]

$$\vdash \forall \text{elementTest } x. \\ \text{authenticationTest } \text{elementTest } x \iff \\ \text{FOLDR } (\lambda p \ q. \ p \wedge \ q) \ \text{T} \ (\text{MAP } \text{elementTest } x)$$

[commandList\_def]

$$\vdash \forall x. \text{commandList } x = \text{MAP } \text{extractCommand } x$$

[inputList\_def]

$$\vdash \forall xs. \text{inputList } xs = \text{MAP } \text{extractInput } xs$$

[propCommandList\_def]

$$\vdash \forall x. \text{propCommandList } x = \text{MAP } \text{extractPropCommand } x$$

[TR\_def]

$$\vdash \text{TR} = \\ (\lambda a_0 \ a_1 \ a_2 \ a_3. \\ \forall TR'. \\ (\forall a_0 \ a_1 \ a_2 \ a_3. \\ (\exists \text{elementTest } NS \ M \ Oi \ Os \ Out \ s \ \text{context } \text{stateInterp } x \\ \text{ins } \text{outs}. \\ (a_0 = (M, Oi, Os)) \wedge (a_1 = \text{exec } (\text{inputList } x)) \wedge \\ (a_2 = \\ \text{CFG } \text{elementTest } \text{stateInterp } \text{context } (x::\text{ins}) \ s \\ \text{outs}) \wedge \\ (a_3 = \\ \text{CFG } \text{elementTest } \text{stateInterp } \text{context } \text{ins} \\ (NS \ s \ (\text{exec } (\text{inputList } x))) \\ (Out \ s \ (\text{exec } (\text{inputList } x)::\text{outs})) \wedge \\ \text{authenticationTest } \text{elementTest } x \wedge \\ \text{CFGInterpret } (M, Oi, Os) \\ (\text{CFG } \text{elementTest } \text{stateInterp } \text{context } (x::\text{ins}) \ s \\ \text{outs})) \vee \\ (\exists \text{elementTest } NS \ M \ Oi \ Os \ Out \ s \ \text{context } \text{stateInterp } x \\ \text{ins } \text{outs}. \\ (a_0 = (M, Oi, Os)) \wedge (a_1 = \text{trap } (\text{inputList } x)) \wedge \\ (a_2 = \\ \text{CFG } \text{elementTest } \text{stateInterp } \text{context } (x::\text{ins}) \ s \\ \text{outs}) \wedge \\ (a_3 = \\ \text{CFG } \text{elementTest } \text{stateInterp } \text{context } \text{ins} \\ (NS \ s \ (\text{trap } (\text{inputList } x))) \\ (Out \ s \ (\text{trap } (\text{inputList } x)::\text{outs})) \wedge \\ \text{authenticationTest } \text{elementTest } x \wedge \\ \text{CFGInterpret } (M, Oi, Os) \\ (\text{CFG } \text{elementTest } \text{stateInterp } \text{context } (x::\text{ins}) \ s$$

$$\begin{aligned}
& \text{outs})) \vee \\
& (\exists \text{elementTest } NS \ M \ Oi \ Os \ Out \ s \ \text{context } stateInterp \ x \\
& \quad \text{ins } outs. \\
& \quad (a_0 = (M, Oi, Os)) \wedge (a_1 = \text{discard } (\text{inputList } x)) \wedge \\
& \quad (a_2 = \\
& \quad \quad \text{CFG } \text{elementTest } stateInterp \ \text{context } (x::ins) \ s \\
& \quad \quad \text{outs}) \wedge \\
& \quad (a_3 = \\
& \quad \quad \text{CFG } \text{elementTest } stateInterp \ \text{context } ins \\
& \quad \quad (NS \ s \ (\text{discard } (\text{inputList } x))) \\
& \quad \quad (Out \ s \ (\text{discard } (\text{inputList } x))::outs)) \wedge \\
& \quad \neg \text{authenticationTest } \text{elementTest } x) \Rightarrow \\
& TR' \ a_0 \ a_1 \ a_2 \ a_3) \Rightarrow \\
& TR' \ a_0 \ a_1 \ a_2 \ a_3)
\end{aligned}$$

### 3.3 Theorems

[CFGInterpret\_def]

$$\begin{aligned}
& \vdash \text{CFGInterpret } (M, Oi, Os) \\
& \quad (\text{CFG } \text{elementTest } stateInterp \ \text{context } (x::ins) \ state \\
& \quad \quad \text{outStream}) \iff \\
& \quad (M, Oi, Os) \ \text{satList } \text{context } x \wedge (M, Oi, Os) \ \text{satList } x \wedge \\
& \quad (M, Oi, Os) \ \text{satList } stateInterp \ state \ x
\end{aligned}$$

[CFGInterpret\_ind]

$$\begin{aligned}
& \vdash \forall P. \\
& \quad (\forall M \ Oi \ Os \ \text{elementTest } stateInterp \ \text{context } x \ \text{ins } state \\
& \quad \quad \text{outStream}. \\
& \quad \quad P \ (M, Oi, Os) \\
& \quad \quad (\text{CFG } \text{elementTest } stateInterp \ \text{context } (x::ins) \ state \\
& \quad \quad \quad \text{outStream})) \wedge \\
& \quad (\forall v_{15} \ v_{10} \ v_{11} \ v_{12} \ v_{13} \ v_{14}. \\
& \quad \quad P \ v_{15} \ (\text{CFG } v_{10} \ v_{11} \ v_{12} \ [] \ v_{13} \ v_{14})) \Rightarrow \\
& \quad \forall v \ v_1 \ v_2 \ v_3. \ P \ (v, v_1, v_2) \ v_3
\end{aligned}$$

[configuration\_one\_one]

$$\begin{aligned}
& \vdash \forall a_0 \ a_1 \ a_2 \ a_3 \ a_4 \ a_5 \ a'_0 \ a'_1 \ a'_2 \ a'_3 \ a'_4 \ a'_5. \\
& \quad (\text{CFG } a_0 \ a_1 \ a_2 \ a_3 \ a_4 \ a_5 = \text{CFG } a'_0 \ a'_1 \ a'_2 \ a'_3 \ a'_4 \ a'_5) \iff \\
& \quad (a_0 = a'_0) \wedge (a_1 = a'_1) \wedge (a_2 = a'_2) \wedge (a_3 = a'_3) \wedge \\
& \quad (a_4 = a'_4) \wedge (a_5 = a'_5)
\end{aligned}$$

[extractCommand\_def]

$$\vdash \text{extractCommand } (P \ \text{says prop } (\text{SOME } cmd)) = cmd$$

[extractCommand\_ind]

$$\begin{aligned}
& \vdash \forall P'. \\
& \quad (\forall P \ cmd. \ P' \ (P \ \text{says prop } (\text{SOME } cmd))) \wedge P' \ \text{TT} \wedge P' \ \text{FF} \wedge \\
& \quad (\forall v_1. \ P' \ (\text{prop } v_1)) \wedge (\forall v_3. \ P' \ (\text{notf } v_3)) \wedge
\end{aligned}$$

$$\begin{aligned}
& (\forall v_6 v_7. P' (v_6 \text{ andf } v_7)) \wedge (\forall v_{10} v_{11}. P' (v_{10} \text{ orf } v_{11})) \wedge \\
& (\forall v_{14} v_{15}. P' (v_{14} \text{ impf } v_{15})) \wedge \\
& (\forall v_{18} v_{19}. P' (v_{18} \text{ eqf } v_{19})) \wedge (\forall v_{129}. P' (v_{129} \text{ says TT})) \wedge \\
& (\forall v_{130}. P' (v_{130} \text{ says FF})) \wedge \\
& (\forall v_{132}. P' (v_{132} \text{ says prop NONE})) \wedge \\
& (\forall v_{133} v_{66}. P' (v_{133} \text{ says notf } v_{66})) \wedge \\
& (\forall v_{134} v_{69} v_{70}. P' (v_{134} \text{ says } (v_{69} \text{ andf } v_{70}))) \wedge \\
& (\forall v_{135} v_{73} v_{74}. P' (v_{135} \text{ says } (v_{73} \text{ orf } v_{74}))) \wedge \\
& (\forall v_{136} v_{77} v_{78}. P' (v_{136} \text{ says } (v_{77} \text{ impf } v_{78}))) \wedge \\
& (\forall v_{137} v_{81} v_{82}. P' (v_{137} \text{ says } (v_{81} \text{ eqf } v_{82}))) \wedge \\
& (\forall v_{138} v_{85} v_{86}. P' (v_{138} \text{ says } v_{85} \text{ says } v_{86})) \wedge \\
& (\forall v_{139} v_{89} v_{90}. P' (v_{139} \text{ says } v_{89} \text{ speaks\_for } v_{90})) \wedge \\
& (\forall v_{140} v_{93} v_{94}. P' (v_{140} \text{ says } v_{93} \text{ controls } v_{94})) \wedge \\
& (\forall v_{141} v_{98} v_{99} v_{100}. P' (v_{141} \text{ says reps } v_{98} v_{99} v_{100})) \wedge \\
& (\forall v_{142} v_{103} v_{104}. P' (v_{142} \text{ says } v_{103} \text{ domi } v_{104})) \wedge \\
& (\forall v_{143} v_{107} v_{108}. P' (v_{143} \text{ says } v_{107} \text{ eqi } v_{108})) \wedge \\
& (\forall v_{144} v_{111} v_{112}. P' (v_{144} \text{ says } v_{111} \text{ doms } v_{112})) \wedge \\
& (\forall v_{145} v_{115} v_{116}. P' (v_{145} \text{ says } v_{115} \text{ eqs } v_{116})) \wedge \\
& (\forall v_{146} v_{119} v_{120}. P' (v_{146} \text{ says } v_{119} \text{ eqn } v_{120})) \wedge \\
& (\forall v_{147} v_{123} v_{124}. P' (v_{147} \text{ says } v_{123} \text{ lte } v_{124})) \wedge \\
& (\forall v_{148} v_{127} v_{128}. P' (v_{148} \text{ says } v_{127} \text{ lt } v_{128})) \wedge \\
& (\forall v_{24} v_{25}. P' (v_{24} \text{ speaks\_for } v_{25})) \wedge \\
& (\forall v_{28} v_{29}. P' (v_{28} \text{ controls } v_{29})) \wedge \\
& (\forall v_{33} v_{34} v_{35}. P' (\text{reps } v_{33} v_{34} v_{35})) \wedge \\
& (\forall v_{38} v_{39}. P' (v_{38} \text{ domi } v_{39})) \wedge \\
& (\forall v_{42} v_{43}. P' (v_{42} \text{ eqi } v_{43})) \wedge \\
& (\forall v_{46} v_{47}. P' (v_{46} \text{ doms } v_{47})) \wedge \\
& (\forall v_{50} v_{51}. P' (v_{50} \text{ eqs } v_{51})) \wedge \\
& (\forall v_{54} v_{55}. P' (v_{54} \text{ eqn } v_{55})) \wedge \\
& (\forall v_{58} v_{59}. P' (v_{58} \text{ lte } v_{59})) \wedge \\
& (\forall v_{62} v_{63}. P' (v_{62} \text{ lt } v_{63})) \Rightarrow \\
& \forall v. P' v
\end{aligned}$$

[extractInput\_def]

$\vdash \text{extractInput } (P \text{ says prop } x) = x$

[extractInput\_ind]

$\vdash \forall P'.$

$$\begin{aligned}
& (\forall P x. P' (P \text{ says prop } x)) \wedge P' \text{ TT} \wedge P' \text{ FF} \wedge \\
& (\forall v_1. P' (\text{prop } v_1)) \wedge (\forall v_3. P' (\text{notf } v_3)) \wedge \\
& (\forall v_6 v_7. P' (v_6 \text{ andf } v_7)) \wedge (\forall v_{10} v_{11}. P' (v_{10} \text{ orf } v_{11})) \wedge \\
& (\forall v_{14} v_{15}. P' (v_{14} \text{ impf } v_{15})) \wedge \\
& (\forall v_{18} v_{19}. P' (v_{18} \text{ eqf } v_{19})) \wedge (\forall v_{129}. P' (v_{129} \text{ says TT})) \wedge \\
& (\forall v_{130}. P' (v_{130} \text{ says FF})) \wedge \\
& (\forall v_{131} v_{66}. P' (v_{131} \text{ says notf } v_{66})) \wedge \\
& (\forall v_{132} v_{69} v_{70}. P' (v_{132} \text{ says } (v_{69} \text{ andf } v_{70}))) \wedge \\
& (\forall v_{133} v_{73} v_{74}. P' (v_{133} \text{ says } (v_{73} \text{ orf } v_{74}))) \wedge \\
& (\forall v_{134} v_{77} v_{78}. P' (v_{134} \text{ says } (v_{77} \text{ impf } v_{78}))) \wedge \\
& (\forall v_{135} v_{81} v_{82}. P' (v_{135} \text{ says } (v_{81} \text{ eqf } v_{82}))) \wedge
\end{aligned}$$

$$\begin{aligned}
& (\forall v136 \ v85 \ v86. \ P' \ (v136 \ \text{says} \ v85 \ \text{says} \ v86)) \wedge \\
& (\forall v137 \ v89 \ v90. \ P' \ (v137 \ \text{says} \ v89 \ \text{speaks\_for} \ v90)) \wedge \\
& (\forall v138 \ v93 \ v94. \ P' \ (v138 \ \text{says} \ v93 \ \text{controls} \ v94)) \wedge \\
& (\forall v139 \ v98 \ v99 \ v100. \ P' \ (v139 \ \text{says} \ \text{reps} \ v98 \ v99 \ v100)) \wedge \\
& (\forall v140 \ v103 \ v104. \ P' \ (v140 \ \text{says} \ v103 \ \text{domi} \ v104)) \wedge \\
& (\forall v141 \ v107 \ v108. \ P' \ (v141 \ \text{says} \ v107 \ \text{eqi} \ v108)) \wedge \\
& (\forall v142 \ v111 \ v112. \ P' \ (v142 \ \text{says} \ v111 \ \text{doms} \ v112)) \wedge \\
& (\forall v143 \ v115 \ v116. \ P' \ (v143 \ \text{says} \ v115 \ \text{eqs} \ v116)) \wedge \\
& (\forall v144 \ v119 \ v120. \ P' \ (v144 \ \text{says} \ v119 \ \text{eqn} \ v120)) \wedge \\
& (\forall v145 \ v123 \ v124. \ P' \ (v145 \ \text{says} \ v123 \ \text{lte} \ v124)) \wedge \\
& (\forall v146 \ v127 \ v128. \ P' \ (v146 \ \text{says} \ v127 \ \text{lt} \ v128)) \wedge \\
& (\forall v24 \ v25. \ P' \ (v24 \ \text{speaks\_for} \ v25)) \wedge \\
& (\forall v28 \ v29. \ P' \ (v28 \ \text{controls} \ v29)) \wedge \\
& (\forall v33 \ v34 \ v35. \ P' \ (\text{reps} \ v33 \ v34 \ v35)) \wedge \\
& (\forall v38 \ v39. \ P' \ (v38 \ \text{domi} \ v39)) \wedge \\
& (\forall v42 \ v43. \ P' \ (v42 \ \text{eqi} \ v43)) \wedge \\
& (\forall v46 \ v47. \ P' \ (v46 \ \text{doms} \ v47)) \wedge \\
& (\forall v50 \ v51. \ P' \ (v50 \ \text{eqs} \ v51)) \wedge \\
& (\forall v54 \ v55. \ P' \ (v54 \ \text{eqn} \ v55)) \wedge \\
& (\forall v58 \ v59. \ P' \ (v58 \ \text{lte} \ v59)) \wedge \\
& (\forall v62 \ v63. \ P' \ (v62 \ \text{lt} \ v63)) \Rightarrow \\
& \forall v. \ P' \ v
\end{aligned}$$

[extractPropCommand\_def]

$$\vdash \text{extractPropCommand} \ (P \ \text{says} \ \text{prop} \ (\text{SOME} \ \text{cmd})) = \text{prop} \ (\text{SOME} \ \text{cmd})$$

[extractPropCommand\_ind]

$$\begin{aligned}
& \vdash \forall P'. \\
& \quad (\forall P \ \text{cmd}. \ P' \ (P \ \text{says} \ \text{prop} \ (\text{SOME} \ \text{cmd}))) \wedge P' \ \text{TT} \wedge P' \ \text{FF} \wedge \\
& \quad (\forall v_1. \ P' \ (\text{prop} \ v_1)) \wedge (\forall v_3. \ P' \ (\text{notf} \ v_3)) \wedge \\
& \quad (\forall v_6 \ v_7. \ P' \ (v_6 \ \text{andf} \ v_7)) \wedge (\forall v_{10} \ v_{11}. \ P' \ (v_{10} \ \text{orf} \ v_{11})) \wedge \\
& \quad (\forall v_{14} \ v_{15}. \ P' \ (v_{14} \ \text{impf} \ v_{15})) \wedge \\
& \quad (\forall v_{18} \ v_{19}. \ P' \ (v_{18} \ \text{eqf} \ v_{19})) \wedge (\forall v_{129}. \ P' \ (v_{129} \ \text{says} \ \text{TT})) \wedge \\
& \quad (\forall v_{130}. \ P' \ (v_{130} \ \text{says} \ \text{FF})) \wedge \\
& \quad (\forall v_{132}. \ P' \ (v_{132} \ \text{says} \ \text{prop} \ \text{NONE})) \wedge \\
& \quad (\forall v_{133} \ v_{66}. \ P' \ (v_{133} \ \text{says} \ \text{notf} \ v_{66})) \wedge \\
& \quad (\forall v_{134} \ v_{69} \ v_{70}. \ P' \ (v_{134} \ \text{says} \ (v_{69} \ \text{andf} \ v_{70}))) \wedge \\
& \quad (\forall v_{135} \ v_{73} \ v_{74}. \ P' \ (v_{135} \ \text{says} \ (v_{73} \ \text{orf} \ v_{74}))) \wedge \\
& \quad (\forall v_{136} \ v_{77} \ v_{78}. \ P' \ (v_{136} \ \text{says} \ (v_{77} \ \text{impf} \ v_{78}))) \wedge \\
& \quad (\forall v_{137} \ v_{81} \ v_{82}. \ P' \ (v_{137} \ \text{says} \ (v_{81} \ \text{eqf} \ v_{82}))) \wedge \\
& \quad (\forall v_{138} \ v_{85} \ v_{86}. \ P' \ (v_{138} \ \text{says} \ v_{85} \ \text{says} \ v_{86})) \wedge \\
& \quad (\forall v_{139} \ v_{89} \ v_{90}. \ P' \ (v_{139} \ \text{says} \ v_{89} \ \text{speaks\_for} \ v_{90})) \wedge \\
& \quad (\forall v_{140} \ v_{93} \ v_{94}. \ P' \ (v_{140} \ \text{says} \ v_{93} \ \text{controls} \ v_{94})) \wedge \\
& \quad (\forall v_{141} \ v_{98} \ v_{99} \ v_{100}. \ P' \ (v_{141} \ \text{says} \ \text{reps} \ v_{98} \ v_{99} \ v_{100})) \wedge \\
& \quad (\forall v_{142} \ v_{103} \ v_{104}. \ P' \ (v_{142} \ \text{says} \ v_{103} \ \text{domi} \ v_{104})) \wedge \\
& \quad (\forall v_{143} \ v_{107} \ v_{108}. \ P' \ (v_{143} \ \text{says} \ v_{107} \ \text{eqi} \ v_{108})) \wedge \\
& \quad (\forall v_{144} \ v_{111} \ v_{112}. \ P' \ (v_{144} \ \text{says} \ v_{111} \ \text{doms} \ v_{112})) \wedge \\
& \quad (\forall v_{145} \ v_{115} \ v_{116}. \ P' \ (v_{145} \ \text{says} \ v_{115} \ \text{eqs} \ v_{116})) \wedge \\
& \quad (\forall v_{146} \ v_{119} \ v_{120}. \ P' \ (v_{146} \ \text{says} \ v_{119} \ \text{eqn} \ v_{120})) \wedge
\end{aligned}$$

$$\begin{aligned}
& (\forall v_{147} v_{123} v_{124}. P' (v_{147} \text{ says } v_{123} \text{ lte } v_{124})) \wedge \\
& (\forall v_{148} v_{127} v_{128}. P' (v_{148} \text{ says } v_{127} \text{ lt } v_{128})) \wedge \\
& (\forall v_{24} v_{25}. P' (v_{24} \text{ speaks\_for } v_{25})) \wedge \\
& (\forall v_{28} v_{29}. P' (v_{28} \text{ controls } v_{29})) \wedge \\
& (\forall v_{33} v_{34} v_{35}. P' (\text{reps } v_{33} v_{34} v_{35})) \wedge \\
& (\forall v_{38} v_{39}. P' (v_{38} \text{ domi } v_{39})) \wedge \\
& (\forall v_{42} v_{43}. P' (v_{42} \text{ eqi } v_{43})) \wedge \\
& (\forall v_{46} v_{47}. P' (v_{46} \text{ doms } v_{47})) \wedge \\
& (\forall v_{50} v_{51}. P' (v_{50} \text{ eqs } v_{51})) \wedge \\
& (\forall v_{54} v_{55}. P' (v_{54} \text{ eqn } v_{55})) \wedge \\
& (\forall v_{58} v_{59}. P' (v_{58} \text{ lte } v_{59})) \wedge \\
& (\forall v_{62} v_{63}. P' (v_{62} \text{ lt } v_{63})) \Rightarrow \\
& \forall v. P' v
\end{aligned}$$

[TR\_cases]

$$\begin{aligned}
& \vdash \forall a_0 a_1 a_2 a_3. \\
& \text{TR } a_0 a_1 a_2 a_3 \iff \\
& (\exists \text{elementTest } NS \ M \ Oi \ Os \ Out \ s \ context \ stateInterp \ x \ ins \\
& \quad outs. \\
& \quad (a_0 = (M, Oi, Os)) \wedge (a_1 = \text{exec } (\text{inputList } x)) \wedge \\
& \quad (a_2 = \\
& \quad \quad \text{CFG elementTest stateInterp context } (x::ins) \ s \ outs) \wedge \\
& \quad (a_3 = \\
& \quad \quad \text{CFG elementTest stateInterp context } ins \\
& \quad \quad \quad (NS \ s \ (\text{exec } (\text{inputList } x))) \\
& \quad \quad \quad (Out \ s \ (\text{exec } (\text{inputList } x))::outs)) \wedge \\
& \quad \text{authenticationTest elementTest } x \wedge \\
& \quad \text{CFGInterpret } (M, Oi, Os) \\
& \quad \quad (\text{CFG elementTest stateInterp context } (x::ins) \ s \\
& \quad \quad \quad outs)) \vee \\
& (\exists \text{elementTest } NS \ M \ Oi \ Os \ Out \ s \ context \ stateInterp \ x \ ins \\
& \quad outs. \\
& \quad (a_0 = (M, Oi, Os)) \wedge (a_1 = \text{trap } (\text{inputList } x)) \wedge \\
& \quad (a_2 = \\
& \quad \quad \text{CFG elementTest stateInterp context } (x::ins) \ s \ outs) \wedge \\
& \quad (a_3 = \\
& \quad \quad \text{CFG elementTest stateInterp context } ins \\
& \quad \quad \quad (NS \ s \ (\text{trap } (\text{inputList } x))) \\
& \quad \quad \quad (Out \ s \ (\text{trap } (\text{inputList } x))::outs)) \wedge \\
& \quad \text{authenticationTest elementTest } x \wedge \\
& \quad \text{CFGInterpret } (M, Oi, Os) \\
& \quad \quad (\text{CFG elementTest stateInterp context } (x::ins) \ s \\
& \quad \quad \quad outs)) \vee \\
& \exists \text{elementTest } NS \ M \ Oi \ Os \ Out \ s \ context \ stateInterp \ x \ ins \\
& \quad outs. \\
& \quad (a_0 = (M, Oi, Os)) \wedge (a_1 = \text{discard } (\text{inputList } x)) \wedge \\
& \quad (a_2 = \\
& \quad \quad \text{CFG elementTest stateInterp context } (x::ins) \ s \ outs) \wedge \\
& \quad (a_3 =
\end{aligned}$$



CFG elementTest stateInterp context ins  
 (NS s (discard (inputList x)))  
 (Out s (discard (inputList x))::outs))  $\wedge$   
 $\neg$ authenticationTest elementTest x

[TR\_discard\_cmd\_rule]

$\vdash$  TR (M, Oi, Os) (discard (inputList x))  
 (CFG elementTest stateInterp context (x::ins) s outs)  
 (CFG elementTest stateInterp context ins  
 (NS s (discard (inputList x)))  
 (Out s (discard (inputList x))::outs))  $\iff$   
 $\neg$ authenticationTest elementTest x

[TR\_EQ\_rules\_thm]

$\vdash$  (TR (M, Oi, Os) (exec (inputList x))  
 (CFG elementTest stateInterp context (x::ins) s outs)  
 (CFG elementTest stateInterp context ins  
 (NS s (exec (inputList x)))  
 (Out s (exec (inputList x))::outs))  $\iff$   
 authenticationTest elementTest x  $\wedge$   
 CFGInterpret (M, Oi, Os)  
 (CFG elementTest stateInterp context (x::ins) s outs))  $\wedge$   
 (TR (M, Oi, Os) (trap (inputList x))  
 (CFG elementTest stateInterp context (x::ins) s outs)  
 (CFG elementTest stateInterp context ins  
 (NS s (trap (inputList x)))  
 (Out s (trap (inputList x))::outs))  $\iff$   
 authenticationTest elementTest x  $\wedge$   
 CFGInterpret (M, Oi, Os)  
 (CFG elementTest stateInterp context (x::ins) s outs))  $\wedge$   
 (TR (M, Oi, Os) (discard (inputList x))  
 (CFG elementTest stateInterp context (x::ins) s outs)  
 (CFG elementTest stateInterp context ins  
 (NS s (discard (inputList x)))  
 (Out s (discard (inputList x))::outs))  $\iff$   
 $\neg$ authenticationTest elementTest x)

[TR\_exec\_cmd\_rule]

$\vdash \forall$  elementTest context stateInterp x ins s outs.  
 ( $\forall$  M Oi Os.  
 CFGInterpret (M, Oi, Os)  
 (CFG elementTest stateInterp context (x::ins) s  
 outs)  $\Rightarrow$   
 (M, Oi, Os) satList propCommandList x)  $\Rightarrow$   
 $\forall$  NS Out M Oi Os.  
 TR (M, Oi, Os) (exec (inputList x))  
 (CFG elementTest stateInterp context (x::ins) s outs)  
 (CFG elementTest stateInterp context ins

$$\begin{aligned}
& (NS \ s \ (\text{exec} \ (\text{inputList} \ x))) \\
& (\text{Out} \ s \ (\text{exec} \ (\text{inputList} \ x))::\text{outs})) \iff \\
& \text{authenticationTest} \ \text{elementTest} \ x \wedge \\
& \text{CFGInterpret} \ (M, Oi, Os) \\
& \quad (\text{CFG} \ \text{elementTest} \ \text{stateInterp} \ \text{context} \ (x::\text{ins}) \ s \ \text{outs}) \wedge \\
& (M, Oi, Os) \ \text{satList} \ \text{propCommandList} \ x
\end{aligned}$$

[TR\_ind]

 $\vdash \forall TR'.$ 

$$\begin{aligned}
& (\forall \text{elementTest} \ NS \ M \ Oi \ Os \ Out \ s \ \text{context} \ \text{stateInterp} \ x \ \text{ins} \\
& \quad \text{outs}. \\
& \text{authenticationTest} \ \text{elementTest} \ x \wedge \\
& \text{CFGInterpret} \ (M, Oi, Os) \\
& \quad (\text{CFG} \ \text{elementTest} \ \text{stateInterp} \ \text{context} \ (x::\text{ins}) \ s \\
& \quad \text{outs}) \Rightarrow \\
& TR' \ (M, Oi, Os) \ (\text{exec} \ (\text{inputList} \ x)) \\
& \quad (\text{CFG} \ \text{elementTest} \ \text{stateInterp} \ \text{context} \ (x::\text{ins}) \ s \ \text{outs}) \\
& \quad (\text{CFG} \ \text{elementTest} \ \text{stateInterp} \ \text{context} \ \text{ins} \\
& \quad \quad (NS \ s \ (\text{exec} \ (\text{inputList} \ x))) \\
& \quad \quad (\text{Out} \ s \ (\text{exec} \ (\text{inputList} \ x))::\text{outs}))) \wedge \\
& (\forall \text{elementTest} \ NS \ M \ Oi \ Os \ Out \ s \ \text{context} \ \text{stateInterp} \ x \ \text{ins} \\
& \quad \text{outs}. \\
& \text{authenticationTest} \ \text{elementTest} \ x \wedge \\
& \text{CFGInterpret} \ (M, Oi, Os) \\
& \quad (\text{CFG} \ \text{elementTest} \ \text{stateInterp} \ \text{context} \ (x::\text{ins}) \ s \\
& \quad \text{outs}) \Rightarrow \\
& TR' \ (M, Oi, Os) \ (\text{trap} \ (\text{inputList} \ x)) \\
& \quad (\text{CFG} \ \text{elementTest} \ \text{stateInterp} \ \text{context} \ (x::\text{ins}) \ s \ \text{outs}) \\
& \quad (\text{CFG} \ \text{elementTest} \ \text{stateInterp} \ \text{context} \ \text{ins} \\
& \quad \quad (NS \ s \ (\text{trap} \ (\text{inputList} \ x))) \\
& \quad \quad (\text{Out} \ s \ (\text{trap} \ (\text{inputList} \ x))::\text{outs}))) \wedge \\
& (\forall \text{elementTest} \ NS \ M \ Oi \ Os \ Out \ s \ \text{context} \ \text{stateInterp} \ x \ \text{ins} \\
& \quad \text{outs}. \\
& \neg \text{authenticationTest} \ \text{elementTest} \ x \Rightarrow \\
& TR' \ (M, Oi, Os) \ (\text{discard} \ (\text{inputList} \ x)) \\
& \quad (\text{CFG} \ \text{elementTest} \ \text{stateInterp} \ \text{context} \ (x::\text{ins}) \ s \ \text{outs}) \\
& \quad (\text{CFG} \ \text{elementTest} \ \text{stateInterp} \ \text{context} \ \text{ins} \\
& \quad \quad (NS \ s \ (\text{discard} \ (\text{inputList} \ x))) \\
& \quad \quad (\text{Out} \ s \ (\text{discard} \ (\text{inputList} \ x))::\text{outs}))) \Rightarrow \\
& \forall a_0 \ a_1 \ a_2 \ a_3. \ TR \ a_0 \ a_1 \ a_2 \ a_3 \Rightarrow TR' \ a_0 \ a_1 \ a_2 \ a_3
\end{aligned}$$

[TR\_rules]

$$\begin{aligned}
& \vdash (\forall \text{elementTest} \ NS \ M \ Oi \ Os \ Out \ s \ \text{context} \ \text{stateInterp} \ x \ \text{ins} \\
& \quad \text{outs}. \\
& \text{authenticationTest} \ \text{elementTest} \ x \wedge \\
& \text{CFGInterpret} \ (M, Oi, Os) \\
& \quad (\text{CFG} \ \text{elementTest} \ \text{stateInterp} \ \text{context} \ (x::\text{ins}) \ s \ \text{outs}) \Rightarrow \\
& TR \ (M, Oi, Os) \ (\text{exec} \ (\text{inputList} \ x)) \\
& \quad (\text{CFG} \ \text{elementTest} \ \text{stateInterp} \ \text{context} \ (x::\text{ins}) \ s \ \text{outs})
\end{aligned}$$

```

(CFG elementTest stateInterp context ins
  (NS s (exec (inputList x)))
  (Out s (exec (inputList x))::outs))) ∧
(∀ elementTest NS M Oi Os Out s context stateInterp x ins
  outs.
  authenticationTest elementTest x ∧
  CFGInterpret (M, Oi, Os)
    (CFG elementTest stateInterp context (x::ins) s outs) ⇒
  TR (M, Oi, Os) (trap (inputList x))
    (CFG elementTest stateInterp context (x::ins) s outs)
    (CFG elementTest stateInterp context ins
      (NS s (trap (inputList x)))
      (Out s (trap (inputList x))::outs))) ∧
  ∀ elementTest NS M Oi Os Out s context stateInterp x ins outs.
    ¬authenticationTest elementTest x ⇒
  TR (M, Oi, Os) (discard (inputList x))
    (CFG elementTest stateInterp context (x::ins) s outs)
    (CFG elementTest stateInterp context ins
      (NS s (discard (inputList x)))
      (Out s (discard (inputList x))::outs)))

```

[TR\_strongind]

```

⊢ ∀ TR'.
  (∀ elementTest NS M Oi Os Out s context stateInterp x ins
    outs.
    authenticationTest elementTest x ∧
    CFGInterpret (M, Oi, Os)
      (CFG elementTest stateInterp context (x::ins) s
        outs) ⇒
    TR' (M, Oi, Os) (exec (inputList x))
      (CFG elementTest stateInterp context (x::ins) s outs)
      (CFG elementTest stateInterp context ins
        (NS s (exec (inputList x)))
        (Out s (exec (inputList x))::outs))) ∧
    (∀ elementTest NS M Oi Os Out s context stateInterp x ins
      outs.
      authenticationTest elementTest x ∧
      CFGInterpret (M, Oi, Os)
        (CFG elementTest stateInterp context (x::ins) s
          outs) ⇒
      TR' (M, Oi, Os) (trap (inputList x))
        (CFG elementTest stateInterp context (x::ins) s outs)
        (CFG elementTest stateInterp context ins
          (NS s (trap (inputList x)))
          (Out s (trap (inputList x))::outs))) ∧
      (∀ elementTest NS M Oi Os Out s context stateInterp x ins
        outs.
        ¬authenticationTest elementTest x ⇒
        TR' (M, Oi, Os) (discard (inputList x))

```

$$\begin{aligned}
& (\text{CFG elementTest stateInterp context } (x::\text{ins}) \text{ s outs}) \\
& (\text{CFG elementTest stateInterp context ins} \\
& \quad (\text{NS s (discard (inputList x))}) \\
& \quad (\text{Out s (discard (inputList x))::outs})) \Rightarrow \\
& \forall a_0 \ a_1 \ a_2 \ a_3. \text{ TR } a_0 \ a_1 \ a_2 \ a_3 \Rightarrow \text{TR}' a_0 \ a_1 \ a_2 \ a_3
\end{aligned}$$

[TR\_trap\_cmd\_rule]

$$\begin{aligned}
& \vdash \forall \text{elementTest context stateInterp } x \text{ ins s outs.} \\
& \quad (\forall M \ Oi \ Os. \\
& \quad \quad \text{CFGInterpret } (M, Oi, Os) \\
& \quad \quad (\text{CFG elementTest stateInterp context } (x::\text{ins}) \text{ s} \\
& \quad \quad \quad \text{outs}) \Rightarrow \\
& \quad \quad (M, Oi, Os) \text{ sat prop NONE}) \Rightarrow \\
& \quad \forall \text{NS Out } M \ Oi \ Os. \\
& \quad \text{TR } (M, Oi, Os) (\text{trap (inputList x)}) \\
& \quad (\text{CFG elementTest stateInterp context } (x::\text{ins}) \text{ s outs}) \\
& \quad (\text{CFG elementTest stateInterp context ins} \\
& \quad \quad (\text{NS s (trap (inputList x))}) \\
& \quad \quad (\text{Out s (trap (inputList x))::outs})) \iff \\
& \quad \text{authenticationTest elementTest } x \wedge \\
& \quad \text{CFGInterpret } (M, Oi, Os) \\
& \quad \quad (\text{CFG elementTest stateInterp context } (x::\text{ins}) \text{ s outs}) \wedge \\
& \quad \quad (M, Oi, Os) \text{ sat prop NONE}
\end{aligned}$$

[TRrule0]

$$\begin{aligned}
& \vdash \text{TR } (M, Oi, Os) (\text{exec (inputList x)}) \\
& \quad (\text{CFG elementTest stateInterp context } (x::\text{ins}) \text{ s outs}) \\
& \quad (\text{CFG elementTest stateInterp context ins} \\
& \quad \quad (\text{NS s (exec (inputList x))}) \\
& \quad \quad (\text{Out s (exec (inputList x))::outs})) \iff \\
& \quad \text{authenticationTest elementTest } x \wedge \\
& \quad \text{CFGInterpret } (M, Oi, Os) \\
& \quad \quad (\text{CFG elementTest stateInterp context } (x::\text{ins}) \text{ s outs})
\end{aligned}$$

[TRrule1]

$$\begin{aligned}
& \vdash \text{TR } (M, Oi, Os) (\text{trap (inputList x)}) \\
& \quad (\text{CFG elementTest stateInterp context } (x::\text{ins}) \text{ s outs}) \\
& \quad (\text{CFG elementTest stateInterp context ins} \\
& \quad \quad (\text{NS s (trap (inputList x))}) \\
& \quad \quad (\text{Out s (trap (inputList x))::outs})) \iff \\
& \quad \text{authenticationTest elementTest } x \wedge \\
& \quad \text{CFGInterpret } (M, Oi, Os) \\
& \quad \quad (\text{CFG elementTest stateInterp context } (x::\text{ins}) \text{ s outs})
\end{aligned}$$

[trType\_distinct\_clauses]

$$\begin{aligned}
& \vdash (\forall a' \ a. \text{ discard } a \neq \text{trap } a') \wedge (\forall a' \ a. \text{ discard } a \neq \text{exec } a') \wedge \\
& \quad \forall a' \ a. \text{ trap } a \neq \text{exec } a'
\end{aligned}$$

[trType\_one\_one]

$$\begin{aligned} \vdash (\forall a \ a'. (\text{discard } a = \text{discard } a') \iff (a = a')) \wedge \\ (\forall a \ a'. (\text{trap } a = \text{trap } a') \iff (a = a')) \wedge \\ \forall a \ a'. (\text{exec } a = \text{exec } a') \iff (a = a') \end{aligned}$$

## 4 satList Theory

**Built:** 10 June 2018

**Parent Theories:** aclDrules

### 4.1 Definitions

[satList\_def]

$$\begin{aligned} \vdash \forall M \ Oi \ Os \ formList. \\ (M, Oi, Os) \text{ satList } formList \iff \\ \text{FOLDR } (\lambda x \ y. x \wedge y) \ \text{T} \ (\text{MAP } (\lambda f. (M, Oi, Os) \text{ sat } f) \ formList) \end{aligned}$$

### 4.2 Theorems

[satList\_conj]

$$\begin{aligned} \vdash \forall l_1 \ l_2 \ M \ Oi \ Os. \\ (M, Oi, Os) \text{ satList } l_1 \wedge (M, Oi, Os) \text{ satList } l_2 \iff \\ (M, Oi, Os) \text{ satList } (l_1 ++ l_2) \end{aligned}$$

[satList\_CONS]

$$\begin{aligned} \vdash \forall h \ t \ M \ Oi \ Os. \\ (M, Oi, Os) \text{ satList } (h :: t) \iff \\ (M, Oi, Os) \text{ sat } h \wedge (M, Oi, Os) \text{ satList } t \end{aligned}$$

[satList\_nil]

$$\vdash (M, Oi, Os) \text{ satList } []$$

## 5 PBTypeIntegrated Theory

**Built:** 10 June 2018

**Parent Theories:** OMNIType

### 5.1 Datatypes

```
omniCommand = ssmPlanPBComplete | ssmMoveToORPComplete
              | ssmConductORPComplete | ssmMoveToPBComplete
              | ssmConductPBComplete | invalidOmniCommand
```

```
plCommand = crossLD | conductORP | moveToPB | conductPB
            | completePB | incomplete
```

$$slCommand = PL \text{ PTypeIntegrated\$plCommand} \mid OMNI \text{ omniCommand}$$

$$slOutput = PlanPB \mid MoveToORP \mid ConductORP \mid MoveToPB \\ \mid ConductPB \mid CompletePB \mid unAuthenticated \\ \mid unAuthorized$$

$$slState = PLAN\_PB \mid MOVE\_TO\_ORP \mid CONDUCT\_ORP \mid MOVE\_TO\_PB \\ \mid CONDUCT\_PB \mid COMPLETE\_PB$$

$$stateRole = PlatoonLeader \mid Omni$$

## 5.2 Theorems

[omniCommand\_distinct\_clauses]

$$\begin{aligned} \vdash & \text{ssmPlanPBComplete} \neq \text{ssmMoveToORPComplete} \wedge \\ & \text{ssmPlanPBComplete} \neq \text{ssmConductORPComplete} \wedge \\ & \text{ssmPlanPBComplete} \neq \text{ssmMoveToPBComplete} \wedge \\ & \text{ssmPlanPBComplete} \neq \text{ssmConductPBComplete} \wedge \\ & \text{ssmPlanPBComplete} \neq \text{invalidOmniCommand} \wedge \\ & \text{ssmMoveToORPComplete} \neq \text{ssmConductORPComplete} \wedge \\ & \text{ssmMoveToORPComplete} \neq \text{ssmMoveToPBComplete} \wedge \\ & \text{ssmMoveToORPComplete} \neq \text{ssmConductPBComplete} \wedge \\ & \text{ssmMoveToORPComplete} \neq \text{invalidOmniCommand} \wedge \\ & \text{ssmConductORPComplete} \neq \text{ssmMoveToPBComplete} \wedge \\ & \text{ssmConductORPComplete} \neq \text{ssmConductPBComplete} \wedge \\ & \text{ssmConductORPComplete} \neq \text{invalidOmniCommand} \wedge \\ & \text{ssmMoveToPBComplete} \neq \text{ssmConductPBComplete} \wedge \\ & \text{ssmMoveToPBComplete} \neq \text{invalidOmniCommand} \wedge \\ & \text{ssmConductPBComplete} \neq \text{invalidOmniCommand} \end{aligned}$$

[plCommand\_distinct\_clauses]

$$\begin{aligned} \vdash & \text{crossLD} \neq \text{conductORP} \wedge \text{crossLD} \neq \text{moveToPB} \wedge \\ & \text{crossLD} \neq \text{conductPB} \wedge \text{crossLD} \neq \text{completePB} \wedge \\ & \text{crossLD} \neq \text{incomplete} \wedge \text{conductORP} \neq \text{moveToPB} \wedge \\ & \text{conductORP} \neq \text{conductPB} \wedge \text{conductORP} \neq \text{completePB} \wedge \\ & \text{conductORP} \neq \text{incomplete} \wedge \text{moveToPB} \neq \text{conductPB} \wedge \\ & \text{moveToPB} \neq \text{completePB} \wedge \text{moveToPB} \neq \text{incomplete} \wedge \\ & \text{conductPB} \neq \text{completePB} \wedge \text{conductPB} \neq \text{incomplete} \wedge \\ & \text{completePB} \neq \text{incomplete} \end{aligned}$$

[slCommand\_distinct\_clauses]

$$\vdash \forall a' \ a. \text{PL } a \neq \text{OMNI } a'$$

[slCommand\_one\_one]

$$\begin{aligned} \vdash & (\forall a \ a'. (\text{PL } a = \text{PL } a') \iff (a = a')) \wedge \\ & \forall a \ a'. (\text{OMNI } a = \text{OMNI } a') \iff (a = a') \end{aligned}$$

[slOutput\_distinct\_clauses]

$\vdash$  PlanPB  $\neq$  MoveToORP  $\wedge$  PlanPB  $\neq$  ConductORP  $\wedge$   
 PlanPB  $\neq$  MoveToPB  $\wedge$  PlanPB  $\neq$  ConductPB  $\wedge$   
 PlanPB  $\neq$  CompletePB  $\wedge$  PlanPB  $\neq$  unAuthenticated  $\wedge$   
 PlanPB  $\neq$  unAuthorized  $\wedge$  MoveToORP  $\neq$  ConductORP  $\wedge$   
 MoveToORP  $\neq$  MoveToPB  $\wedge$  MoveToORP  $\neq$  ConductPB  $\wedge$   
 MoveToORP  $\neq$  CompletePB  $\wedge$  MoveToORP  $\neq$  unAuthenticated  $\wedge$   
 MoveToORP  $\neq$  unAuthorized  $\wedge$  ConductORP  $\neq$  MoveToPB  $\wedge$   
 ConductORP  $\neq$  ConductPB  $\wedge$  ConductORP  $\neq$  CompletePB  $\wedge$   
 ConductORP  $\neq$  unAuthenticated  $\wedge$  ConductORP  $\neq$  unAuthorized  $\wedge$   
 MoveToPB  $\neq$  ConductPB  $\wedge$  MoveToPB  $\neq$  CompletePB  $\wedge$   
 MoveToPB  $\neq$  unAuthenticated  $\wedge$  MoveToPB  $\neq$  unAuthorized  $\wedge$   
 ConductPB  $\neq$  CompletePB  $\wedge$  ConductPB  $\neq$  unAuthenticated  $\wedge$   
 ConductPB  $\neq$  unAuthorized  $\wedge$  CompletePB  $\neq$  unAuthenticated  $\wedge$   
 CompletePB  $\neq$  unAuthorized  $\wedge$  unAuthenticated  $\neq$  unAuthorized

[slState\_distinct\_clauses]

$\vdash$  PLAN\_PB  $\neq$  MOVE\_TO\_ORP  $\wedge$  PLAN\_PB  $\neq$  CONDUCT\_ORP  $\wedge$   
 PLAN\_PB  $\neq$  MOVE\_TO\_PB  $\wedge$  PLAN\_PB  $\neq$  CONDUCT\_PB  $\wedge$   
 PLAN\_PB  $\neq$  COMPLETE\_PB  $\wedge$  MOVE\_TO\_ORP  $\neq$  CONDUCT\_ORP  $\wedge$   
 MOVE\_TO\_ORP  $\neq$  MOVE\_TO\_PB  $\wedge$  MOVE\_TO\_ORP  $\neq$  CONDUCT\_PB  $\wedge$   
 MOVE\_TO\_ORP  $\neq$  COMPLETE\_PB  $\wedge$  CONDUCT\_ORP  $\neq$  MOVE\_TO\_PB  $\wedge$   
 CONDUCT\_ORP  $\neq$  CONDUCT\_PB  $\wedge$  CONDUCT\_ORP  $\neq$  COMPLETE\_PB  $\wedge$   
 MOVE\_TO\_PB  $\neq$  CONDUCT\_PB  $\wedge$  MOVE\_TO\_PB  $\neq$  COMPLETE\_PB  $\wedge$   
 CONDUCT\_PB  $\neq$  COMPLETE\_PB

[stateRole\_distinct\_clauses]

$\vdash$  PlatoonLeader  $\neq$  Omni

## 6 PBIntegratedDef Theory

**Built:** 10 June 2018

**Parent Theories:** PBTypeIntegrated, aclfoundation

### 6.1 Definitions

[secAuthorization\_def]

$\vdash \forall xs. \text{secAuthorization } xs = \text{secHelper } (\text{getOmniCommand } xs)$

[secHelper\_def]

$\vdash \forall cmd.$   
 $\text{secHelper } cmd =$   
 $[\text{Name Omni controls prop (SOME (SLc (OMNI } cmd)))]$

## 6.2 Theorems

[getOmniCommand\_def]

$$\begin{aligned}
& \vdash (\text{getOmniCommand } [] = \text{invalidOmniCommand}) \wedge \\
& (\forall xs \text{ cmd.} \\
& \quad \text{getOmniCommand} \\
& \quad \quad (\text{Name Omni controls prop (SOME (SLc (OMNI cmd)))::xs}) = \\
& \quad \quad \text{cmd}) \wedge \\
& (\forall xs. \text{getOmniCommand (TT::xs)} = \text{getOmniCommand xs}) \wedge \\
& (\forall xs. \text{getOmniCommand (FF::xs)} = \text{getOmniCommand xs}) \wedge \\
& (\forall xs \ v_2. \text{getOmniCommand (prop } v_2::xs) = \text{getOmniCommand xs}) \wedge \\
& (\forall xs \ v_3. \text{getOmniCommand (notf } v_3::xs) = \text{getOmniCommand xs}) \wedge \\
& (\forall xs \ v_5 \ v_4. \\
& \quad \text{getOmniCommand (v}_4 \text{ andf } v_5::xs) = \text{getOmniCommand xs}) \wedge \\
& (\forall xs \ v_7 \ v_6. \\
& \quad \text{getOmniCommand (v}_6 \text{ orf } v_7::xs) = \text{getOmniCommand xs}) \wedge \\
& (\forall xs \ v_9 \ v_8. \\
& \quad \text{getOmniCommand (v}_8 \text{ impf } v_9::xs) = \text{getOmniCommand xs}) \wedge \\
& (\forall xs \ v_{11} \ v_{10}. \\
& \quad \text{getOmniCommand (v}_{10} \text{ eqf } v_{11}::xs) = \text{getOmniCommand xs}) \wedge \\
& (\forall xs \ v_{13} \ v_{12}. \\
& \quad \text{getOmniCommand (v}_{12} \text{ says } v_{13}::xs) = \text{getOmniCommand xs}) \wedge \\
& (\forall xs \ v_{15} \ v_{14}. \\
& \quad \text{getOmniCommand (v}_{14} \text{ speaks_for } v_{15}::xs) = \\
& \quad \text{getOmniCommand xs}) \wedge \\
& (\forall xs \ v_{16}. \\
& \quad \text{getOmniCommand (v}_{16} \text{ controls TT::xs)} = \\
& \quad \text{getOmniCommand xs}) \wedge \\
& (\forall xs \ v_{16}. \\
& \quad \text{getOmniCommand (v}_{16} \text{ controls FF::xs)} = \\
& \quad \text{getOmniCommand xs}) \wedge \\
& (\forall xs \ v_{134}. \\
& \quad \text{getOmniCommand (Name v}_{134} \text{ controls prop NONE::xs)} = \\
& \quad \text{getOmniCommand xs}) \wedge \\
& (\forall xs \ v_{144}. \\
& \quad \text{getOmniCommand} \\
& \quad \quad (\text{Name PlatoonLeader controls prop (SOME v}_{144}\text{)::xs}) = \\
& \quad \quad \text{getOmniCommand xs}) \wedge \\
& (\forall xs \ v_{146}. \\
& \quad \text{getOmniCommand} \\
& \quad \quad (\text{Name Omni controls prop (SOME (ESCc v}_{146}\text{)::xs}) = \\
& \quad \quad \text{getOmniCommand xs}) \wedge \\
& (\forall xs \ v_{150}. \\
& \quad \text{getOmniCommand} \\
& \quad \quad (\text{Name Omni controls prop (SOME (SLc (PL v}_{150}\text{)))::xs}) = \\
& \quad \quad \text{getOmniCommand xs}) \wedge \\
& (\forall xs \ v_{68} \ v_{136} \ v_{135}. \\
& \quad \text{getOmniCommand (v}_{135} \text{ meet } v_{136} \text{ controls prop } v_{68}::xs) = \\
& \quad \text{getOmniCommand xs}) \wedge
\end{aligned}$$



$(\forall xs \ v_{68} \ v_{138} \ v_{137}.$   
 $\quad \text{getOmniCommand } (v_{137} \text{ quoting } v_{138} \text{ controls prop } v_{68}::xs) =$   
 $\quad \text{getOmniCommand } xs) \wedge$   
 $(\forall xs \ v_{69} \ v_{16}.$   
 $\quad \text{getOmniCommand } (v_{16} \text{ controls notf } v_{69}::xs) =$   
 $\quad \text{getOmniCommand } xs) \wedge$   
 $(\forall xs \ v_{71} \ v_{70} \ v_{16}.$   
 $\quad \text{getOmniCommand } (v_{16} \text{ controls } (v_{70} \text{ andf } v_{71})::xs) =$   
 $\quad \text{getOmniCommand } xs) \wedge$   
 $(\forall xs \ v_{73} \ v_{72} \ v_{16}.$   
 $\quad \text{getOmniCommand } (v_{16} \text{ controls } (v_{72} \text{ orf } v_{73})::xs) =$   
 $\quad \text{getOmniCommand } xs) \wedge$   
 $(\forall xs \ v_{75} \ v_{74} \ v_{16}.$   
 $\quad \text{getOmniCommand } (v_{16} \text{ controls } (v_{74} \text{ impf } v_{75})::xs) =$   
 $\quad \text{getOmniCommand } xs) \wedge$   
 $(\forall xs \ v_{77} \ v_{76} \ v_{16}.$   
 $\quad \text{getOmniCommand } (v_{16} \text{ controls } (v_{76} \text{ eqf } v_{77})::xs) =$   
 $\quad \text{getOmniCommand } xs) \wedge$   
 $(\forall xs \ v_{79} \ v_{78} \ v_{16}.$   
 $\quad \text{getOmniCommand } (v_{16} \text{ controls } v_{78} \text{ says } v_{79}::xs) =$   
 $\quad \text{getOmniCommand } xs) \wedge$   
 $(\forall xs \ v_{81} \ v_{80} \ v_{16}.$   
 $\quad \text{getOmniCommand } (v_{16} \text{ controls } v_{80} \text{ speaks\_for } v_{81}::xs) =$   
 $\quad \text{getOmniCommand } xs) \wedge$   
 $(\forall xs \ v_{83} \ v_{82} \ v_{16}.$   
 $\quad \text{getOmniCommand } (v_{16} \text{ controls } v_{82} \text{ controls } v_{83}::xs) =$   
 $\quad \text{getOmniCommand } xs) \wedge$   
 $(\forall xs \ v_{86} \ v_{85} \ v_{84} \ v_{16}.$   
 $\quad \text{getOmniCommand } (v_{16} \text{ controls reps } v_{84} \ v_{85} \ v_{86}::xs) =$   
 $\quad \text{getOmniCommand } xs) \wedge$   
 $(\forall xs \ v_{88} \ v_{87} \ v_{16}.$   
 $\quad \text{getOmniCommand } (v_{16} \text{ controls } v_{87} \text{ domi } v_{88}::xs) =$   
 $\quad \text{getOmniCommand } xs) \wedge$   
 $(\forall xs \ v_{90} \ v_{89} \ v_{16}.$   
 $\quad \text{getOmniCommand } (v_{16} \text{ controls } v_{89} \text{ eqi } v_{90}::xs) =$   
 $\quad \text{getOmniCommand } xs) \wedge$   
 $(\forall xs \ v_{92} \ v_{91} \ v_{16}.$   
 $\quad \text{getOmniCommand } (v_{16} \text{ controls } v_{91} \text{ doms } v_{92}::xs) =$   
 $\quad \text{getOmniCommand } xs) \wedge$   
 $(\forall xs \ v_{94} \ v_{93} \ v_{16}.$   
 $\quad \text{getOmniCommand } (v_{16} \text{ controls } v_{93} \text{ eqs } v_{94}::xs) =$   
 $\quad \text{getOmniCommand } xs) \wedge$   
 $(\forall xs \ v_{96} \ v_{95} \ v_{16}.$   
 $\quad \text{getOmniCommand } (v_{16} \text{ controls } v_{95} \text{ eqn } v_{96}::xs) =$   
 $\quad \text{getOmniCommand } xs) \wedge$   
 $(\forall xs \ v_{98} \ v_{97} \ v_{16}.$   
 $\quad \text{getOmniCommand } (v_{16} \text{ controls } v_{97} \text{ lte } v_{98}::xs) =$   
 $\quad \text{getOmniCommand } xs) \wedge$   
 $(\forall xs \ v_{99} \ v_{16} \ v_{100}.$

```

    getOmniCommand (v16 controls v99 lt v100::xs) =
    getOmniCommand xs) ∧
  (∀ xs v20 v19 v18.
    getOmniCommand (reps v18 v19 v20::xs) =
    getOmniCommand xs) ∧
  (∀ xs v22 v21.
    getOmniCommand (v21 domi v22::xs) = getOmniCommand xs) ∧
  (∀ xs v24 v23.
    getOmniCommand (v23 eqi v24::xs) = getOmniCommand xs) ∧
  (∀ xs v26 v25.
    getOmniCommand (v25 doms v26::xs) = getOmniCommand xs) ∧
  (∀ xs v28 v27.
    getOmniCommand (v27 eqs v28::xs) = getOmniCommand xs) ∧
  (∀ xs v30 v29.
    getOmniCommand (v29 eqn v30::xs) = getOmniCommand xs) ∧
  (∀ xs v32 v31.
    getOmniCommand (v31 lte v32::xs) = getOmniCommand xs) ∧
  ∀ xs v34 v33.
    getOmniCommand (v33 lt v34::xs) = getOmniCommand xs

```

[getOmniCommand\_ind]

```

⊢ ∀ P.
  P [] ∧
  (∀ cmd xs.
    P
      (Name Omni controls prop (SOME (SLc (OMNI cmd))))::
      xs)) ∧ (∀ xs. P xs ⇒ P (TT::xs)) ∧
  (∀ xs. P xs ⇒ P (FF::xs)) ∧
  (∀ v2 xs. P xs ⇒ P (prop v2::xs)) ∧
  (∀ v3 xs. P xs ⇒ P (notf v3::xs)) ∧
  (∀ v4 v5 xs. P xs ⇒ P (v4 andf v5::xs)) ∧
  (∀ v6 v7 xs. P xs ⇒ P (v6 orf v7::xs)) ∧
  (∀ v8 v9 xs. P xs ⇒ P (v8 impf v9::xs)) ∧
  (∀ v10 v11 xs. P xs ⇒ P (v10 eqf v11::xs)) ∧
  (∀ v12 v13 xs. P xs ⇒ P (v12 says v13::xs)) ∧
  (∀ v14 v15 xs. P xs ⇒ P (v14 speaks_for v15::xs)) ∧
  (∀ v16 xs. P xs ⇒ P (v16 controls TT::xs)) ∧
  (∀ v16 xs. P xs ⇒ P (v16 controls FF::xs)) ∧
  (∀ v134 xs. P xs ⇒ P (Name v134 controls prop NONE::xs)) ∧
  (∀ v144 xs.
    P xs ⇒
    P (Name PlatoonLeader controls prop (SOME v144)::xs)) ∧
  (∀ v146 xs.
    P xs ⇒
    P (Name Omni controls prop (SOME (ESCc v146))::xs)) ∧
  (∀ v150 xs.
    P xs ⇒
    P
      (Name Omni controls prop (SOME (SLc (PL v150))))::

```

$$\begin{aligned}
& xs)) \wedge \\
& (\forall v135 \ v136 \ v68 \ xs. \\
& \quad P \ xs \Rightarrow P \ (v135 \text{ meet } v136 \text{ controls prop } v68::xs)) \wedge \\
& (\forall v137 \ v138 \ v68 \ xs. \\
& \quad P \ xs \Rightarrow P \ (v137 \text{ quoting } v138 \text{ controls prop } v68::xs)) \wedge \\
& (\forall v16 \ v69 \ xs. \ P \ xs \Rightarrow P \ (v16 \text{ controls notf } v69::xs)) \wedge \\
& (\forall v16 \ v70 \ v71 \ xs. \\
& \quad P \ xs \Rightarrow P \ (v16 \text{ controls } (v70 \text{ andf } v71)::xs)) \wedge \\
& (\forall v16 \ v72 \ v73 \ xs. \\
& \quad P \ xs \Rightarrow P \ (v16 \text{ controls } (v72 \text{ orf } v73)::xs)) \wedge \\
& (\forall v16 \ v74 \ v75 \ xs. \\
& \quad P \ xs \Rightarrow P \ (v16 \text{ controls } (v74 \text{ impf } v75)::xs)) \wedge \\
& (\forall v16 \ v76 \ v77 \ xs. \\
& \quad P \ xs \Rightarrow P \ (v16 \text{ controls } (v76 \text{ eqf } v77)::xs)) \wedge \\
& (\forall v16 \ v78 \ v79 \ xs. \\
& \quad P \ xs \Rightarrow P \ (v16 \text{ controls } v78 \text{ says } v79::xs)) \wedge \\
& (\forall v16 \ v80 \ v81 \ xs. \\
& \quad P \ xs \Rightarrow P \ (v16 \text{ controls } v80 \text{ speaks\_for } v81::xs)) \wedge \\
& (\forall v16 \ v82 \ v83 \ xs. \\
& \quad P \ xs \Rightarrow P \ (v16 \text{ controls } v82 \text{ controls } v83::xs)) \wedge \\
& (\forall v16 \ v84 \ v85 \ v86 \ xs. \\
& \quad P \ xs \Rightarrow P \ (v16 \text{ controls reps } v84 \ v85 \ v86::xs)) \wedge \\
& (\forall v16 \ v87 \ v88 \ xs. \\
& \quad P \ xs \Rightarrow P \ (v16 \text{ controls } v87 \text{ domi } v88::xs)) \wedge \\
& (\forall v16 \ v89 \ v90 \ xs. \\
& \quad P \ xs \Rightarrow P \ (v16 \text{ controls } v89 \text{ eqi } v90::xs)) \wedge \\
& (\forall v16 \ v91 \ v92 \ xs. \\
& \quad P \ xs \Rightarrow P \ (v16 \text{ controls } v91 \text{ doms } v92::xs)) \wedge \\
& (\forall v16 \ v93 \ v94 \ xs. \\
& \quad P \ xs \Rightarrow P \ (v16 \text{ controls } v93 \text{ eqs } v94::xs)) \wedge \\
& (\forall v16 \ v95 \ v96 \ xs. \\
& \quad P \ xs \Rightarrow P \ (v16 \text{ controls } v95 \text{ eqn } v96::xs)) \wedge \\
& (\forall v16 \ v97 \ v98 \ xs. \\
& \quad P \ xs \Rightarrow P \ (v16 \text{ controls } v97 \text{ lte } v98::xs)) \wedge \\
& (\forall v16 \ v99 \ v100 \ xs. \\
& \quad P \ xs \Rightarrow P \ (v16 \text{ controls } v99 \text{ lt } v100::xs)) \wedge \\
& (\forall v18 \ v19 \ v20 \ xs. \ P \ xs \Rightarrow P \ (\text{reps } v18 \ v19 \ v20::xs)) \wedge \\
& (\forall v21 \ v22 \ xs. \ P \ xs \Rightarrow P \ (v21 \text{ domi } v22::xs)) \wedge \\
& (\forall v23 \ v24 \ xs. \ P \ xs \Rightarrow P \ (v23 \text{ eqi } v24::xs)) \wedge \\
& (\forall v25 \ v26 \ xs. \ P \ xs \Rightarrow P \ (v25 \text{ doms } v26::xs)) \wedge \\
& (\forall v27 \ v28 \ xs. \ P \ xs \Rightarrow P \ (v27 \text{ eqs } v28::xs)) \wedge \\
& (\forall v29 \ v30 \ xs. \ P \ xs \Rightarrow P \ (v29 \text{ eqn } v30::xs)) \wedge \\
& (\forall v31 \ v32 \ xs. \ P \ xs \Rightarrow P \ (v31 \text{ lte } v32::xs)) \wedge \\
& (\forall v33 \ v34 \ xs. \ P \ xs \Rightarrow P \ (v33 \text{ lt } v34::xs)) \Rightarrow \\
& \forall v. \ P \ v
\end{aligned}$$

[secContext\_def]

$$\begin{aligned}
& \vdash (\text{secContext PLAN\_PB } (x::xs) = \\
& \quad [\text{prop } (\text{SOME } (\text{SLc } (\text{OMNI ssmPlanPBComplete})))] \text{ impf}
\end{aligned}$$

```

    Name PlatoonLeader controls
    prop (SOME (SLc (PL crossLD))))))  $\wedge$ 
(secContext MOVE_TO_ORP ( $x::xs$ ) =
  [prop (SOME (SLc (OMNI ssmMoveToORPComplete))) impf
    Name PlatoonLeader controls
    prop (SOME (SLc (PL conductORP))))))  $\wedge$ 
(secContext CONDUCT_ORP ( $x::xs$ ) =
  [prop (SOME (SLc (OMNI ssmConductORPComplete))) impf
    Name PlatoonLeader controls
    prop (SOME (SLc (PL moveToPB))))))  $\wedge$ 
(secContext MOVE_TO_PB ( $x::xs$ ) =
  [prop (SOME (SLc (OMNI ssmMoveToPBComplete))) impf
    Name PlatoonLeader controls
    prop (SOME (SLc (PL conductPB))))))  $\wedge$ 
(secContext CONDUCT_PB ( $x::xs$ ) =
  [prop (SOME (SLc (OMNI ssmConductPBComplete))) impf
    Name PlatoonLeader controls
    prop (SOME (SLc (PL completePB))))))
[secContext_ind]
 $\vdash \forall P.$ 
  ( $\forall x\ xs. P\ \text{PLAN\_PB}\ (x::xs)$ )  $\wedge$ 
  ( $\forall x\ xs. P\ \text{MOVE\_TO\_ORP}\ (x::xs)$ )  $\wedge$ 
  ( $\forall x\ xs. P\ \text{CONDUCT\_ORP}\ (x::xs)$ )  $\wedge$ 
  ( $\forall x\ xs. P\ \text{MOVE\_TO\_PB}\ (x::xs)$ )  $\wedge$ 
  ( $\forall x\ xs. P\ \text{CONDUCT\_PB}\ (x::xs)$ )  $\wedge$  ( $\forall v_4. P\ v_4\ []$ )  $\wedge$ 
  ( $\forall v_5\ v_6. P\ \text{COMPLETE\_PB}\ (v_5::v_6)$ )  $\Rightarrow$ 
   $\forall v\ v_1. P\ v\ v_1$ 

```

## 7 ssmConductORP Theory

**Built:** 10 June 2018

**Parent Theories:** ConductORPType, ssm11, OMNIType

### 7.1 Definitions

```

[secContextConductORP_def]
 $\vdash \forall plc\ cmd\ psg\ cmd\ incomplete.$ 
  secContextConductORP  $plc\ cmd\ psg\ cmd\ incomplete$  =
  [Name PlatoonLeader controls prop (SOME (SLc (PL  $plc\ cmd$ ))));
    Name PlatoonSergeant controls
    prop (SOME (SLc (PSG  $psg\ cmd$ )));
    Name PlatoonLeader says
    prop (SOME (SLc (PSG  $psg\ cmd$ ))) impf prop NONE;
    Name PlatoonSergeant says
    prop (SOME (SLc (PL  $plc\ cmd$ ))) impf prop NONE]
[ssmConductORPStateInterp_def]
 $\vdash \forall slState. \text{ssmConductORPStateInterp}\ slState = \text{TT}$ 

```

## 7.2 Theorems

[authTestConductORP\_cmd\_reject\_lemma]

$\vdash \forall cmd. \neg \text{authTestConductORP} (\text{prop } (\text{SOME } cmd))$

[authTestConductORP\_def]

$\vdash (\text{authTestConductORP} (\text{Name PlatoonLeader says prop } cmd) \iff$   
 $T) \wedge$   
 $(\text{authTestConductORP} (\text{Name PlatoonSergeant says prop } cmd) \iff$   
 $T) \wedge (\text{authTestConductORP } TT \iff F) \wedge$   
 $(\text{authTestConductORP } FF \iff F) \wedge$   
 $(\text{authTestConductORP} (\text{prop } v) \iff F) \wedge$   
 $(\text{authTestConductORP} (\text{notf } v_1) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_2 \text{ andf } v_3) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_4 \text{ orf } v_5) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_6 \text{ impf } v_7) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_8 \text{ eqf } v_9) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says } TT) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says } FF) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{133} \text{ meet } v_{134} \text{ says prop } v_{66}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{135} \text{ quoting } v_{136} \text{ says prop } v_{66}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says notf } v_{67}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says } (v_{68} \text{ andf } v_{69})) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says } (v_{70} \text{ orf } v_{71})) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says } (v_{72} \text{ impf } v_{73})) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says } (v_{74} \text{ eqf } v_{75})) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says } v_{76} \text{ says } v_{77}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says } v_{78} \text{ speaks\_for } v_{79}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says } v_{80} \text{ controls } v_{81}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says reps } v_{82} \ v_{83} \ v_{84}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says } v_{85} \text{ domi } v_{86}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says } v_{87} \text{ eqi } v_{88}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says } v_{89} \text{ doms } v_{90}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says } v_{91} \text{ eqs } v_{92}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says } v_{93} \text{ eqn } v_{94}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says } v_{95} \text{ lte } v_{96}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{10} \text{ says } v_{97} \text{ lt } v_{98}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{12} \text{ speaks\_for } v_{13}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{14} \text{ controls } v_{15}) \iff F) \wedge$   
 $(\text{authTestConductORP} (\text{reps } v_{16} \ v_{17} \ v_{18}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{19} \text{ domi } v_{20}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{21} \text{ eqi } v_{22}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{23} \text{ doms } v_{24}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{25} \text{ eqs } v_{26}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{27} \text{ eqn } v_{28}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{29} \text{ lte } v_{30}) \iff F) \wedge$   
 $(\text{authTestConductORP} (v_{31} \text{ lt } v_{32}) \iff F)$

[authTestConductORP\_ind]

$\vdash \forall P.$   
 $(\forall cmd. P (Name PlatoonLeader says prop cmd)) \wedge$   
 $(\forall cmd. P (Name PlatoonSergeant says prop cmd)) \wedge P \text{ TT} \wedge$   
 $P \text{ FF} \wedge (\forall v. P (prop v)) \wedge (\forall v_1. P (notf v_1)) \wedge$   
 $(\forall v_2 v_3. P (v_2 \text{ andf } v_3)) \wedge (\forall v_4 v_5. P (v_4 \text{ orf } v_5)) \wedge$   
 $(\forall v_6 v_7. P (v_6 \text{ impf } v_7)) \wedge (\forall v_8 v_9. P (v_8 \text{ eqf } v_9)) \wedge$   
 $(\forall v_{10}. P (v_{10} \text{ says TT})) \wedge (\forall v_{10}. P (v_{10} \text{ says FF})) \wedge$   
 $(\forall v_{133} v_{134} v_{66}. P (v_{133} \text{ meet } v_{134} \text{ says prop } v_{66})) \wedge$   
 $(\forall v_{135} v_{136} v_{66}. P (v_{135} \text{ quoting } v_{136} \text{ says prop } v_{66})) \wedge$   
 $(\forall v_{10} v_{67}. P (v_{10} \text{ says notf } v_{67})) \wedge$   
 $(\forall v_{10} v_{68} v_{69}. P (v_{10} \text{ says } (v_{68} \text{ andf } v_{69}))) \wedge$   
 $(\forall v_{10} v_{70} v_{71}. P (v_{10} \text{ says } (v_{70} \text{ orf } v_{71}))) \wedge$   
 $(\forall v_{10} v_{72} v_{73}. P (v_{10} \text{ says } (v_{72} \text{ impf } v_{73}))) \wedge$   
 $(\forall v_{10} v_{74} v_{75}. P (v_{10} \text{ says } (v_{74} \text{ eqf } v_{75}))) \wedge$   
 $(\forall v_{10} v_{76} v_{77}. P (v_{10} \text{ says } v_{76} \text{ says } v_{77})) \wedge$   
 $(\forall v_{10} v_{78} v_{79}. P (v_{10} \text{ says } v_{78} \text{ speaks\_for } v_{79})) \wedge$   
 $(\forall v_{10} v_{80} v_{81}. P (v_{10} \text{ says } v_{80} \text{ controls } v_{81})) \wedge$   
 $(\forall v_{10} v_{82} v_{83} v_{84}. P (v_{10} \text{ says reps } v_{82} v_{83} v_{84})) \wedge$   
 $(\forall v_{10} v_{85} v_{86}. P (v_{10} \text{ says } v_{85} \text{ domi } v_{86})) \wedge$   
 $(\forall v_{10} v_{87} v_{88}. P (v_{10} \text{ says } v_{87} \text{ eqi } v_{88})) \wedge$   
 $(\forall v_{10} v_{89} v_{90}. P (v_{10} \text{ says } v_{89} \text{ doms } v_{90})) \wedge$   
 $(\forall v_{10} v_{91} v_{92}. P (v_{10} \text{ says } v_{91} \text{ eqs } v_{92})) \wedge$   
 $(\forall v_{10} v_{93} v_{94}. P (v_{10} \text{ says } v_{93} \text{ eqn } v_{94})) \wedge$   
 $(\forall v_{10} v_{95} v_{96}. P (v_{10} \text{ says } v_{95} \text{ lte } v_{96})) \wedge$   
 $(\forall v_{10} v_{97} v_{98}. P (v_{10} \text{ says } v_{97} \text{ lt } v_{98})) \wedge$   
 $(\forall v_{12} v_{13}. P (v_{12} \text{ speaks\_for } v_{13})) \wedge$   
 $(\forall v_{14} v_{15}. P (v_{14} \text{ controls } v_{15})) \wedge$   
 $(\forall v_{16} v_{17} v_{18}. P (\text{reps } v_{16} v_{17} v_{18})) \wedge$   
 $(\forall v_{19} v_{20}. P (v_{19} \text{ domi } v_{20})) \wedge$   
 $(\forall v_{21} v_{22}. P (v_{21} \text{ eqi } v_{22})) \wedge$   
 $(\forall v_{23} v_{24}. P (v_{23} \text{ doms } v_{24})) \wedge$   
 $(\forall v_{25} v_{26}. P (v_{25} \text{ eqs } v_{26})) \wedge (\forall v_{27} v_{28}. P (v_{27} \text{ eqn } v_{28})) \wedge$   
 $(\forall v_{29} v_{30}. P (v_{29} \text{ lte } v_{30})) \wedge (\forall v_{31} v_{32}. P (v_{31} \text{ lt } v_{32})) \Rightarrow$   
 $\forall v. P v$

[conductORPNS\_def]

$\vdash (\text{conductORPNS CONDUCT\_ORP (exec (PL secure))} = \text{SECURE}) \wedge$   
 $(\text{conductORPNS CONDUCT\_ORP (exec (PL plIncomplete))} =$   
 $\text{CONDUCT\_ORP}) \wedge$   
 $(\text{conductORPNS SECURE (exec (PSG actionsIn))} = \text{ACTIONS\_IN}) \wedge$   
 $(\text{conductORPNS SECURE (exec (PSG psgIncomplete))} = \text{SECURE}) \wedge$   
 $(\text{conductORPNS ACTIONS\_IN (exec (PL withdraw))} = \text{WITHDRAW}) \wedge$   
 $(\text{conductORPNS ACTIONS\_IN (exec (PL plIncomplete))} =$   
 $\text{ACTIONS\_IN}) \wedge$   
 $(\text{conductORPNS WITHDRAW (exec (PL complete))} = \text{COMPLETE}) \wedge$   
 $(\text{conductORPNS WITHDRAW (exec (PL plIncomplete))} = \text{WITHDRAW}) \wedge$   
 $(\text{conductORPNS } s (\text{trap (PL cmd')}) = s) \wedge$   
 $(\text{conductORPNS } s (\text{trap (PSG cmd)}) = s) \wedge$

(conductORPNS  $s$  (discard (PL  $cmd'$ )) =  $s$ )  $\wedge$   
 (conductORPNS  $s$  (discard (PSG  $cmd$ )) =  $s$ )

[conductORPNS\_ind]

$\vdash \forall P.$   
 $P$  CONDUCT\_ORP (exec (PL secure))  $\wedge$   
 $P$  CONDUCT\_ORP (exec (PL plIncomplete))  $\wedge$   
 $P$  SECURE (exec (PSG actionsIn))  $\wedge$   
 $P$  SECURE (exec (PSG psgIncomplete))  $\wedge$   
 $P$  ACTIONS\_IN (exec (PL withdraw))  $\wedge$   
 $P$  ACTIONS\_IN (exec (PL plIncomplete))  $\wedge$   
 $P$  WITHDRAW (exec (PL complete))  $\wedge$   
 $P$  WITHDRAW (exec (PL plIncomplete))  $\wedge$   
 $(\forall s \text{ cmd}. P \ s \ (\text{trap} \ (\text{PL} \ cmd))) \wedge$   
 $(\forall s \text{ cmd}. P \ s \ (\text{trap} \ (\text{PSG} \ cmd))) \wedge$   
 $(\forall s \text{ cmd}. P \ s \ (\text{discard} \ (\text{PL} \ cmd))) \wedge$   
 $(\forall s \text{ cmd}. P \ s \ (\text{discard} \ (\text{PSG} \ cmd))) \wedge$   
 $P$  CONDUCT\_ORP (exec (PL withdraw))  $\wedge$   
 $P$  CONDUCT\_ORP (exec (PL complete))  $\wedge$   
 $(\forall v_{11}. P \ \text{CONDUCT\_ORP} \ (\text{exec} \ (\text{PSG} \ v_{11}))) \wedge$   
 $(\forall v_{13}. P \ \text{SECURE} \ (\text{exec} \ (\text{PL} \ v_{13}))) \wedge$   
 $P$  ACTIONS\_IN (exec (PL secure))  $\wedge$   
 $P$  ACTIONS\_IN (exec (PL complete))  $\wedge$   
 $(\forall v_{17}. P \ \text{ACTIONS\_IN} \ (\text{exec} \ (\text{PSG} \ v_{17}))) \wedge$   
 $P$  WITHDRAW (exec (PL secure))  $\wedge$   
 $P$  WITHDRAW (exec (PL withdraw))  $\wedge$   
 $(\forall v_{20}. P \ \text{WITHDRAW} \ (\text{exec} \ (\text{PSG} \ v_{20}))) \wedge$   
 $(\forall v_{21}. P \ \text{COMPLETE} \ (\text{exec} \ v_{21})) \Rightarrow$   
 $\forall v \ v_1. P \ v \ v_1$

[conductORPOut\_def]

$\vdash$  (conductORPOut CONDUCT\_ORP (exec (PL secure)) = Secure)  $\wedge$   
 (conductORPOut CONDUCT\_ORP (exec (PL plIncomplete)) =  
 ConductORP)  $\wedge$   
 (conductORPOut SECURE (exec (PSG actionsIn)) = ActionsIn)  $\wedge$   
 (conductORPOut SECURE (exec (PSG psgIncomplete)) = Secure)  $\wedge$   
 (conductORPOut ACTIONS\_IN (exec (PL withdraw)) = Withdraw)  $\wedge$   
 (conductORPOut ACTIONS\_IN (exec (PL plIncomplete)) =  
 ActionsIn)  $\wedge$   
 (conductORPOut WITHDRAW (exec (PL complete)) = Complete)  $\wedge$   
 (conductORPOut WITHDRAW (exec (PL plIncomplete)) =  
 Withdraw)  $\wedge$   
 (conductORPOut  $s$  (trap (PL  $cmd'$ )) = unauthorized)  $\wedge$   
 (conductORPOut  $s$  (trap (PSG  $cmd$ )) = unauthorized)  $\wedge$   
 (conductORPOut  $s$  (discard (PL  $cmd'$ )) = unAuthenticated)  $\wedge$   
 (conductORPOut  $s$  (discard (PSG  $cmd$ )) = unAuthenticated)

[conductORPOut\_ind]

$\vdash \forall P.$   
 $P \text{ CONDUCT\_ORP } (\text{exec } (\text{PL } \text{secure})) \wedge$   
 $P \text{ CONDUCT\_ORP } (\text{exec } (\text{PL } \text{plIncomplete})) \wedge$   
 $P \text{ SECURE } (\text{exec } (\text{PSG } \text{actionsIn})) \wedge$   
 $P \text{ SECURE } (\text{exec } (\text{PSG } \text{psgIncomplete})) \wedge$   
 $P \text{ ACTIONS\_IN } (\text{exec } (\text{PL } \text{withdraw})) \wedge$   
 $P \text{ ACTIONS\_IN } (\text{exec } (\text{PL } \text{plIncomplete})) \wedge$   
 $P \text{ WITHDRAW } (\text{exec } (\text{PL } \text{complete})) \wedge$   
 $P \text{ WITHDRAW } (\text{exec } (\text{PL } \text{plIncomplete})) \wedge$   
 $(\forall s \text{ cmd}. P \ s \ (\text{trap } (\text{PL } \text{cmd}))) \wedge$   
 $(\forall s \text{ cmd}. P \ s \ (\text{trap } (\text{PSG } \text{cmd}))) \wedge$   
 $(\forall s \text{ cmd}. P \ s \ (\text{discard } (\text{PL } \text{cmd}))) \wedge$   
 $(\forall s \text{ cmd}. P \ s \ (\text{discard } (\text{PSG } \text{cmd}))) \wedge$   
 $P \text{ CONDUCT\_ORP } (\text{exec } (\text{PL } \text{withdraw})) \wedge$   
 $P \text{ CONDUCT\_ORP } (\text{exec } (\text{PL } \text{complete})) \wedge$   
 $(\forall v_{11}. P \text{ CONDUCT\_ORP } (\text{exec } (\text{PSG } v_{11}))) \wedge$   
 $(\forall v_{13}. P \text{ SECURE } (\text{exec } (\text{PL } v_{13}))) \wedge$   
 $P \text{ ACTIONS\_IN } (\text{exec } (\text{PL } \text{secure})) \wedge$   
 $P \text{ ACTIONS\_IN } (\text{exec } (\text{PL } \text{complete})) \wedge$   
 $(\forall v_{17}. P \text{ ACTIONS\_IN } (\text{exec } (\text{PSG } v_{17}))) \wedge$   
 $P \text{ WITHDRAW } (\text{exec } (\text{PL } \text{secure})) \wedge$   
 $P \text{ WITHDRAW } (\text{exec } (\text{PL } \text{withdraw})) \wedge$   
 $(\forall v_{20}. P \text{ WITHDRAW } (\text{exec } (\text{PSG } v_{20}))) \wedge$   
 $(\forall v_{21}. P \text{ COMPLETE } (\text{exec } v_{21})) \Rightarrow$   
 $\forall v \ v_1. P \ v \ v_1$

[PlatoonLeader\_exec\_plCommand\_justified\_thm]

$\vdash \forall NS \text{ Out } M \ Oi \ Os.$   
 $\text{TR } (M, Oi, Os) \ (\text{exec } (\text{SLc } (\text{PL } \text{plCommand})))$   
 $(\text{CFG } \text{authTestConductORP } \text{ssmConductORPStateInterp}$   
 $(\text{secContextConductORP } \text{plCommand } \text{psgCommand } \text{incomplete})$   
 $(\text{Name PlatoonLeader says}$   
 $\text{prop } (\text{SOME } (\text{SLc } (\text{PL } \text{plCommand})))::\text{ins}) \ s \ \text{outs})$   
 $(\text{CFG } \text{authTestConductORP } \text{ssmConductORPStateInterp}$   
 $(\text{secContextConductORP } \text{plCommand } \text{psgCommand } \text{incomplete})$   
 $\text{ins } (NS \ s \ (\text{exec } (\text{SLc } (\text{PL } \text{plCommand}))))$   
 $(\text{Out } s \ (\text{exec } (\text{SLc } (\text{PL } \text{plCommand})))::\text{outs})) \iff$   
 $\text{authTestConductORP}$   
 $(\text{Name PlatoonLeader says}$   
 $\text{prop } (\text{SOME } (\text{SLc } (\text{PL } \text{plCommand})))) \wedge$   
 $\text{CFGInterpret } (M, Oi, Os)$   
 $(\text{CFG } \text{authTestConductORP } \text{ssmConductORPStateInterp}$   
 $(\text{secContextConductORP } \text{plCommand } \text{psgCommand } \text{incomplete})$   
 $(\text{Name PlatoonLeader says}$   
 $\text{prop } (\text{SOME } (\text{SLc } (\text{PL } \text{plCommand})))::\text{ins}) \ s \ \text{outs}) \wedge$   
 $(M, Oi, Os) \ \text{sat } \text{prop } (\text{SOME } (\text{SLc } (\text{PL } \text{plCommand}))))$

[PlatoonLeader\_plCommand\_lemma]

$\vdash \text{CFGInterpret } (M, Oi, Os)$   
 $(\text{CFG } \text{authTestConductORP } \text{ssmConductORPStateInterp}$



```

(secContextConductORP plCommand psgCommand incomplete)
(Name PlatoonLeader says
 prop (SOME (SLc (PL plCommand)))::ins) s outs) ⇒
(M, Oi, Os) sat prop (SOME (SLc (PL plCommand)))

```

[PlatoonSergeant\_exec\_psgCommand\_justified\_thm]

```

⊢ ∀ NS Out M Oi Os.
  TR (M, Oi, Os) (exec (SLc (PSG psgCommand)))
  (CFG authTestConductORP ssmConductORPStateInterp
   (secContextConductORP plCommand psgCommand incomplete)
   (Name PlatoonSergeant says
    prop (SOME (SLc (PSG psgCommand)))::ins) s outs)
  (CFG authTestConductORP ssmConductORPStateInterp
   (secContextConductORP plCommand psgCommand incomplete)
   ins (NS s (exec (SLc (PSG psgCommand))))
   (Out s (exec (SLc (PSG psgCommand)))::outs)) ⇐⇒
authTestConductORP
  (Name PlatoonSergeant says
   prop (SOME (SLc (PSG psgCommand)))) ∧
CFGInterpret (M, Oi, Os)
  (CFG authTestConductORP ssmConductORPStateInterp
   (secContextConductORP plCommand psgCommand incomplete)
   (Name PlatoonSergeant says
    prop (SOME (SLc (PSG psgCommand)))::ins) s outs) ∧
(M, Oi, Os) sat prop (SOME (SLc (PSG psgCommand)))

```

[PlatoonSergeant\_psgCommand\_lemma]

```

⊢ CFGInterpret (M, Oi, Os)
  (CFG authTestConductORP ssmConductORPStateInterp
   (secContextConductORP plCommand psgCommand incomplete)
   (Name PlatoonSergeant says
    prop (SOME (SLc (PSG psgCommand)))::ins) s outs) ⇒
(M, Oi, Os) sat prop (SOME (SLc (PSG psgCommand)))

```

## 8 ConductORPType Theory

**Built:** 10 June 2018

**Parent Theories:** indexedLists, patternMatches

### 8.1 Datatypes

*plCommand* = secure | withdraw | complete | plIncomplete

*psgCommand* = actionsIn | psgIncomplete

*slCommand* =  
 PL ConductORPType\$plCommand  
 | PSG ConductORPType\$psgCommand

$slOutput = \text{ConductORP} \mid \text{Secure} \mid \text{ActionsIn} \mid \text{Withdraw} \mid \text{Complete}$   
 $\mid \text{unAuthenticated} \mid \text{unAuthorized}$

$slState = \text{CONDUCT\_ORP} \mid \text{SECURE} \mid \text{ACTIONS\_IN} \mid \text{WITHDRAW}$   
 $\mid \text{COMPLETE}$

$stateRole = \text{PlatoonLeader} \mid \text{PlatoonSergeant}$

## 8.2 Theorems

[plCommand\_distinct\_clauses]

$\vdash \text{secure} \neq \text{withdraw} \wedge \text{secure} \neq \text{complete} \wedge$   
 $\text{secure} \neq \text{plIncomplete} \wedge \text{withdraw} \neq \text{complete} \wedge$   
 $\text{withdraw} \neq \text{plIncomplete} \wedge \text{complete} \neq \text{plIncomplete}$

[psgCommand\_distinct\_clauses]

$\vdash \text{actionsIn} \neq \text{psgIncomplete}$

[slCommand\_distinct\_clauses]

$\vdash \forall a' a. \text{PL } a \neq \text{PSG } a'$

[slCommand\_one\_one]

$\vdash (\forall a a'. (\text{PL } a = \text{PL } a') \iff (a = a')) \wedge$   
 $\forall a a'. (\text{PSG } a = \text{PSG } a') \iff (a = a')$

[slOutput\_distinct\_clauses]

$\vdash \text{ConductORP} \neq \text{Secure} \wedge \text{ConductORP} \neq \text{ActionsIn} \wedge$   
 $\text{ConductORP} \neq \text{Withdraw} \wedge \text{ConductORP} \neq \text{Complete} \wedge$   
 $\text{ConductORP} \neq \text{unAuthenticated} \wedge \text{ConductORP} \neq \text{unAuthorized} \wedge$   
 $\text{Secure} \neq \text{ActionsIn} \wedge \text{Secure} \neq \text{Withdraw} \wedge \text{Secure} \neq \text{Complete} \wedge$   
 $\text{Secure} \neq \text{unAuthenticated} \wedge \text{Secure} \neq \text{unAuthorized} \wedge$   
 $\text{ActionsIn} \neq \text{Withdraw} \wedge \text{ActionsIn} \neq \text{Complete} \wedge$   
 $\text{ActionsIn} \neq \text{unAuthenticated} \wedge \text{ActionsIn} \neq \text{unAuthorized} \wedge$   
 $\text{Withdraw} \neq \text{Complete} \wedge \text{Withdraw} \neq \text{unAuthenticated} \wedge$   
 $\text{Withdraw} \neq \text{unAuthorized} \wedge \text{Complete} \neq \text{unAuthenticated} \wedge$   
 $\text{Complete} \neq \text{unAuthorized} \wedge \text{unAuthenticated} \neq \text{unAuthorized}$

[slRole\_distinct\_clauses]

$\vdash \text{PlatoonLeader} \neq \text{PlatoonSergeant}$

[slState\_distinct\_clauses]

$\vdash \text{CONDUCT\_ORP} \neq \text{SECURE} \wedge \text{CONDUCT\_ORP} \neq \text{ACTIONS\_IN} \wedge$   
 $\text{CONDUCT\_ORP} \neq \text{WITHDRAW} \wedge \text{CONDUCT\_ORP} \neq \text{COMPLETE} \wedge$   
 $\text{SECURE} \neq \text{ACTIONS\_IN} \wedge \text{SECURE} \neq \text{WITHDRAW} \wedge \text{SECURE} \neq \text{COMPLETE} \wedge$   
 $\text{ACTIONS\_IN} \neq \text{WITHDRAW} \wedge \text{ACTIONS\_IN} \neq \text{COMPLETE} \wedge$   
 $\text{WITHDRAW} \neq \text{COMPLETE}$

## 9 ssmConductPB Theory

**Built:** 10 June 2018

**Parent Theories:** ConductPBType, ssm11, OMNIType

### 9.1 Definitions

[secContextConductPB\_def]

```

⊢ ∀ plcmd psgcmd incomplete.
  secContextConductPB plcmd psgcmd incomplete =
  [Name PlatoonLeader controls prop (SOME (SLc (PL plcmd)));
   Name PlatoonSergeant controls
   prop (SOME (SLc (PSG psgcmd)));
   Name PlatoonLeader says
   prop (SOME (SLc (PSG psgcmd))) impf prop NONE;
   Name PlatoonSergeant says
   prop (SOME (SLc (PL plcmd))) impf prop NONE]

```

[ssmConductPBStateInterp\_def]

```

⊢ ∀ slState. ssmConductPBStateInterp slState = TT

```

### 9.2 Theorems

[authTestConductPB\_cmd\_reject\_lemma]

```

⊢ ∀ cmd. ¬authTestConductPB (prop (SOME cmd))

```

[authTestConductPB\_def]

```

⊢ (authTestConductPB (Name PlatoonLeader says prop cmd) ⇔ T) ∧
  (authTestConductPB (Name PlatoonSergeant says prop cmd) ⇔
   T) ∧ (authTestConductPB TT ⇔ F) ∧
  (authTestConductPB FF ⇔ F) ∧
  (authTestConductPB (prop v) ⇔ F) ∧
  (authTestConductPB (notf v1) ⇔ F) ∧
  (authTestConductPB (v2 andf v3) ⇔ F) ∧
  (authTestConductPB (v4 orf v5) ⇔ F) ∧
  (authTestConductPB (v6 impf v7) ⇔ F) ∧
  (authTestConductPB (v8 eqf v9) ⇔ F) ∧
  (authTestConductPB (v10 says TT) ⇔ F) ∧
  (authTestConductPB (v10 says FF) ⇔ F) ∧
  (authTestConductPB (v133 meet v134 says prop v66) ⇔ F) ∧
  (authTestConductPB (v135 quoting v136 says prop v66) ⇔ F) ∧
  (authTestConductPB (v10 says notf v67) ⇔ F) ∧
  (authTestConductPB (v10 says (v68 andf v69)) ⇔ F) ∧
  (authTestConductPB (v10 says (v70 orf v71)) ⇔ F) ∧
  (authTestConductPB (v10 says (v72 impf v73)) ⇔ F) ∧
  (authTestConductPB (v10 says (v74 eqf v75)) ⇔ F) ∧
  (authTestConductPB (v10 says v76 says v77) ⇔ F) ∧

```

$(\text{authTestConductPB } (v_{10} \text{ says } v_{78} \text{ speaks\_for } v_{79}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{10} \text{ says } v_{80} \text{ controls } v_{81}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{10} \text{ says reps } v_{82} v_{83} v_{84}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{10} \text{ says } v_{85} \text{ domi } v_{86}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{10} \text{ says } v_{87} \text{ eqi } v_{88}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{10} \text{ says } v_{89} \text{ doms } v_{90}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{10} \text{ says } v_{91} \text{ eqs } v_{92}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{10} \text{ says } v_{93} \text{ eqn } v_{94}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{10} \text{ says } v_{95} \text{ lte } v_{96}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{10} \text{ says } v_{97} \text{ lt } v_{98}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{12} \text{ speaks\_for } v_{13}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{14} \text{ controls } v_{15}) \iff F) \wedge$   
 $(\text{authTestConductPB } (\text{reps } v_{16} v_{17} v_{18}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{19} \text{ domi } v_{20}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{21} \text{ eqi } v_{22}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{23} \text{ doms } v_{24}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{25} \text{ eqs } v_{26}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{27} \text{ eqn } v_{28}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{29} \text{ lte } v_{30}) \iff F) \wedge$   
 $(\text{authTestConductPB } (v_{31} \text{ lt } v_{32}) \iff F)$

[authTestConductPB\_ind]

$\vdash \forall P.$

$(\forall \text{cmd}. P (\text{Name PlatoonLeader says prop cmd})) \wedge$   
 $(\forall \text{cmd}. P (\text{Name PlatoonSergeant says prop cmd})) \wedge P \text{ TT} \wedge$   
 $P \text{ FF} \wedge (\forall v. P (\text{prop } v)) \wedge (\forall v_1. P (\text{notf } v_1)) \wedge$   
 $(\forall v_2 v_3. P (v_2 \text{ andf } v_3)) \wedge (\forall v_4 v_5. P (v_4 \text{ orf } v_5)) \wedge$   
 $(\forall v_6 v_7. P (v_6 \text{ impf } v_7)) \wedge (\forall v_8 v_9. P (v_8 \text{ eqf } v_9)) \wedge$   
 $(\forall v_{10}. P (v_{10} \text{ says TT})) \wedge (\forall v_{10}. P (v_{10} \text{ says FF})) \wedge$   
 $(\forall v_{133} v_{134} v_{66}. P (v_{133} \text{ meet } v_{134} \text{ says prop } v_{66})) \wedge$   
 $(\forall v_{135} v_{136} v_{66}. P (v_{135} \text{ quoting } v_{136} \text{ says prop } v_{66})) \wedge$   
 $(\forall v_{10} v_{67}. P (v_{10} \text{ says notf } v_{67})) \wedge$   
 $(\forall v_{10} v_{68} v_{69}. P (v_{10} \text{ says } (v_{68} \text{ andf } v_{69}))) \wedge$   
 $(\forall v_{10} v_{70} v_{71}. P (v_{10} \text{ says } (v_{70} \text{ orf } v_{71}))) \wedge$   
 $(\forall v_{10} v_{72} v_{73}. P (v_{10} \text{ says } (v_{72} \text{ impf } v_{73}))) \wedge$   
 $(\forall v_{10} v_{74} v_{75}. P (v_{10} \text{ says } (v_{74} \text{ eqf } v_{75}))) \wedge$   
 $(\forall v_{10} v_{76} v_{77}. P (v_{10} \text{ says } v_{76} \text{ says } v_{77})) \wedge$   
 $(\forall v_{10} v_{78} v_{79}. P (v_{10} \text{ says } v_{78} \text{ speaks\_for } v_{79})) \wedge$   
 $(\forall v_{10} v_{80} v_{81}. P (v_{10} \text{ says } v_{80} \text{ controls } v_{81})) \wedge$   
 $(\forall v_{10} v_{82} v_{83} v_{84}. P (v_{10} \text{ says reps } v_{82} v_{83} v_{84})) \wedge$   
 $(\forall v_{10} v_{85} v_{86}. P (v_{10} \text{ says } v_{85} \text{ domi } v_{86})) \wedge$   
 $(\forall v_{10} v_{87} v_{88}. P (v_{10} \text{ says } v_{87} \text{ eqi } v_{88})) \wedge$   
 $(\forall v_{10} v_{89} v_{90}. P (v_{10} \text{ says } v_{89} \text{ doms } v_{90})) \wedge$   
 $(\forall v_{10} v_{91} v_{92}. P (v_{10} \text{ says } v_{91} \text{ eqs } v_{92})) \wedge$   
 $(\forall v_{10} v_{93} v_{94}. P (v_{10} \text{ says } v_{93} \text{ eqn } v_{94})) \wedge$   
 $(\forall v_{10} v_{95} v_{96}. P (v_{10} \text{ says } v_{95} \text{ lte } v_{96})) \wedge$   
 $(\forall v_{10} v_{97} v_{98}. P (v_{10} \text{ says } v_{97} \text{ lt } v_{98})) \wedge$   
 $(\forall v_{12} v_{13}. P (v_{12} \text{ speaks\_for } v_{13})) \wedge$   
 $(\forall v_{14} v_{15}. P (v_{14} \text{ controls } v_{15})) \wedge$

$$\begin{aligned}
& (\forall v_{16} v_{17} v_{18}. P (\text{reps } v_{16} v_{17} v_{18})) \wedge \\
& (\forall v_{19} v_{20}. P (v_{19} \text{ domi } v_{20})) \wedge \\
& (\forall v_{21} v_{22}. P (v_{21} \text{ eqi } v_{22})) \wedge \\
& (\forall v_{23} v_{24}. P (v_{23} \text{ doms } v_{24})) \wedge \\
& (\forall v_{25} v_{26}. P (v_{25} \text{ eqs } v_{26})) \wedge (\forall v_{27} v_{28}. P (v_{27} \text{ eqn } v_{28})) \wedge \\
& (\forall v_{29} v_{30}. P (v_{29} \text{ lte } v_{30})) \wedge (\forall v_{31} v_{32}. P (v_{31} \text{ lt } v_{32})) \Rightarrow \\
& \forall v. P v
\end{aligned}$$

[conductPBNS\_def]

$$\begin{aligned}
& \vdash (\text{conductPBNS CONDUCT\_PB (exec (PL securePB))} = \text{SECURE\_PB}) \wedge \\
& (\text{conductPBNS CONDUCT\_PB (exec (PL plIncompletePB))} = \\
& \text{CONDUCT\_PB}) \wedge \\
& (\text{conductPBNS SECURE\_PB (exec (PSG actionsInPB))} = \\
& \text{ACTIONS\_IN\_PB}) \wedge \\
& (\text{conductPBNS SECURE\_PB (exec (PSG psgIncompletePB))} = \\
& \text{SECURE\_PB}) \wedge \\
& (\text{conductPBNS ACTIONS\_IN\_PB (exec (PL withdrawPB))} = \\
& \text{WITHDRAW\_PB}) \wedge \\
& (\text{conductPBNS ACTIONS\_IN\_PB (exec (PL plIncompletePB))} = \\
& \text{ACTIONS\_IN\_PB}) \wedge \\
& (\text{conductPBNS WITHDRAW\_PB (exec (PL completePB))} = \\
& \text{COMPLETE\_PB}) \wedge \\
& (\text{conductPBNS WITHDRAW\_PB (exec (PL plIncompletePB))} = \\
& \text{WITHDRAW\_PB}) \wedge (\text{conductPBNS } s \text{ (trap (PL cmd'))} = s) \wedge \\
& (\text{conductPBNS } s \text{ (trap (PSG cmd))} = s) \wedge \\
& (\text{conductPBNS } s \text{ (discard (PL cmd'))} = s) \wedge \\
& (\text{conductPBNS } s \text{ (discard (PSG cmd))} = s)
\end{aligned}$$

[conductPBNS\_ind]

$$\begin{aligned}
& \vdash \forall P. \\
& P \text{ CONDUCT\_PB (exec (PL securePB))} \wedge \\
& P \text{ CONDUCT\_PB (exec (PL plIncompletePB))} \wedge \\
& P \text{ SECURE\_PB (exec (PSG actionsInPB))} \wedge \\
& P \text{ SECURE\_PB (exec (PSG psgIncompletePB))} \wedge \\
& P \text{ ACTIONS\_IN\_PB (exec (PL withdrawPB))} \wedge \\
& P \text{ ACTIONS\_IN\_PB (exec (PL plIncompletePB))} \wedge \\
& P \text{ WITHDRAW\_PB (exec (PL completePB))} \wedge \\
& P \text{ WITHDRAW\_PB (exec (PL plIncompletePB))} \wedge \\
& (\forall s \text{ cmd}. P s \text{ (trap (PL cmd))}) \wedge \\
& (\forall s \text{ cmd}. P s \text{ (trap (PSG cmd))}) \wedge \\
& (\forall s \text{ cmd}. P s \text{ (discard (PL cmd))}) \wedge \\
& (\forall s \text{ cmd}. P s \text{ (discard (PSG cmd))}) \wedge \\
& P \text{ CONDUCT\_PB (exec (PL withdrawPB))} \wedge \\
& P \text{ CONDUCT\_PB (exec (PL completePB))} \wedge \\
& (\forall v_{11}. P \text{ CONDUCT\_PB (exec (PSG } v_{11}))}) \wedge \\
& (\forall v_{13}. P \text{ SECURE\_PB (exec (PL } v_{13}))}) \wedge \\
& P \text{ ACTIONS\_IN\_PB (exec (PL securePB))} \wedge \\
& P \text{ ACTIONS\_IN\_PB (exec (PL completePB))} \wedge \\
& (\forall v_{17}. P \text{ ACTIONS\_IN\_PB (exec (PSG } v_{17}))}) \wedge
\end{aligned}$$

$$\begin{aligned}
& P \text{ WITHDRAW\_PB } (\text{exec } (\text{PL securePB})) \wedge \\
& P \text{ WITHDRAW\_PB } (\text{exec } (\text{PL withdrawPB})) \wedge \\
& (\forall v_{20}. P \text{ WITHDRAW\_PB } (\text{exec } (\text{PSG } v_{20}))) \wedge \\
& (\forall v_{21}. P \text{ COMPLETE\_PB } (\text{exec } v_{21})) \Rightarrow \\
& \forall v \ v_1. P \ v \ v_1
\end{aligned}$$

[conductPBOut\_def]

$$\begin{aligned}
& \vdash (\text{conductPBOut CONDUCT\_PB } (\text{exec } (\text{PL securePB})) = \text{ConductPB}) \wedge \\
& (\text{conductPBOut CONDUCT\_PB } (\text{exec } (\text{PL plIncompletePB})) = \\
& \text{ConductPB}) \wedge \\
& (\text{conductPBOut SECURE\_PB } (\text{exec } (\text{PSG actionsInPB})) = \\
& \text{SecurePB}) \wedge \\
& (\text{conductPBOut SECURE\_PB } (\text{exec } (\text{PSG psgIncompletePB})) = \\
& \text{SecurePB}) \wedge \\
& (\text{conductPBOut ACTIONS\_IN\_PB } (\text{exec } (\text{PL withdrawPB})) = \\
& \text{ActionsInPB}) \wedge \\
& (\text{conductPBOut ACTIONS\_IN\_PB } (\text{exec } (\text{PL plIncompletePB})) = \\
& \text{ActionsInPB}) \wedge \\
& (\text{conductPBOut WITHDRAW\_PB } (\text{exec } (\text{PL completePB})) = \\
& \text{WithdrawPB}) \wedge \\
& (\text{conductPBOut WITHDRAW\_PB } (\text{exec } (\text{PL plIncompletePB})) = \\
& \text{WithdrawPB}) \wedge \\
& (\text{conductPBOut } s \ (\text{trap } (\text{PL } cmd')) = \text{unAuthorized}) \wedge \\
& (\text{conductPBOut } s \ (\text{trap } (\text{PSG } cmd)) = \text{unAuthorized}) \wedge \\
& (\text{conductPBOut } s \ (\text{discard } (\text{PL } cmd')) = \text{unAuthenticated}) \wedge \\
& (\text{conductPBOut } s \ (\text{discard } (\text{PSG } cmd)) = \text{unAuthenticated})
\end{aligned}$$

[conductPBOut\_ind]

$$\begin{aligned}
& \vdash \forall P. \\
& P \text{ CONDUCT\_PB } (\text{exec } (\text{PL securePB})) \wedge \\
& P \text{ CONDUCT\_PB } (\text{exec } (\text{PL plIncompletePB})) \wedge \\
& P \text{ SECURE\_PB } (\text{exec } (\text{PSG actionsInPB})) \wedge \\
& P \text{ SECURE\_PB } (\text{exec } (\text{PSG psgIncompletePB})) \wedge \\
& P \text{ ACTIONS\_IN\_PB } (\text{exec } (\text{PL withdrawPB})) \wedge \\
& P \text{ ACTIONS\_IN\_PB } (\text{exec } (\text{PL plIncompletePB})) \wedge \\
& P \text{ WITHDRAW\_PB } (\text{exec } (\text{PL completePB})) \wedge \\
& P \text{ WITHDRAW\_PB } (\text{exec } (\text{PL plIncompletePB})) \wedge \\
& (\forall s \ cmd. P \ s \ (\text{trap } (\text{PL } cmd))) \wedge \\
& (\forall s \ cmd. P \ s \ (\text{trap } (\text{PSG } cmd))) \wedge \\
& (\forall s \ cmd. P \ s \ (\text{discard } (\text{PL } cmd))) \wedge \\
& (\forall s \ cmd. P \ s \ (\text{discard } (\text{PSG } cmd))) \wedge \\
& P \text{ CONDUCT\_PB } (\text{exec } (\text{PL withdrawPB})) \wedge \\
& P \text{ CONDUCT\_PB } (\text{exec } (\text{PL completePB})) \wedge \\
& (\forall v_{11}. P \text{ CONDUCT\_PB } (\text{exec } (\text{PSG } v_{11}))) \wedge \\
& (\forall v_{13}. P \text{ SECURE\_PB } (\text{exec } (\text{PL } v_{13}))) \wedge \\
& P \text{ ACTIONS\_IN\_PB } (\text{exec } (\text{PL securePB})) \wedge \\
& P \text{ ACTIONS\_IN\_PB } (\text{exec } (\text{PL completePB})) \wedge \\
& (\forall v_{17}. P \text{ ACTIONS\_IN\_PB } (\text{exec } (\text{PSG } v_{17}))) \wedge \\
& P \text{ WITHDRAW\_PB } (\text{exec } (\text{PL securePB})) \wedge
\end{aligned}$$

$$\begin{aligned}
& P \text{ WITHDRAW\_PB } (\text{exec } (\text{PL withdrawPB})) \wedge \\
& (\forall v_{20}. P \text{ WITHDRAW\_PB } (\text{exec } (\text{PSG } v_{20}))) \wedge \\
& (\forall v_{21}. P \text{ COMPLETE\_PB } (\text{exec } v_{21})) \Rightarrow \\
& \forall v \ v_1. P \ v \ v_1
\end{aligned}$$

[PlatoonLeader\_exec\_plCommandPB\_justified\_thm]

$$\begin{aligned}
& \vdash \forall NS \text{ Out } M \ Oi \ Os. \\
& \text{TR } (M, Oi, Os) (\text{exec } (\text{SLc } (\text{PL } plCommand))) \\
& \quad (\text{CFG authTestConductPB ssmConductPBStateInterp} \\
& \quad \quad (\text{secContextConductPB } plCommand \ psgCommand \ incomplete) \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop } (\text{SOME } (\text{SLc } (\text{PL } plCommand)))::ins) \ s \ outs) \\
& \quad (\text{CFG authTestConductPB ssmConductPBStateInterp} \\
& \quad \quad (\text{secContextConductPB } plCommand \ psgCommand \ incomplete) \\
& \quad \quad \text{ins } (NS \ s \ (\text{exec } (\text{SLc } (\text{PL } plCommand)))) \\
& \quad \quad (\text{Out } s \ (\text{exec } (\text{SLc } (\text{PL } plCommand)))::outs)) \iff \\
& \text{authTestConductPB} \\
& \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop } (\text{SOME } (\text{SLc } (\text{PL } plCommand)))) \wedge \\
& \text{CFGInterpret } (M, Oi, Os) \\
& \quad (\text{CFG authTestConductPB ssmConductPBStateInterp} \\
& \quad \quad (\text{secContextConductPB } plCommand \ psgCommand \ incomplete) \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop } (\text{SOME } (\text{SLc } (\text{PL } plCommand)))::ins) \ s \ outs) \wedge \\
& \quad (M, Oi, Os) \text{ sat prop } (\text{SOME } (\text{SLc } (\text{PL } plCommand)))
\end{aligned}$$

[PlatoonLeader\_plCommandPB\_lemma]

$$\begin{aligned}
& \vdash \text{CFGInterpret } (M, Oi, Os) \\
& \quad (\text{CFG authTestConductPB ssmConductPBStateInterp} \\
& \quad \quad (\text{secContextConductPB } plCommand \ psgCommand \ incomplete) \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop } (\text{SOME } (\text{SLc } (\text{PL } plCommand)))::ins) \ s \ outs) \Rightarrow \\
& \quad (M, Oi, Os) \text{ sat prop } (\text{SOME } (\text{SLc } (\text{PL } plCommand)))
\end{aligned}$$

[PlatoonSergeant\_exec\_psgCommandPB\_justified\_thm]

$$\begin{aligned}
& \vdash \forall NS \text{ Out } M \ Oi \ Os. \\
& \text{TR } (M, Oi, Os) (\text{exec } (\text{SLc } (\text{PSG } psgCommand))) \\
& \quad (\text{CFG authTestConductPB ssmConductPBStateInterp} \\
& \quad \quad (\text{secContextConductPB } plCommand \ psgCommand \ incomplete) \\
& \quad \quad (\text{Name PlatoonSergeant says} \\
& \quad \quad \quad \text{prop } (\text{SOME } (\text{SLc } (\text{PSG } psgCommand)))::ins) \ s \ outs) \\
& \quad (\text{CFG authTestConductPB ssmConductPBStateInterp} \\
& \quad \quad (\text{secContextConductPB } plCommand \ psgCommand \ incomplete) \\
& \quad \quad \text{ins } (NS \ s \ (\text{exec } (\text{SLc } (\text{PSG } psgCommand)))) \\
& \quad \quad (\text{Out } s \ (\text{exec } (\text{SLc } (\text{PSG } psgCommand)))::outs)) \iff \\
& \text{authTestConductPB} \\
& \quad (\text{Name PlatoonSergeant says} \\
& \quad \quad \text{prop } (\text{SOME } (\text{SLc } (\text{PSG } psgCommand)))) \wedge
\end{aligned}$$

```

CFGInterpret (M, Oi, Os)
  (CFG authTestConductPB ssmConductPBStateInterp
   (secContextConductPB plCommand psgCommand incomplete)
   (Name PlatoonSergeant says
    prop (SOME (SLc (PSG psgCommand)))::ins) s outs) ∧
  (M, Oi, Os) sat prop (SOME (SLc (PSG psgCommand)))

```

[PlatoonSergeant\_psgCommandPB\_lemma]

```

⊢ CFGInterpret (M, Oi, Os)
  (CFG authTestConductPB ssmConductPBStateInterp
   (secContextConductPB plCommand psgCommand incomplete)
   (Name PlatoonSergeant says
    prop (SOME (SLc (PSG psgCommand)))::ins) s outs) ⇒
  (M, Oi, Os) sat prop (SOME (SLc (PSG psgCommand)))

```

## 10 ConductPBType Theory

**Built:** 10 June 2018

**Parent Theories:** indexedLists, patternMatches

### 10.1 Datatypes

```

plCommandPB = securePB | withdrawPB | completePB
             | plIncompletePB

```

```

psgCommandPB = actionsInPB | psgIncompletePB

```

```

slCommand = PL plCommandPB | PSG psgCommandPB

```

```

slOutput = ConductPB | SecurePB | ActionsInPB | WithdrawPB
          | CompletePB | unAuthenticated | unAuthorized

```

```

slState = CONDUCT_PB | SECURE_PB | ACTIONS_IN_PB | WITHDRAW_PB
         | COMPLETE_PB

```

```

stateRole = PlatoonLeader | PlatoonSergeant

```

### 10.2 Theorems

[plCommandPB\_distinct\_clauses]

```

⊢ securePB ≠ withdrawPB ∧ securePB ≠ completePB ∧
  securePB ≠ plIncompletePB ∧ withdrawPB ≠ completePB ∧
  withdrawPB ≠ plIncompletePB ∧ completePB ≠ plIncompletePB

```

[psgCommandPB\_distinct\_clauses]

```

⊢ actionsInPB ≠ psgIncompletePB

```



[slCommand\_distinct\_clauses]

$\vdash \forall a' a. \text{PL } a \neq \text{PSG } a'$

[slCommand\_one\_one]

$\vdash (\forall a a'. (\text{PL } a = \text{PL } a') \iff (a = a')) \wedge$   
 $\forall a a'. (\text{PSG } a = \text{PSG } a') \iff (a = a')$

[slOutput\_distinct\_clauses]

$\vdash \text{ConductPB} \neq \text{SecurePB} \wedge \text{ConductPB} \neq \text{ActionsInPB} \wedge$   
 $\text{ConductPB} \neq \text{WithdrawPB} \wedge \text{ConductPB} \neq \text{CompletePB} \wedge$   
 $\text{ConductPB} \neq \text{unAuthenticated} \wedge \text{ConductPB} \neq \text{unAuthorized} \wedge$   
 $\text{SecurePB} \neq \text{ActionsInPB} \wedge \text{SecurePB} \neq \text{WithdrawPB} \wedge$   
 $\text{SecurePB} \neq \text{CompletePB} \wedge \text{SecurePB} \neq \text{unAuthenticated} \wedge$   
 $\text{SecurePB} \neq \text{unAuthorized} \wedge \text{ActionsInPB} \neq \text{WithdrawPB} \wedge$   
 $\text{ActionsInPB} \neq \text{CompletePB} \wedge \text{ActionsInPB} \neq \text{unAuthenticated} \wedge$   
 $\text{ActionsInPB} \neq \text{unAuthorized} \wedge \text{WithdrawPB} \neq \text{CompletePB} \wedge$   
 $\text{WithdrawPB} \neq \text{unAuthenticated} \wedge \text{WithdrawPB} \neq \text{unAuthorized} \wedge$   
 $\text{CompletePB} \neq \text{unAuthenticated} \wedge \text{CompletePB} \neq \text{unAuthorized} \wedge$   
 $\text{unAuthenticated} \neq \text{unAuthorized}$

[slRole\_distinct\_clauses]

$\vdash \text{PlatoonLeader} \neq \text{PlatoonSergeant}$

[slState\_distinct\_clauses]

$\vdash \text{CONDUCT\_PB} \neq \text{SECURE\_PB} \wedge \text{CONDUCT\_PB} \neq \text{ACTIONS\_IN\_PB} \wedge$   
 $\text{CONDUCT\_PB} \neq \text{WITHDRAW\_PB} \wedge \text{CONDUCT\_PB} \neq \text{COMPLETE\_PB} \wedge$   
 $\text{SECURE\_PB} \neq \text{ACTIONS\_IN\_PB} \wedge \text{SECURE\_PB} \neq \text{WITHDRAW\_PB} \wedge$   
 $\text{SECURE\_PB} \neq \text{COMPLETE\_PB} \wedge \text{ACTIONS\_IN\_PB} \neq \text{WITHDRAW\_PB} \wedge$   
 $\text{ACTIONS\_IN\_PB} \neq \text{COMPLETE\_PB} \wedge \text{WITHDRAW\_PB} \neq \text{COMPLETE\_PB}$

## 11 ssmMoveToORP Theory

**Built:** 10 June 2018

**Parent Theories:** MoveToORPType, ssm11, OMNIType

### 11.1 Definitions

[secContextMoveToORP\_def]

$\vdash \forall cmd.$   
 $\text{secContextMoveToORP } cmd =$   
 $[\text{Name PlatoonLeader controls prop (SOME (SLc } cmd))]$

[ssmMoveToORPStateInterp\_def]

$\vdash \forall state. \text{ssmMoveToORPStateInterp } state = \text{TT}$

## 11.2 Theorems

[authTestMoveToORP\_cmd\_reject\_lemma]

$$\vdash \forall cmd. \neg \text{authTestMoveToORP} (\text{prop } (SOME \text{ cmd}))$$

[authTestMoveToORP\_def]

$$\begin{aligned} \vdash & (\text{authTestMoveToORP} (\text{Name PlatoonLeader says prop cmd}) \iff T) \wedge \\ & (\text{authTestMoveToORP TT} \iff F) \wedge (\text{authTestMoveToORP FF} \iff F) \wedge \\ & (\text{authTestMoveToORP} (\text{prop } v) \iff F) \wedge \\ & (\text{authTestMoveToORP} (\text{notf } v_1) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_2 \text{ andf } v_3) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_4 \text{ orf } v_5) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_6 \text{ impf } v_7) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_8 \text{ eqf } v_9) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says TT}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says FF}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{133} \text{ meet } v_{134} \text{ says prop } v_{66}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{135} \text{ quoting } v_{136} \text{ says prop } v_{66}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says notf } v_{67}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says } (v_{68} \text{ andf } v_{69})) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says } (v_{70} \text{ orf } v_{71})) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says } (v_{72} \text{ impf } v_{73})) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says } (v_{74} \text{ eqf } v_{75})) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says } v_{76} \text{ says } v_{77}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says } v_{78} \text{ speaks\_for } v_{79}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says } v_{80} \text{ controls } v_{81}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says reps } v_{82} \ v_{83} \ v_{84}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says } v_{85} \text{ domi } v_{86}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says } v_{87} \text{ eqi } v_{88}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says } v_{89} \text{ doms } v_{90}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says } v_{91} \text{ eqs } v_{92}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says } v_{93} \text{ eqn } v_{94}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says } v_{95} \text{ lte } v_{96}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{10} \text{ says } v_{97} \text{ lt } v_{98}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{12} \text{ speaks\_for } v_{13}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{14} \text{ controls } v_{15}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (\text{reps } v_{16} \ v_{17} \ v_{18}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{19} \text{ domi } v_{20}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{21} \text{ eqi } v_{22}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{23} \text{ doms } v_{24}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{25} \text{ eqs } v_{26}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{27} \text{ eqn } v_{28}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{29} \text{ lte } v_{30}) \iff F) \wedge \\ & (\text{authTestMoveToORP} (v_{31} \text{ lt } v_{32}) \iff F) \end{aligned}$$

[authTestMoveToORP\_ind]

$$\begin{aligned} \vdash & \forall P. \\ & (\forall cmd. P (\text{Name PlatoonLeader says prop cmd})) \wedge P \text{ TT} \wedge \\ & P \text{ FF} \wedge (\forall v. P (\text{prop } v)) \wedge (\forall v_1. P (\text{notf } v_1)) \wedge \end{aligned}$$

$$\begin{aligned}
& (\forall v_2 v_3. P (v_2 \text{ andf } v_3)) \wedge (\forall v_4 v_5. P (v_4 \text{ orf } v_5)) \wedge \\
& (\forall v_6 v_7. P (v_6 \text{ impf } v_7)) \wedge (\forall v_8 v_9. P (v_8 \text{ eqf } v_9)) \wedge \\
& (\forall v_{10}. P (v_{10} \text{ says TT})) \wedge (\forall v_{10}. P (v_{10} \text{ says FF})) \wedge \\
& (\forall v_{133} v_{134} v_{66}. P (v_{133} \text{ meet } v_{134} \text{ says prop } v_{66})) \wedge \\
& (\forall v_{135} v_{136} v_{66}. P (v_{135} \text{ quoting } v_{136} \text{ says prop } v_{66})) \wedge \\
& (\forall v_{10} v_{67}. P (v_{10} \text{ says notf } v_{67})) \wedge \\
& (\forall v_{10} v_{68} v_{69}. P (v_{10} \text{ says } (v_{68} \text{ andf } v_{69}))) \wedge \\
& (\forall v_{10} v_{70} v_{71}. P (v_{10} \text{ says } (v_{70} \text{ orf } v_{71}))) \wedge \\
& (\forall v_{10} v_{72} v_{73}. P (v_{10} \text{ says } (v_{72} \text{ impf } v_{73}))) \wedge \\
& (\forall v_{10} v_{74} v_{75}. P (v_{10} \text{ says } (v_{74} \text{ eqf } v_{75}))) \wedge \\
& (\forall v_{10} v_{76} v_{77}. P (v_{10} \text{ says } v_{76} \text{ says } v_{77})) \wedge \\
& (\forall v_{10} v_{78} v_{79}. P (v_{10} \text{ says } v_{78} \text{ speaks\_for } v_{79})) \wedge \\
& (\forall v_{10} v_{80} v_{81}. P (v_{10} \text{ says } v_{80} \text{ controls } v_{81})) \wedge \\
& (\forall v_{10} v_{82} v_{83} v_{84}. P (v_{10} \text{ says reps } v_{82} v_{83} v_{84})) \wedge \\
& (\forall v_{10} v_{85} v_{86}. P (v_{10} \text{ says } v_{85} \text{ domi } v_{86})) \wedge \\
& (\forall v_{10} v_{87} v_{88}. P (v_{10} \text{ says } v_{87} \text{ eqi } v_{88})) \wedge \\
& (\forall v_{10} v_{89} v_{90}. P (v_{10} \text{ says } v_{89} \text{ doms } v_{90})) \wedge \\
& (\forall v_{10} v_{91} v_{92}. P (v_{10} \text{ says } v_{91} \text{ eqs } v_{92})) \wedge \\
& (\forall v_{10} v_{93} v_{94}. P (v_{10} \text{ says } v_{93} \text{ eqn } v_{94})) \wedge \\
& (\forall v_{10} v_{95} v_{96}. P (v_{10} \text{ says } v_{95} \text{ lte } v_{96})) \wedge \\
& (\forall v_{10} v_{97} v_{98}. P (v_{10} \text{ says } v_{97} \text{ lt } v_{98})) \wedge \\
& (\forall v_{12} v_{13}. P (v_{12} \text{ speaks\_for } v_{13})) \wedge \\
& (\forall v_{14} v_{15}. P (v_{14} \text{ controls } v_{15})) \wedge \\
& (\forall v_{16} v_{17} v_{18}. P (\text{reps } v_{16} v_{17} v_{18})) \wedge \\
& (\forall v_{19} v_{20}. P (v_{19} \text{ domi } v_{20})) \wedge \\
& (\forall v_{21} v_{22}. P (v_{21} \text{ eqi } v_{22})) \wedge \\
& (\forall v_{23} v_{24}. P (v_{23} \text{ doms } v_{24})) \wedge \\
& (\forall v_{25} v_{26}. P (v_{25} \text{ eqs } v_{26})) \wedge (\forall v_{27} v_{28}. P (v_{27} \text{ eqn } v_{28})) \wedge \\
& (\forall v_{29} v_{30}. P (v_{29} \text{ lte } v_{30})) \wedge (\forall v_{31} v_{32}. P (v_{31} \text{ lt } v_{32})) \Rightarrow \\
& \forall v. P v
\end{aligned}$$

[moveToORPNS\_def]

$$\begin{aligned}
& \vdash (\text{moveToORPNS MOVE\_TO\_ORP (exec (SLc pltForm))} = \text{PLT\_FORM}) \wedge \\
& (\text{moveToORPNS MOVE\_TO\_ORP (exec (SLc incomplete))} = \\
& \quad \text{MOVE\_TO\_ORP}) \wedge \\
& (\text{moveToORPNS PLT\_FORM (exec (SLc pltMove))} = \text{PLT\_MOVE}) \wedge \\
& (\text{moveToORPNS PLT\_FORM (exec (SLc incomplete))} = \text{PLT\_FORM}) \wedge \\
& (\text{moveToORPNS PLT\_MOVE (exec (SLc pltSecureHalt))} = \\
& \quad \text{PLT\_SECURE\_HALT}) \wedge \\
& (\text{moveToORPNS PLT\_MOVE (exec (SLc incomplete))} = \text{PLT\_MOVE}) \wedge \\
& (\text{moveToORPNS PLT\_SECURE\_HALT (exec (SLc complete))} = \\
& \quad \text{COMPLETE}) \wedge \\
& (\text{moveToORPNS PLT\_SECURE\_HALT (exec (SLc incomplete))} = \\
& \quad \text{PLT\_SECURE\_HALT}) \wedge (\text{moveToORPNS } s \text{ (trap (SLc cmd))} = s) \wedge \\
& (\text{moveToORPNS } s \text{ (discard (SLc cmd))} = s)
\end{aligned}$$

[moveToORPNS\_ind]

$$\begin{aligned}
& \vdash \forall P. \\
& \quad P \text{ MOVE\_TO\_ORP (exec (SLc pltForm))} \wedge
\end{aligned}$$

$P \text{ MOVE\_TO\_ORP (exec (SLc incomplete))} \wedge$   
 $P \text{ PLT\_FORM (exec (SLc pltMove))} \wedge$   
 $P \text{ PLT\_FORM (exec (SLc incomplete))} \wedge$   
 $P \text{ PLT\_MOVE (exec (SLc pltSecureHalt))} \wedge$   
 $P \text{ PLT\_MOVE (exec (SLc incomplete))} \wedge$   
 $P \text{ PLT\_SECURE\_HALT (exec (SLc complete))} \wedge$   
 $P \text{ PLT\_SECURE\_HALT (exec (SLc incomplete))} \wedge$   
 $(\forall s \text{ cmd. } P \text{ s (trap (SLc cmd))}) \wedge$   
 $(\forall s \text{ cmd. } P \text{ s (discard (SLc cmd))}) \wedge$   
 $(\forall s \text{ v}_6. P \text{ s (discard (ESCc v}_6\text{))}) \wedge$   
 $(\forall s \text{ v}_9. P \text{ s (trap (ESCc v}_9\text{))}) \wedge$   
 $(\forall v_{12}. P \text{ MOVE\_TO\_ORP (exec (ESCc v}_{12}\text{))}) \wedge$   
 $P \text{ MOVE\_TO\_ORP (exec (SLc pltMove))} \wedge$   
 $P \text{ MOVE\_TO\_ORP (exec (SLc pltSecureHalt))} \wedge$   
 $P \text{ MOVE\_TO\_ORP (exec (SLc complete))} \wedge$   
 $(\forall v_{15}. P \text{ PLT\_FORM (exec (ESCc v}_{15}\text{))}) \wedge$   
 $P \text{ PLT\_FORM (exec (SLc pltForm))} \wedge$   
 $P \text{ PLT\_FORM (exec (SLc pltSecureHalt))} \wedge$   
 $P \text{ PLT\_FORM (exec (SLc complete))} \wedge$   
 $(\forall v_{18}. P \text{ PLT\_MOVE (exec (ESCc v}_{18}\text{))}) \wedge$   
 $P \text{ PLT\_MOVE (exec (SLc pltForm))} \wedge$   
 $P \text{ PLT\_MOVE (exec (SLc pltMove))} \wedge$   
 $P \text{ PLT\_MOVE (exec (SLc complete))} \wedge$   
 $(\forall v_{21}. P \text{ PLT\_SECURE\_HALT (exec (ESCc v}_{21}\text{))}) \wedge$   
 $P \text{ PLT\_SECURE\_HALT (exec (SLc pltForm))} \wedge$   
 $P \text{ PLT\_SECURE\_HALT (exec (SLc pltMove))} \wedge$   
 $P \text{ PLT\_SECURE\_HALT (exec (SLc pltSecureHalt))} \wedge$   
 $(\forall v_{23}. P \text{ COMPLETE (exec v}_{23}\text{)}) \Rightarrow$   
 $\forall v \text{ v}_1. P \text{ v v}_1$

[moveToORPOut\_def]

$\vdash (\text{moveToORPOut MOVE\_TO\_ORP (exec (SLc pltForm))} = \text{PLTForm}) \wedge$   
 $(\text{moveToORPOut MOVE\_TO\_ORP (exec (SLc incomplete))} =$   
 $\text{MoveToORP}) \wedge$   
 $(\text{moveToORPOut PLT\_FORM (exec (SLc pltMove))} = \text{PLTMove}) \wedge$   
 $(\text{moveToORPOut PLT\_FORM (exec (SLc incomplete))} = \text{PLTForm}) \wedge$   
 $(\text{moveToORPOut PLT\_MOVE (exec (SLc pltSecureHalt))} =$   
 $\text{PLTSecureHalt}) \wedge$   
 $(\text{moveToORPOut PLT\_MOVE (exec (SLc incomplete))} = \text{PLTMove}) \wedge$   
 $(\text{moveToORPOut PLT\_SECURE\_HALT (exec (SLc complete))} =$   
 $\text{Complete}) \wedge$   
 $(\text{moveToORPOut PLT\_SECURE\_HALT (exec (SLc incomplete))} =$   
 $\text{PLTSecureHalt}) \wedge$   
 $(\text{moveToORPOut s (trap (SLc cmd))} = \text{unAuthorized}) \wedge$   
 $(\text{moveToORPOut s (discard (SLc cmd))} = \text{unAuthenticated})$

[moveToORPOut\_ind]

$\vdash \forall P.$   
 $P \text{ MOVE\_TO\_ORP (exec (SLc pltForm))} \wedge$

$P \text{ MOVE\_TO\_ORP (exec (SLc incomplete))} \wedge$   
 $P \text{ PLT\_FORM (exec (SLc pltMove))} \wedge$   
 $P \text{ PLT\_FORM (exec (SLc incomplete))} \wedge$   
 $P \text{ PLT\_MOVE (exec (SLc pltSecureHalt))} \wedge$   
 $P \text{ PLT\_MOVE (exec (SLc incomplete))} \wedge$   
 $P \text{ PLT\_SECURE\_HALT (exec (SLc complete))} \wedge$   
 $P \text{ PLT\_SECURE\_HALT (exec (SLc incomplete))} \wedge$   
 $(\forall s \text{ cmd. } P \text{ s (trap (SLc cmd))}) \wedge$   
 $(\forall s \text{ cmd. } P \text{ s (discard (SLc cmd))}) \wedge$   
 $(\forall s \text{ v}_6. P \text{ s (discard (ESCc v}_6\text{))}) \wedge$   
 $(\forall s \text{ v}_9. P \text{ s (trap (ESCc v}_9\text{))}) \wedge$   
 $(\forall v_{12}. P \text{ MOVE\_TO\_ORP (exec (ESCc v}_{12}\text{))}) \wedge$   
 $P \text{ MOVE\_TO\_ORP (exec (SLc pltMove))} \wedge$   
 $P \text{ MOVE\_TO\_ORP (exec (SLc pltSecureHalt))} \wedge$   
 $P \text{ MOVE\_TO\_ORP (exec (SLc complete))} \wedge$   
 $(\forall v_{15}. P \text{ PLT\_FORM (exec (ESCc v}_{15}\text{))}) \wedge$   
 $P \text{ PLT\_FORM (exec (SLc pltForm))} \wedge$   
 $P \text{ PLT\_FORM (exec (SLc pltSecureHalt))} \wedge$   
 $P \text{ PLT\_FORM (exec (SLc complete))} \wedge$   
 $(\forall v_{18}. P \text{ PLT\_MOVE (exec (ESCc v}_{18}\text{))}) \wedge$   
 $P \text{ PLT\_MOVE (exec (SLc pltForm))} \wedge$   
 $P \text{ PLT\_MOVE (exec (SLc pltMove))} \wedge$   
 $P \text{ PLT\_MOVE (exec (SLc complete))} \wedge$   
 $(\forall v_{21}. P \text{ PLT\_SECURE\_HALT (exec (ESCc v}_{21}\text{))}) \wedge$   
 $P \text{ PLT\_SECURE\_HALT (exec (SLc pltForm))} \wedge$   
 $P \text{ PLT\_SECURE\_HALT (exec (SLc pltMove))} \wedge$   
 $P \text{ PLT\_SECURE\_HALT (exec (SLc pltSecureHalt))} \wedge$   
 $(\forall v_{23}. P \text{ COMPLETE (exec v}_{23}\text{)}) \Rightarrow$   
 $\forall v \text{ v}_1. P \text{ v v}_1$

[PlatoonLeader\_exec\_slCommand\_justified\_thm]

$\vdash \forall NS \text{ Out } M \text{ Oi } Os.$

$\text{TR } (M, Oi, Os) \text{ (exec (SLc slCommand))}$   
 $(\text{CFG authTestMoveToORP ssmMoveToORPStateInterp}$   
 $(\text{secContextMoveToORP slCommand})$   
 $(\text{Name PlatoonLeader says prop (SOME (SLc slCommand))} ::$   
 $\text{ins) s outs})$   
 $(\text{CFG authTestMoveToORP ssmMoveToORPStateInterp}$   
 $(\text{secContextMoveToORP slCommand) ins}$   
 $(NS \text{ s (exec (SLc slCommand))})$   
 $(\text{Out s (exec (SLc slCommand))} :: \text{outs})) \iff$   
 $\text{authTestMoveToORP}$   
 $(\text{Name PlatoonLeader says prop (SOME (SLc slCommand))}) \wedge$   
 $\text{CFGInterpret } (M, Oi, Os)$   
 $(\text{CFG authTestMoveToORP ssmMoveToORPStateInterp}$   
 $(\text{secContextMoveToORP slCommand})$   
 $(\text{Name PlatoonLeader says prop (SOME (SLc slCommand))} ::$   
 $\text{ins) s outs}) \wedge$   
 $(M, Oi, Os) \text{ sat prop (SOME (SLc slCommand))}$

[PlatoonLeader\_slCommand\_lemma]

```

⊢ CFGInterpret (M, Oi, Os)
  (CFG authTestMoveToORP ssmMoveToORPStateInterp
   (secContextMoveToORP slCommand)
   (Name PlatoonLeader says prop (SOME (SLc slCommand)))::
    ins) s outs) ⇒
  (M, Oi, Os) sat prop (SOME (SLc slCommand))

```

## 12 MoveToORPType Theory

**Built:** 10 June 2018

**Parent Theories:** indexedLists, patternMatches

### 12.1 Datatypes

```

slCommand = pltForm | pltMove | pltSecureHalt | complete
           | incomplete

```

```

slOutput = MoveToORP | PLTForm | PLTMove | PLTSecureHalt
           | Complete | unauthorized | unAuthenticated

```

```

slState = MOVE_TO_ORP | PLT_FORM | PLT_MOVE | PLT_SECURE_HALT
          | COMPLETE

```

```

stateRole = PlatoonLeader

```

### 12.2 Theorems

[slCommand\_distinct\_clauses]

```

⊢ pltForm ≠ pltMove ∧ pltForm ≠ pltSecureHalt ∧
  pltForm ≠ complete ∧ pltForm ≠ incomplete ∧
  pltMove ≠ pltSecureHalt ∧ pltMove ≠ complete ∧
  pltMove ≠ incomplete ∧ pltSecureHalt ≠ complete ∧
  pltSecureHalt ≠ incomplete ∧ complete ≠ incomplete

```

[slOutput\_distinct\_clauses]

```

⊢ MoveToORP ≠ PLTForm ∧ MoveToORP ≠ PLTMove ∧
  MoveToORP ≠ PLTSecureHalt ∧ MoveToORP ≠ Complete ∧
  MoveToORP ≠ unauthorized ∧ MoveToORP ≠ unAuthenticated ∧
  PLTForm ≠ PLTMove ∧ PLTForm ≠ PLTSecureHalt ∧
  PLTForm ≠ Complete ∧ PLTForm ≠ unauthorized ∧
  PLTForm ≠ unAuthenticated ∧ PLTMove ≠ PLTSecureHalt ∧
  PLTMove ≠ Complete ∧ PLTMove ≠ unauthorized ∧
  PLTMove ≠ unAuthenticated ∧ PLTSecureHalt ≠ Complete ∧
  PLTSecureHalt ≠ unauthorized ∧
  PLTSecureHalt ≠ unAuthenticated ∧ Complete ≠ unauthorized ∧
  Complete ≠ unAuthenticated ∧ unauthorized ≠ unAuthenticated

```

[slState\_distinct\_clauses]

```

⊢ MOVE_TO_ORP ≠ PLT_FORM ∧ MOVE_TO_ORP ≠ PLT_MOVE ∧
  MOVE_TO_ORP ≠ PLT_SECURE_HALT ∧ MOVE_TO_ORP ≠ COMPLETE ∧
  PLT_FORM ≠ PLT_MOVE ∧ PLT_FORM ≠ PLT_SECURE_HALT ∧
  PLT_FORM ≠ COMPLETE ∧ PLT_MOVE ≠ PLT_SECURE_HALT ∧
  PLT_MOVE ≠ COMPLETE ∧ PLT_SECURE_HALT ≠ COMPLETE

```

## 13 ssmMoveToPB Theory

**Built:** 10 June 2018

**Parent Theories:** MoveToPBType, ssm11, OMNIType

### 13.1 Definitions

[secContextMoveToPB\_def]

```

⊢ ∀ cmd.
  secContextMoveToPB cmd =
    [Name PlatoonLeader controls prop (SOME (SLc cmd))]

```

[ssmMoveToPBStateInterp\_def]

```

⊢ ∀ state. ssmMoveToPBStateInterp state = TT

```

### 13.2 Theorems

[authTestMoveToPB\_cmd\_reject\_lemma]

```

⊢ ∀ cmd. ¬authTestMoveToPB (prop (SOME cmd))

```

[authTestMoveToPB\_def]

```

⊢ (authTestMoveToPB (Name PlatoonLeader says prop cmd) ⇔ T) ∧
  (authTestMoveToPB TT ⇔ F) ∧ (authTestMoveToPB FF ⇔ F) ∧
  (authTestMoveToPB (prop v) ⇔ F) ∧
  (authTestMoveToPB (notf v1) ⇔ F) ∧
  (authTestMoveToPB (v2 andf v3) ⇔ F) ∧
  (authTestMoveToPB (v4 orf v5) ⇔ F) ∧
  (authTestMoveToPB (v6 impf v7) ⇔ F) ∧
  (authTestMoveToPB (v8 eqf v9) ⇔ F) ∧
  (authTestMoveToPB (v10 says TT) ⇔ F) ∧
  (authTestMoveToPB (v10 says FF) ⇔ F) ∧
  (authTestMoveToPB (v133 meet v134 says prop v66) ⇔ F) ∧
  (authTestMoveToPB (v135 quoting v136 says prop v66) ⇔ F) ∧
  (authTestMoveToPB (v10 says notf v67) ⇔ F) ∧
  (authTestMoveToPB (v10 says (v68 andf v69)) ⇔ F) ∧
  (authTestMoveToPB (v10 says (v70 orf v71)) ⇔ F) ∧
  (authTestMoveToPB (v10 says (v72 impf v73)) ⇔ F) ∧
  (authTestMoveToPB (v10 says (v74 eqf v75)) ⇔ F) ∧
  (authTestMoveToPB (v10 says v76 says v77) ⇔ F) ∧

```

$(\text{authTestMoveToPB } (v_{10} \text{ says } v_{78} \text{ speaks\_for } v_{79}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{10} \text{ says } v_{80} \text{ controls } v_{81}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{10} \text{ says reps } v_{82} \ v_{83} \ v_{84}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{10} \text{ says } v_{85} \text{ domi } v_{86}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{10} \text{ says } v_{87} \text{ eqi } v_{88}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{10} \text{ says } v_{89} \text{ doms } v_{90}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{10} \text{ says } v_{91} \text{ eqs } v_{92}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{10} \text{ says } v_{93} \text{ eqn } v_{94}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{10} \text{ says } v_{95} \text{ lte } v_{96}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{10} \text{ says } v_{97} \text{ lt } v_{98}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{12} \text{ speaks\_for } v_{13}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{14} \text{ controls } v_{15}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (\text{reps } v_{16} \ v_{17} \ v_{18}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{19} \text{ domi } v_{20}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{21} \text{ eqi } v_{22}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{23} \text{ doms } v_{24}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{25} \text{ eqs } v_{26}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{27} \text{ eqn } v_{28}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{29} \text{ lte } v_{30}) \iff F) \wedge$   
 $(\text{authTestMoveToPB } (v_{31} \text{ lt } v_{32}) \iff F)$

[authTestMoveToPB\_ind]

$\vdash \forall P.$

$(\forall \text{cmd}. P (\text{Name PlatoonLeader says prop cmd})) \wedge P \text{ TT} \wedge$   
 $P \text{ FF} \wedge (\forall v. P (\text{prop } v)) \wedge (\forall v_1. P (\text{notf } v_1)) \wedge$   
 $(\forall v_2 \ v_3. P (v_2 \text{ andf } v_3)) \wedge (\forall v_4 \ v_5. P (v_4 \text{ orf } v_5)) \wedge$   
 $(\forall v_6 \ v_7. P (v_6 \text{ impf } v_7)) \wedge (\forall v_8 \ v_9. P (v_8 \text{ eqf } v_9)) \wedge$   
 $(\forall v_{10}. P (v_{10} \text{ says TT})) \wedge (\forall v_{10}. P (v_{10} \text{ says FF})) \wedge$   
 $(\forall v_{133} \ v_{134} \ v_{66}. P (v_{133} \text{ meet } v_{134} \text{ says prop } v_{66})) \wedge$   
 $(\forall v_{135} \ v_{136} \ v_{66}. P (v_{135} \text{ quoting } v_{136} \text{ says prop } v_{66})) \wedge$   
 $(\forall v_{10} \ v_{67}. P (v_{10} \text{ says notf } v_{67})) \wedge$   
 $(\forall v_{10} \ v_{68} \ v_{69}. P (v_{10} \text{ says } (v_{68} \text{ andf } v_{69}))) \wedge$   
 $(\forall v_{10} \ v_{70} \ v_{71}. P (v_{10} \text{ says } (v_{70} \text{ orf } v_{71}))) \wedge$   
 $(\forall v_{10} \ v_{72} \ v_{73}. P (v_{10} \text{ says } (v_{72} \text{ impf } v_{73}))) \wedge$   
 $(\forall v_{10} \ v_{74} \ v_{75}. P (v_{10} \text{ says } (v_{74} \text{ eqf } v_{75}))) \wedge$   
 $(\forall v_{10} \ v_{76} \ v_{77}. P (v_{10} \text{ says } v_{76} \text{ says } v_{77})) \wedge$   
 $(\forall v_{10} \ v_{78} \ v_{79}. P (v_{10} \text{ says } v_{78} \text{ speaks\_for } v_{79})) \wedge$   
 $(\forall v_{10} \ v_{80} \ v_{81}. P (v_{10} \text{ says } v_{80} \text{ controls } v_{81})) \wedge$   
 $(\forall v_{10} \ v_{82} \ v_{83} \ v_{84}. P (v_{10} \text{ says reps } v_{82} \ v_{83} \ v_{84})) \wedge$   
 $(\forall v_{10} \ v_{85} \ v_{86}. P (v_{10} \text{ says } v_{85} \text{ domi } v_{86})) \wedge$   
 $(\forall v_{10} \ v_{87} \ v_{88}. P (v_{10} \text{ says } v_{87} \text{ eqi } v_{88})) \wedge$   
 $(\forall v_{10} \ v_{89} \ v_{90}. P (v_{10} \text{ says } v_{89} \text{ doms } v_{90})) \wedge$   
 $(\forall v_{10} \ v_{91} \ v_{92}. P (v_{10} \text{ says } v_{91} \text{ eqs } v_{92})) \wedge$   
 $(\forall v_{10} \ v_{93} \ v_{94}. P (v_{10} \text{ says } v_{93} \text{ eqn } v_{94})) \wedge$   
 $(\forall v_{10} \ v_{95} \ v_{96}. P (v_{10} \text{ says } v_{95} \text{ lte } v_{96})) \wedge$   
 $(\forall v_{10} \ v_{97} \ v_{98}. P (v_{10} \text{ says } v_{97} \text{ lt } v_{98})) \wedge$   
 $(\forall v_{12} \ v_{13}. P (v_{12} \text{ speaks\_for } v_{13})) \wedge$   
 $(\forall v_{14} \ v_{15}. P (v_{14} \text{ controls } v_{15})) \wedge$   
 $(\forall v_{16} \ v_{17} \ v_{18}. P (\text{reps } v_{16} \ v_{17} \ v_{18})) \wedge$



$$\begin{aligned}
& (\forall v_{19} v_{20}. P (v_{19} \text{ domi } v_{20})) \wedge \\
& (\forall v_{21} v_{22}. P (v_{21} \text{ eqi } v_{22})) \wedge \\
& (\forall v_{23} v_{24}. P (v_{23} \text{ doms } v_{24})) \wedge \\
& (\forall v_{25} v_{26}. P (v_{25} \text{ eqs } v_{26})) \wedge (\forall v_{27} v_{28}. P (v_{27} \text{ eqn } v_{28})) \wedge \\
& (\forall v_{29} v_{30}. P (v_{29} \text{ lte } v_{30})) \wedge (\forall v_{31} v_{32}. P (v_{31} \text{ lt } v_{32})) \Rightarrow \\
& \forall v. P v
\end{aligned}$$

[moveToPBNS\_def]

$$\begin{aligned}
& \vdash (\text{moveToPBNS MOVE\_TO\_PB (exec (SLc pltForm))} = \text{PLT\_FORM}) \wedge \\
& (\text{moveToPBNS MOVE\_TO\_PB (exec (SLc incomplete))} = \\
& \text{MOVE\_TO\_PB}) \wedge \\
& (\text{moveToPBNS PLT\_FORM (exec (SLc pltMove))} = \text{PLT\_MOVE}) \wedge \\
& (\text{moveToPBNS PLT\_FORM (exec (SLc incomplete))} = \text{PLT\_FORM}) \wedge \\
& (\text{moveToPBNS PLT\_MOVE (exec (SLc pltHalt))} = \text{PLT\_HALT}) \wedge \\
& (\text{moveToPBNS PLT\_MOVE (exec (SLc incomplete))} = \text{PLT\_MOVE}) \wedge \\
& (\text{moveToPBNS PLT\_HALT (exec (SLc complete))} = \text{COMPLETE}) \wedge \\
& (\text{moveToPBNS PLT\_HALT (exec (SLc incomplete))} = \text{PLT\_HALT}) \wedge \\
& (\text{moveToPBNS } s \text{ (trap (SLc cmd))} = s) \wedge \\
& (\text{moveToPBNS } s \text{ (discard (SLc cmd))} = s)
\end{aligned}$$

[moveToPBNS\_ind]

$$\begin{aligned}
& \vdash \forall P. \\
& P \text{ MOVE\_TO\_PB (exec (SLc pltForm))} \wedge \\
& P \text{ MOVE\_TO\_PB (exec (SLc incomplete))} \wedge \\
& P \text{ PLT\_FORM (exec (SLc pltMove))} \wedge \\
& P \text{ PLT\_FORM (exec (SLc incomplete))} \wedge \\
& P \text{ PLT\_MOVE (exec (SLc pltHalt))} \wedge \\
& P \text{ PLT\_MOVE (exec (SLc incomplete))} \wedge \\
& P \text{ PLT\_HALT (exec (SLc complete))} \wedge \\
& P \text{ PLT\_HALT (exec (SLc incomplete))} \wedge \\
& (\forall s \text{ cmd}. P s \text{ (trap (SLc cmd))}) \wedge \\
& (\forall s \text{ cmd}. P s \text{ (discard (SLc cmd))}) \wedge \\
& (\forall s v_6. P s \text{ (discard (ESCc } v_6 \text{))}) \wedge \\
& (\forall s v_9. P s \text{ (trap (ESCc } v_9 \text{))}) \wedge \\
& (\forall v_{12}. P \text{ MOVE\_TO\_PB (exec (ESCc } v_{12} \text{))}) \wedge \\
& P \text{ MOVE\_TO\_PB (exec (SLc pltMove))} \wedge \\
& P \text{ MOVE\_TO\_PB (exec (SLc pltHalt))} \wedge \\
& P \text{ MOVE\_TO\_PB (exec (SLc complete))} \wedge \\
& (\forall v_{15}. P \text{ PLT\_FORM (exec (ESCc } v_{15} \text{))}) \wedge \\
& P \text{ PLT\_FORM (exec (SLc pltForm))} \wedge \\
& P \text{ PLT\_FORM (exec (SLc pltHalt))} \wedge \\
& P \text{ PLT\_FORM (exec (SLc complete))} \wedge \\
& (\forall v_{18}. P \text{ PLT\_MOVE (exec (ESCc } v_{18} \text{))}) \wedge \\
& P \text{ PLT\_MOVE (exec (SLc pltForm))} \wedge \\
& P \text{ PLT\_MOVE (exec (SLc pltMove))} \wedge \\
& P \text{ PLT\_MOVE (exec (SLc complete))} \wedge \\
& (\forall v_{21}. P \text{ PLT\_HALT (exec (ESCc } v_{21} \text{))}) \wedge \\
& P \text{ PLT\_HALT (exec (SLc pltForm))} \wedge \\
& P \text{ PLT\_HALT (exec (SLc pltMove))} \wedge
\end{aligned}$$

$$\begin{aligned}
& P \text{ PLT\_HALT } (\text{exec } (\text{SLc pltHalt})) \wedge \\
& (\forall v_{23}. P \text{ COMPLETE } (\text{exec } v_{23})) \Rightarrow \\
& \forall v \ v_1. P \ v \ v_1
\end{aligned}$$

[moveToPBOut\_def]

$$\begin{aligned}
& \vdash (\text{moveToPBOut MOVE\_TO\_PB } (\text{exec } (\text{SLc pltForm})) = \text{PLTForm}) \wedge \\
& (\text{moveToPBOut MOVE\_TO\_PB } (\text{exec } (\text{SLc incomplete})) = \text{MoveToPB}) \wedge \\
& (\text{moveToPBOut PLT\_FORM } (\text{exec } (\text{SLc pltMove})) = \text{PLTMove}) \wedge \\
& (\text{moveToPBOut PLT\_FORM } (\text{exec } (\text{SLc incomplete})) = \text{PLTForm}) \wedge \\
& (\text{moveToPBOut PLT\_MOVE } (\text{exec } (\text{SLc pltHalt})) = \text{PLTHalt}) \wedge \\
& (\text{moveToPBOut PLT\_MOVE } (\text{exec } (\text{SLc incomplete})) = \text{PLTMove}) \wedge \\
& (\text{moveToPBOut PLT\_HALT } (\text{exec } (\text{SLc complete})) = \text{Complete}) \wedge \\
& (\text{moveToPBOut PLT\_HALT } (\text{exec } (\text{SLc incomplete})) = \text{PLTHalt}) \wedge \\
& (\text{moveToPBOut } s \ (\text{trap } (\text{SLc cmd})) = \text{unAuthorized}) \wedge \\
& (\text{moveToPBOut } s \ (\text{discard } (\text{SLc cmd})) = \text{unAuthenticated})
\end{aligned}$$

[moveToPBOut\_ind]

$$\begin{aligned}
& \vdash \forall P. \\
& P \text{ MOVE\_TO\_PB } (\text{exec } (\text{SLc pltForm})) \wedge \\
& P \text{ MOVE\_TO\_PB } (\text{exec } (\text{SLc incomplete})) \wedge \\
& P \text{ PLT\_FORM } (\text{exec } (\text{SLc pltMove})) \wedge \\
& P \text{ PLT\_FORM } (\text{exec } (\text{SLc incomplete})) \wedge \\
& P \text{ PLT\_MOVE } (\text{exec } (\text{SLc pltHalt})) \wedge \\
& P \text{ PLT\_MOVE } (\text{exec } (\text{SLc incomplete})) \wedge \\
& P \text{ PLT\_HALT } (\text{exec } (\text{SLc complete})) \wedge \\
& P \text{ PLT\_HALT } (\text{exec } (\text{SLc incomplete})) \wedge \\
& (\forall s \ \text{cmd}. P \ s \ (\text{trap } (\text{SLc cmd}))) \wedge \\
& (\forall s \ \text{cmd}. P \ s \ (\text{discard } (\text{SLc cmd}))) \wedge \\
& (\forall s \ v_6. P \ s \ (\text{discard } (\text{ESCc } v_6))) \wedge \\
& (\forall s \ v_9. P \ s \ (\text{trap } (\text{ESCc } v_9))) \wedge \\
& (\forall v_{12}. P \ \text{MOVE\_TO\_PB } (\text{exec } (\text{ESCc } v_{12}))) \wedge \\
& P \text{ MOVE\_TO\_PB } (\text{exec } (\text{SLc pltMove})) \wedge \\
& P \text{ MOVE\_TO\_PB } (\text{exec } (\text{SLc pltHalt})) \wedge \\
& P \text{ MOVE\_TO\_PB } (\text{exec } (\text{SLc complete})) \wedge \\
& (\forall v_{15}. P \ \text{PLT\_FORM } (\text{exec } (\text{ESCc } v_{15}))) \wedge \\
& P \text{ PLT\_FORM } (\text{exec } (\text{SLc pltForm})) \wedge \\
& P \text{ PLT\_FORM } (\text{exec } (\text{SLc pltHalt})) \wedge \\
& P \text{ PLT\_FORM } (\text{exec } (\text{SLc complete})) \wedge \\
& (\forall v_{18}. P \ \text{PLT\_MOVE } (\text{exec } (\text{ESCc } v_{18}))) \wedge \\
& P \text{ PLT\_MOVE } (\text{exec } (\text{SLc pltForm})) \wedge \\
& P \text{ PLT\_MOVE } (\text{exec } (\text{SLc pltMove})) \wedge \\
& P \text{ PLT\_MOVE } (\text{exec } (\text{SLc complete})) \wedge \\
& (\forall v_{21}. P \ \text{PLT\_HALT } (\text{exec } (\text{ESCc } v_{21}))) \wedge \\
& P \text{ PLT\_HALT } (\text{exec } (\text{SLc pltForm})) \wedge \\
& P \text{ PLT\_HALT } (\text{exec } (\text{SLc pltMove})) \wedge \\
& P \text{ PLT\_HALT } (\text{exec } (\text{SLc pltHalt})) \wedge \\
& (\forall v_{23}. P \ \text{COMPLETE } (\text{exec } v_{23})) \Rightarrow \\
& \forall v \ v_1. P \ v \ v_1
\end{aligned}$$

[PlatoonLeader\_exec\_slCommand\_justified\_thm]

```

⊢ ∀ NS Out M Oi Os.
  TR (M, Oi, Os) (exec (SLc slCommand))
    (CFG authTestMoveToPB ssmMoveToPBStateInterp
      (secContextMoveToPB slCommand)
      (Name PlatoonLeader says prop (SOME (SLc slCommand)))::
        ins) s outs)
    (CFG authTestMoveToPB ssmMoveToPBStateInterp
      (secContextMoveToPB slCommand) ins
      (NS s (exec (SLc slCommand))))
    (Out s (exec (SLc slCommand)))::outs) ⇔
authTestMoveToPB
  (Name PlatoonLeader says prop (SOME (SLc slCommand))) ∧
CFGInterpret (M, Oi, Os)
  (CFG authTestMoveToPB ssmMoveToPBStateInterp
    (secContextMoveToPB slCommand)
    (Name PlatoonLeader says prop (SOME (SLc slCommand)))::
      ins) s outs) ⇒
  (M, Oi, Os) sat prop (SOME (SLc slCommand))

```

[PlatoonLeader\_slCommand\_lemma]

```

⊢ CFGInterpret (M, Oi, Os)
  (CFG authTestMoveToPB ssmMoveToPBStateInterp
    (secContextMoveToPB slCommand)
    (Name PlatoonLeader says prop (SOME (SLc slCommand)))::
      ins) s outs) ⇒
  (M, Oi, Os) sat prop (SOME (SLc slCommand))

```

## 14 MoveToPBType Theory

**Built:** 10 June 2018

**Parent Theories:** indexedLists, patternMatches

### 14.1 Datatypes

*slCommand* = pltForm | pltMove | pltHalt | complete | incomplete

*slOutput* = MoveToPB | PLTForm | PLTMove | PLTHalt | Complete  
 | unauthorized | unAuthenticated

*slState* = MOVE\_TO\_PB | PLT\_FORM | PLT\_MOVE | PLT\_HALT | COMPLETE

*stateRole* = PlatoonLeader

### 14.2 Theorems

**[slCommand\_distinct\_clauses]**

$\vdash \text{pltForm} \neq \text{pltMove} \wedge \text{pltForm} \neq \text{pltHalt} \wedge \text{pltForm} \neq \text{complete} \wedge$   
 $\text{pltForm} \neq \text{incomplete} \wedge \text{pltMove} \neq \text{pltHalt} \wedge$   
 $\text{pltMove} \neq \text{complete} \wedge \text{pltMove} \neq \text{incomplete} \wedge$   
 $\text{pltHalt} \neq \text{complete} \wedge \text{pltHalt} \neq \text{incomplete} \wedge$   
 $\text{complete} \neq \text{incomplete}$

**[slOutput\_distinct\_clauses]**

$\vdash \text{MoveToPB} \neq \text{PLTForm} \wedge \text{MoveToPB} \neq \text{PLTMove} \wedge$   
 $\text{MoveToPB} \neq \text{PLTHalt} \wedge \text{MoveToPB} \neq \text{Complete} \wedge$   
 $\text{MoveToPB} \neq \text{unAuthorized} \wedge \text{MoveToPB} \neq \text{unAuthenticated} \wedge$   
 $\text{PLTForm} \neq \text{PLTMove} \wedge \text{PLTForm} \neq \text{PLTHalt} \wedge \text{PLTForm} \neq \text{Complete} \wedge$   
 $\text{PLTForm} \neq \text{unAuthorized} \wedge \text{PLTForm} \neq \text{unAuthenticated} \wedge$   
 $\text{PLTMove} \neq \text{PLTHalt} \wedge \text{PLTMove} \neq \text{Complete} \wedge$   
 $\text{PLTMove} \neq \text{unAuthorized} \wedge \text{PLTMove} \neq \text{unAuthenticated} \wedge$   
 $\text{PLTHalt} \neq \text{Complete} \wedge \text{PLTHalt} \neq \text{unAuthorized} \wedge$   
 $\text{PLTHalt} \neq \text{unAuthenticated} \wedge \text{Complete} \neq \text{unAuthorized} \wedge$   
 $\text{Complete} \neq \text{unAuthenticated} \wedge \text{unAuthorized} \neq \text{unAuthenticated}$

**[slState\_distinct\_clauses]**

$\vdash \text{MOVE\_TO\_PB} \neq \text{PLT\_FORM} \wedge \text{MOVE\_TO\_PB} \neq \text{PLT\_MOVE} \wedge$   
 $\text{MOVE\_TO\_PB} \neq \text{PLT\_HALT} \wedge \text{MOVE\_TO\_PB} \neq \text{COMPLETE} \wedge$   
 $\text{PLT\_FORM} \neq \text{PLT\_MOVE} \wedge \text{PLT\_FORM} \neq \text{PLT\_HALT} \wedge$   
 $\text{PLT\_FORM} \neq \text{COMPLETE} \wedge \text{PLT\_MOVE} \neq \text{PLT\_HALT} \wedge$   
 $\text{PLT\_MOVE} \neq \text{COMPLETE} \wedge \text{PLT\_HALT} \neq \text{COMPLETE}$

## 15 ssmPlanPB Theory

**Built:** 10 June 2018

**Parent Theories:** PlanPBDef, ssm

### 15.1 Theorems

**[inputOK\_def]**

$\vdash (\text{inputOK } (\text{Name PlatoonLeader says prop } cmd) \iff T) \wedge$   
 $(\text{inputOK } (\text{Name PlatoonSergeant says prop } cmd) \iff T) \wedge$   
 $(\text{inputOK } TT \iff F) \wedge (\text{inputOK } FF \iff F) \wedge$   
 $(\text{inputOK } (\text{prop } v) \iff F) \wedge (\text{inputOK } (\text{notf } v_1) \iff F) \wedge$   
 $(\text{inputOK } (v_2 \text{ andf } v_3) \iff F) \wedge (\text{inputOK } (v_4 \text{ orf } v_5) \iff F) \wedge$   
 $(\text{inputOK } (v_6 \text{ impf } v_7) \iff F) \wedge (\text{inputOK } (v_8 \text{ eqf } v_9) \iff F) \wedge$   
 $(\text{inputOK } (v_{10} \text{ says } TT) \iff F) \wedge (\text{inputOK } (v_{10} \text{ says } FF) \iff F) \wedge$   
 $(\text{inputOK } (v_{133} \text{ meet } v_{134} \text{ says prop } v_{66}) \iff F) \wedge$   
 $(\text{inputOK } (v_{135} \text{ quoting } v_{136} \text{ says prop } v_{66}) \iff F) \wedge$   
 $(\text{inputOK } (v_{10} \text{ says notf } v_{67}) \iff F) \wedge$   
 $(\text{inputOK } (v_{10} \text{ says } (v_{68} \text{ andf } v_{69})) \iff F) \wedge$   
 $(\text{inputOK } (v_{10} \text{ says } (v_{70} \text{ orf } v_{71})) \iff F) \wedge$

$(\text{inputOK } (v_{10} \text{ says } (v_{72} \text{ impf } v_{73})) \iff F) \wedge$   
 $(\text{inputOK } (v_{10} \text{ says } (v_{74} \text{ eqf } v_{75})) \iff F) \wedge$   
 $(\text{inputOK } (v_{10} \text{ says } v_{76} \text{ says } v_{77}) \iff F) \wedge$   
 $(\text{inputOK } (v_{10} \text{ says } v_{78} \text{ speaks\_for } v_{79}) \iff F) \wedge$   
 $(\text{inputOK } (v_{10} \text{ says } v_{80} \text{ controls } v_{81}) \iff F) \wedge$   
 $(\text{inputOK } (v_{10} \text{ says reps } v_{82} \ v_{83} \ v_{84}) \iff F) \wedge$   
 $(\text{inputOK } (v_{10} \text{ says } v_{85} \text{ domi } v_{86}) \iff F) \wedge$   
 $(\text{inputOK } (v_{10} \text{ says } v_{87} \text{ eqi } v_{88}) \iff F) \wedge$   
 $(\text{inputOK } (v_{10} \text{ says } v_{89} \text{ doms } v_{90}) \iff F) \wedge$   
 $(\text{inputOK } (v_{10} \text{ says } v_{91} \text{ eqs } v_{92}) \iff F) \wedge$   
 $(\text{inputOK } (v_{10} \text{ says } v_{93} \text{ eqn } v_{94}) \iff F) \wedge$   
 $(\text{inputOK } (v_{10} \text{ says } v_{95} \text{ lte } v_{96}) \iff F) \wedge$   
 $(\text{inputOK } (v_{10} \text{ says } v_{97} \text{ lt } v_{98}) \iff F) \wedge$   
 $(\text{inputOK } (v_{12} \text{ speaks\_for } v_{13}) \iff F) \wedge$   
 $(\text{inputOK } (v_{14} \text{ controls } v_{15}) \iff F) \wedge$   
 $(\text{inputOK } (\text{reps } v_{16} \ v_{17} \ v_{18}) \iff F) \wedge$   
 $(\text{inputOK } (v_{19} \text{ domi } v_{20}) \iff F) \wedge$   
 $(\text{inputOK } (v_{21} \text{ eqi } v_{22}) \iff F) \wedge$   
 $(\text{inputOK } (v_{23} \text{ doms } v_{24}) \iff F) \wedge$   
 $(\text{inputOK } (v_{25} \text{ eqs } v_{26}) \iff F) \wedge (\text{inputOK } (v_{27} \text{ eqn } v_{28}) \iff F) \wedge$   
 $(\text{inputOK } (v_{29} \text{ lte } v_{30}) \iff F) \wedge (\text{inputOK } (v_{31} \text{ lt } v_{32}) \iff F)$

[inputOK\_ind]

$\vdash \forall P.$

$(\forall \text{cmd}. P (\text{Name PlatoonLeader says prop cmd})) \wedge$   
 $(\forall \text{cmd}. P (\text{Name PlatoonSergeant says prop cmd})) \wedge P \text{ TT} \wedge$   
 $P \text{ FF} \wedge (\forall v. P (\text{prop } v)) \wedge (\forall v_1. P (\text{notf } v_1)) \wedge$   
 $(\forall v_2 \ v_3. P (v_2 \text{ andf } v_3)) \wedge (\forall v_4 \ v_5. P (v_4 \text{ orf } v_5)) \wedge$   
 $(\forall v_6 \ v_7. P (v_6 \text{ impf } v_7)) \wedge (\forall v_8 \ v_9. P (v_8 \text{ eqf } v_9)) \wedge$   
 $(\forall v_{10}. P (v_{10} \text{ says TT})) \wedge (\forall v_{10}. P (v_{10} \text{ says FF})) \wedge$   
 $(\forall v_{133} \ v_{134} \ v_{66}. P (v_{133} \text{ meet } v_{134} \text{ says prop } v_{66})) \wedge$   
 $(\forall v_{135} \ v_{136} \ v_{66}. P (v_{135} \text{ quoting } v_{136} \text{ says prop } v_{66})) \wedge$   
 $(\forall v_{10} \ v_{67}. P (v_{10} \text{ says notf } v_{67})) \wedge$   
 $(\forall v_{10} \ v_{68} \ v_{69}. P (v_{10} \text{ says } (v_{68} \text{ andf } v_{69}))) \wedge$   
 $(\forall v_{10} \ v_{70} \ v_{71}. P (v_{10} \text{ says } (v_{70} \text{ orf } v_{71}))) \wedge$   
 $(\forall v_{10} \ v_{72} \ v_{73}. P (v_{10} \text{ says } (v_{72} \text{ impf } v_{73}))) \wedge$   
 $(\forall v_{10} \ v_{74} \ v_{75}. P (v_{10} \text{ says } (v_{74} \text{ eqf } v_{75}))) \wedge$   
 $(\forall v_{10} \ v_{76} \ v_{77}. P (v_{10} \text{ says } v_{76} \text{ says } v_{77})) \wedge$   
 $(\forall v_{10} \ v_{78} \ v_{79}. P (v_{10} \text{ says } v_{78} \text{ speaks\_for } v_{79})) \wedge$   
 $(\forall v_{10} \ v_{80} \ v_{81}. P (v_{10} \text{ says } v_{80} \text{ controls } v_{81})) \wedge$   
 $(\forall v_{10} \ v_{82} \ v_{83} \ v_{84}. P (v_{10} \text{ says reps } v_{82} \ v_{83} \ v_{84})) \wedge$   
 $(\forall v_{10} \ v_{85} \ v_{86}. P (v_{10} \text{ says } v_{85} \text{ domi } v_{86})) \wedge$   
 $(\forall v_{10} \ v_{87} \ v_{88}. P (v_{10} \text{ says } v_{87} \text{ eqi } v_{88})) \wedge$   
 $(\forall v_{10} \ v_{89} \ v_{90}. P (v_{10} \text{ says } v_{89} \text{ doms } v_{90})) \wedge$   
 $(\forall v_{10} \ v_{91} \ v_{92}. P (v_{10} \text{ says } v_{91} \text{ eqs } v_{92})) \wedge$   
 $(\forall v_{10} \ v_{93} \ v_{94}. P (v_{10} \text{ says } v_{93} \text{ eqn } v_{94})) \wedge$   
 $(\forall v_{10} \ v_{95} \ v_{96}. P (v_{10} \text{ says } v_{95} \text{ lte } v_{96})) \wedge$   
 $(\forall v_{10} \ v_{97} \ v_{98}. P (v_{10} \text{ says } v_{97} \text{ lt } v_{98})) \wedge$   
 $(\forall v_{12} \ v_{13}. P (v_{12} \text{ speaks\_for } v_{13})) \wedge$

$$\begin{aligned}
& (\forall v_{14} v_{15}. P (v_{14} \text{ controls } v_{15})) \wedge \\
& (\forall v_{16} v_{17} v_{18}. P (\text{reps } v_{16} v_{17} v_{18})) \wedge \\
& (\forall v_{19} v_{20}. P (v_{19} \text{ domi } v_{20})) \wedge \\
& (\forall v_{21} v_{22}. P (v_{21} \text{ eqi } v_{22})) \wedge \\
& (\forall v_{23} v_{24}. P (v_{23} \text{ doms } v_{24})) \wedge \\
& (\forall v_{25} v_{26}. P (v_{25} \text{ eqs } v_{26})) \wedge (\forall v_{27} v_{28}. P (v_{27} \text{ eqn } v_{28})) \wedge \\
& (\forall v_{29} v_{30}. P (v_{29} \text{ lte } v_{30})) \wedge (\forall v_{31} v_{32}. P (v_{31} \text{ lt } v_{32})) \Rightarrow \\
& \forall v. P v
\end{aligned}$$

[planPBNS\_def]

$$\begin{aligned}
& \vdash (\text{planPBNS WARNO (exec } x) = \\
& \quad \text{if} \\
& \quad \quad (\text{getRecon } x = [\text{SOME (SLc (PL recon))}]) \wedge \\
& \quad \quad (\text{getTentativePlan } x = [\text{SOME (SLc (PL tentativePlan))}]) \wedge \\
& \quad \quad (\text{getReport } x = [\text{SOME (SLc (PL report1))}]) \wedge \\
& \quad \quad (\text{getInitMove } x = [\text{SOME (SLc (PSG initiateMovement))}]) \\
& \quad \text{then} \\
& \quad \quad \text{REPORT1} \\
& \quad \text{else WARNO}) \wedge \\
& (\text{planPBNS PLAN\_PB (exec } x) = \\
& \quad \text{if getPlCom } x = \text{receiveMission then RECEIVE\_MISSION} \\
& \quad \text{else PLAN\_PB}) \wedge \\
& (\text{planPBNS RECEIVE\_MISSION (exec } x) = \\
& \quad \text{if getPlCom } x = \text{warno then WARNO else RECEIVE\_MISSION}) \wedge \\
& (\text{planPBNS REPORT1 (exec } x) = \\
& \quad \text{if getPlCom } x = \text{completePlan then COMPLETE\_PLAN} \\
& \quad \text{else REPORT1}) \wedge \\
& (\text{planPBNS COMPLETE\_PLAN (exec } x) = \\
& \quad \text{if getPlCom } x = \text{opoid then OPOID else COMPLETE\_PLAN}) \wedge \\
& (\text{planPBNS OPOID (exec } x) = \\
& \quad \text{if getPlCom } x = \text{supervise then SUPERVISE else OPOID}) \wedge \\
& (\text{planPBNS SUPERVISE (exec } x) = \\
& \quad \text{if getPlCom } x = \text{report2 then REPORT2 else SUPERVISE}) \wedge \\
& (\text{planPBNS REPORT2 (exec } x) = \\
& \quad \text{if getPlCom } x = \text{complete then COMPLETE else REPORT2}) \wedge \\
& (\text{planPBNS } s (\text{trap } v_0) = s) \wedge (\text{planPBNS } s (\text{discard } v_1) = s)
\end{aligned}$$

[planPBNS\_ind]

$$\begin{aligned}
& \vdash \forall P. \\
& \quad (\forall x. P \text{ WARNO (exec } x)) \wedge (\forall x. P \text{ PLAN\_PB (exec } x)) \wedge \\
& \quad (\forall x. P \text{ RECEIVE\_MISSION (exec } x)) \wedge \\
& \quad (\forall x. P \text{ REPORT1 (exec } x)) \wedge (\forall x. P \text{ COMPLETE\_PLAN (exec } x)) \wedge \\
& \quad (\forall x. P \text{ OPOID (exec } x)) \wedge (\forall x. P \text{ SUPERVISE (exec } x)) \wedge \\
& \quad (\forall x. P \text{ REPORT2 (exec } x)) \wedge (\forall s v_0. P s (\text{trap } v_0)) \wedge \\
& \quad (\forall s v_1. P s (\text{discard } v_1)) \wedge \\
& \quad (\forall v_6. P \text{ TENTATIVE\_PLAN (exec } v_6)) \wedge \\
& \quad (\forall v_7. P \text{ INITIATE\_MOVEMENT (exec } v_7)) \wedge \\
& \quad (\forall v_8. P \text{ RECON (exec } v_8)) \wedge (\forall v_9. P \text{ COMPLETE (exec } v_9)) \Rightarrow \\
& \quad \forall v v_1. P v v_1
\end{aligned}$$

**[planPBOut\_def]**

```

⊢ (planPBOut WARNO (exec x) =
  if
    (getRecon x = [SOME (SLc (PL recon))]) ∧
    (getTentativePlan x = [SOME (SLc (PL tentativePlan))]) ∧
    (getReport x = [SOME (SLc (PL report1))]) ∧
    (getInitMove x = [SOME (SLc (PSG initiateMovement))])
  then
    Report1
  else unauthorized) ∧
(planPBOut PLAN_PB (exec x) =
  if getPlCom x = receiveMission then ReceiveMission
  else unauthorized) ∧
(planPBOut RECEIVE_MISSION (exec x) =
  if getPlCom x = warno then Warno else unauthorized) ∧
(planPBOut REPORT1 (exec x) =
  if getPlCom x = completePlan then CompletePlan
  else unauthorized) ∧
(planPBOut COMPLETE_PLAN (exec x) =
  if getPlCom x = opoid then Opoid else unauthorized) ∧
(planPBOut OPOID (exec x) =
  if getPlCom x = supervise then Supervise
  else unauthorized) ∧
(planPBOut SUPERVISE (exec x) =
  if getPlCom x = report2 then Report2 else unauthorized) ∧
(planPBOut REPORT2 (exec x) =
  if getPlCom x = complete then Complete else unauthorized) ∧
(planPBOut s (trap v0) = unauthorized) ∧
(planPBOut s (discard v1) = unAuthenticated)

```

**[planPBOut\_ind]**

```

⊢ ∀ P.
  (∀ x. P WARNO (exec x)) ∧ (∀ x. P PLAN_PB (exec x)) ∧
  (∀ x. P RECEIVE_MISSION (exec x)) ∧
  (∀ x. P REPORT1 (exec x)) ∧ (∀ x. P COMPLETE_PLAN (exec x)) ∧
  (∀ x. P OPOID (exec x)) ∧ (∀ x. P SUPERVISE (exec x)) ∧
  (∀ x. P REPORT2 (exec x)) ∧ (∀ s v0. P s (trap v0)) ∧
  (∀ s v1. P s (discard v1)) ∧
  (∀ v6. P TENTATIVE_PLAN (exec v6)) ∧
  (∀ v7. P INITIATE_MOVEMENT (exec v7)) ∧
  (∀ v8. P RECON (exec v8)) ∧ (∀ v9. P COMPLETE (exec v9)) ⇒
  ∀ v v1. P v v1

```

**[PlatoonLeader\_notWARNO\_notreport1\_exec\_plCommand\_justified\_lemma]**

```

⊢ s ≠ WARNO ⇒
  plCommand ≠ invalidPlCommand ⇒
  plCommand ≠ report1 ⇒
  ∀ NS Out M Oi Os.

```

```

TR (M, Oi, Os)
  (exec
    (inputList
      [Name PlatoonLeader says
        prop (SOME (SLc (PL plCommand))))]))
  (CFG inputOK secContext secContextNull
    ([Name PlatoonLeader says
      prop (SOME (SLc (PL plCommand)))]::ins) s outs)
  (CFG inputOK secContext secContextNull ins
    (NS s
      (exec
        (inputList
          [Name PlatoonLeader says
            prop (SOME (SLc (PL plCommand))))]))
    (Out s
      (exec
        (inputList
          [Name PlatoonLeader says
            prop (SOME (SLc (PL plCommand)))]))::
        outs))  $\iff$ 
  authenticationTest inputOK
    [Name PlatoonLeader says
      prop (SOME (SLc (PL plCommand)))]  $\wedge$ 
  CFGInterpret (M, Oi, Os)
    (CFG inputOK secContext secContextNull
      ([Name PlatoonLeader says
        prop (SOME (SLc (PL plCommand)))]::ins) s outs)  $\wedge$ 
  (M, Oi, Os) satList
  propCommandList
    [Name PlatoonLeader says
      prop (SOME (SLc (PL plCommand)))]

```

[PlatoonLeader\_notWARNO\_notreport1\_exec\_plCommand\_justified\_thm]

```

 $\vdash s \neq \text{WARNO} \Rightarrow$ 
 $plCommand \neq \text{invalidPlCommand} \Rightarrow$ 
 $plCommand \neq \text{report1} \Rightarrow$ 
 $\forall NS \text{ Out } M \text{ Oi } Os.$ 
  TR (M, Oi, Os) (exec [SOME (SLc (PL plCommand))])
    (CFG inputOK secContext secContextNull
      ([Name PlatoonLeader says
        prop (SOME (SLc (PL plCommand)))]::ins) s outs)
    (CFG inputOK secContext secContextNull ins
      (NS s (exec [SOME (SLc (PL plCommand))]))
      (Out s (exec [SOME (SLc (PL plCommand)))]::outs))  $\iff$ 
  authenticationTest inputOK
    [Name PlatoonLeader says
      prop (SOME (SLc (PL plCommand)))]  $\wedge$ 
  CFGInterpret (M, Oi, Os)
    (CFG inputOK secContext secContextNull

```



$$([Name\ PlatoonLeader\ says \\ \text{prop}\ (SOME\ (SLc\ (PL\ plCommand))))]::ins)\ s\ outs) \wedge \\ (M, Oi, Os)\ satList\ [\text{prop}\ (SOME\ (SLc\ (PL\ plCommand)))]$$

[PlatoonLeader\_notWARNO\_notreport1\_exec\_plCommand\_lemma]

$$\vdash s \neq WARNO \Rightarrow \\ plCommand \neq invalidPlCommand \Rightarrow \\ plCommand \neq report1 \Rightarrow \\ \forall M\ Oi\ Os. \\ CFGInterpret\ (M, Oi, Os) \\ (CFG\ inputOK\ secContext\ secContextNull \\ ([Name\ PlatoonLeader\ says \\ \text{prop}\ (SOME\ (SLc\ (PL\ plCommand))))]::ins)\ s\ outs) \Rightarrow \\ (M, Oi, Os)\ satList \\ propCommandList \\ [Name\ PlatoonLeader\ says \\ \text{prop}\ (SOME\ (SLc\ (PL\ plCommand)))]$$

[PlatoonLeader\_psgCommand\_notDiscard\_thm]

$$\vdash \forall NS\ Out\ M\ Oi\ Os. \\ \neg TR\ (M, Oi, Os) \\ (discard \\ (inputList \\ [Name\ PlatoonLeader\ says \\ \text{prop}\ (SOME\ (SLc\ (PSG\ psgCommand)))))) \\ (CFG\ inputOK\ secContext\ secContextNull \\ ([Name\ PlatoonLeader\ says \\ \text{prop}\ (SOME\ (SLc\ (PSG\ psgCommand))))]::ins)\ s\ outs) \\ (CFG\ inputOK\ secContext\ secContextNull\ ins \\ (NS\ s \\ (discard \\ (inputList \\ [Name\ PlatoonLeader\ says \\ \text{prop}\ (SOME\ (SLc\ (PSG\ psgCommand))))))) \\ (Out\ s \\ (discard \\ (inputList \\ [Name\ PlatoonLeader\ says \\ \text{prop}\ (SOME\ (SLc\ (PSG\ psgCommand))))]::outs)))$$

[PlatoonLeader\_trap\_psgCommand\_justified\_lemma]

$$\vdash \forall NS\ Out\ M\ Oi\ Os. \\ TR\ (M, Oi, Os) \\ (trap \\ (inputList \\ [Name\ PlatoonLeader\ says \\ \text{prop}\ (SOME\ (SLc\ (PSG\ psgCommand))))))$$

```

(CFG inputOK secContext secContextNull
  ([Name PlatoonLeader says
    prop (SOME (SLc (PSG psgCommand))))]::ins) s outs)
(CFG inputOK secContext secContextNull ins
  (NS s
    (trap
      (inputList
        [Name PlatoonLeader says
          prop (SOME (SLc (PSG psgCommand))))]))
  (Out s
    (trap
      (inputList
        [Name PlatoonLeader says
          prop (SOME (SLc (PSG psgCommand))))]::
        outs)))  $\iff$ 
authenticationTest inputOK
  [Name PlatoonLeader says
    prop (SOME (SLc (PSG psgCommand)))]  $\wedge$ 
CFGInterpret (M, Oi, Os)
  (CFG inputOK secContext secContextNull
    ([Name PlatoonLeader says
      prop (SOME (SLc (PSG psgCommand))))]::ins) s outs)  $\wedge$ 
  (M, Oi, Os) sat prop NONE

```

[PlatoonLeader\_trap\_psgCommand\_lemma]

```

 $\vdash \forall M \text{ } Oi \text{ } Os.$ 
CFGInterpret (M, Oi, Os)
  (CFG inputOK secContext secContextNull
    ([Name PlatoonLeader says
      prop (SOME (SLc (PSG psgCommand))))]::ins) s outs)  $\Rightarrow$ 
  (M, Oi, Os) sat prop NONE

```

[PlatoonLeader\_WARNO\_exec\_report1\_justified\_lemma]

```

 $\vdash \forall NS \text{ } Out \text{ } M \text{ } Oi \text{ } Os.$ 
TR (M, Oi, Os)
  (exec
    (inputList
      [Name PlatoonLeader says
        prop (SOME (SLc (PL recon)));
        Name PlatoonLeader says
        prop (SOME (SLc (PL tentativePlan)));
        Name PlatoonSergeant says
        prop (SOME (SLc (PSG initiateMovement)));
        Name PlatoonLeader says
        prop (SOME (SLc (PL report1)))]))
  (CFG inputOK secContext secContextNull
    ([Name PlatoonLeader says
      prop (SOME (SLc (PL recon)));
      Name PlatoonLeader says

```

```

    prop (SOME (SLc (PL tentativePlan)));
    Name PlatoonSergeant says
    prop (SOME (SLc (PSG initiateMovement)));
    Name PlatoonLeader says
    prop (SOME (SLc (PL report1)))::ins) WARNNO outs)
(CFG inputOK secContext secContextNull ins
  (NS WARNNO
    (exec
      (inputList
        [Name PlatoonLeader says
          prop (SOME (SLc (PL recon)));
          Name PlatoonLeader says
          prop (SOME (SLc (PL tentativePlan)));
          Name PlatoonSergeant says
          prop (SOME (SLc (PSG initiateMovement)));
          Name PlatoonLeader says
          prop (SOME (SLc (PL report1)))])))
    (Out WARNNO
      (exec
        (inputList
          [Name PlatoonLeader says
            prop (SOME (SLc (PL recon)));
            Name PlatoonLeader says
            prop (SOME (SLc (PL tentativePlan)));
            Name PlatoonSergeant says
            prop (SOME (SLc (PSG initiateMovement)));
            Name PlatoonLeader says
            prop (SOME (SLc (PL report1)))]))::outs))  $\iff$ 
authenticationTest inputOK
  [Name PlatoonLeader says prop (SOME (SLc (PL recon)));
   Name PlatoonLeader says
   prop (SOME (SLc (PL tentativePlan)));
   Name PlatoonSergeant says
   prop (SOME (SLc (PSG initiateMovement)));
   Name PlatoonLeader says
   prop (SOME (SLc (PL report1)))]  $\wedge$ 
CFGInterpret ( $M, O_i, O_s$ )
  (CFG inputOK secContext secContextNull
    ([Name PlatoonLeader says
      prop (SOME (SLc (PL recon)));
      Name PlatoonLeader says
      prop (SOME (SLc (PL tentativePlan)));
      Name PlatoonSergeant says
      prop (SOME (SLc (PSG initiateMovement)));
      Name PlatoonLeader says
      prop (SOME (SLc (PL report1)))]::ins) WARNNO outs)  $\wedge$ 
  ( $M, O_i, O_s$ ) satList
propCommandList
  [Name PlatoonLeader says prop (SOME (SLc (PL recon)));

```

```

Name PlatoonLeader says
prop (SOME (SLc (PL tentativePlan)));
Name PlatoonSergeant says
prop (SOME (SLc (PSG initiateMovement)));
Name PlatoonLeader says prop (SOME (SLc (PL report1))))]

[PlatoonLeader_WARNO_exec_report1_justified_thm]
⊢ ∀ NS Out M Oi Os.
  TR (M, Oi, Os)
    (exec
      [SOME (SLc (PL recon)); SOME (SLc (PL tentativePlan));
       SOME (SLc (PSG initiateMovement));
       SOME (SLc (PL report1))])
    (CFG inputOK secContext secContextNull
      ([Name PlatoonLeader says
        prop (SOME (SLc (PL recon)));
        Name PlatoonLeader says
        prop (SOME (SLc (PL tentativePlan)));
        Name PlatoonSergeant says
        prop (SOME (SLc (PSG initiateMovement)));
        Name PlatoonLeader says
        prop (SOME (SLc (PL report1)))]::ins) WARNO outs)
    (CFG inputOK secContext secContextNull ins
      (NS WARNO
        (exec
          [SOME (SLc (PL recon));
           SOME (SLc (PL tentativePlan));
           SOME (SLc (PSG initiateMovement));
           SOME (SLc (PL report1))]))
      (Out WARNO
        (exec
          [SOME (SLc (PL recon));
           SOME (SLc (PL tentativePlan));
           SOME (SLc (PSG initiateMovement));
           SOME (SLc (PL report1))]]::outs)) ⇔⇒
    authenticationTest inputOK
      [Name PlatoonLeader says prop (SOME (SLc (PL recon)));
       Name PlatoonLeader says
       prop (SOME (SLc (PL tentativePlan)));
       Name PlatoonSergeant says
       prop (SOME (SLc (PSG initiateMovement)));
       Name PlatoonLeader says
       prop (SOME (SLc (PL report1)))] ∧
    CFGInterpret (M, Oi, Os)
      (CFG inputOK secContext secContextNull
        ([Name PlatoonLeader says
          prop (SOME (SLc (PL recon)));
          Name PlatoonLeader says
          prop (SOME (SLc (PL tentativePlan)))]

```

```

      Name PlatoonSergeant says
      prop (SOME (SLc (PSG initiateMovement)));
      Name PlatoonLeader says
      prop (SOME (SLc (PL report1)))::ins) WARNNO outs) ∧
(M, Oi, Os) satList
[prop (SOME (SLc (PL recon)));
 prop (SOME (SLc (PL tentativePlan)));
 prop (SOME (SLc (PSG initiateMovement)));
 prop (SOME (SLc (PL report1)))]

```

#### [PlatoonLeader\_WARNNO\_exec\_report1\_lemma]

```

⊢ ∀ M Oi Os.
  CFGInterpret (M, Oi, Os)
  (CFG inputOK secContext secContextNull
   ([Name PlatoonLeader says
     prop (SOME (SLc (PL recon)));
     Name PlatoonLeader says
     prop (SOME (SLc (PL tentativePlan)));
     Name PlatoonSergeant says
     prop (SOME (SLc (PSG initiateMovement)));
     Name PlatoonLeader says
     prop (SOME (SLc (PL report1)))::ins) WARNNO outs) ⇒
(M, Oi, Os) satList
propCommandList
 [Name PlatoonLeader says prop (SOME (SLc (PL recon)));
  Name PlatoonLeader says
  prop (SOME (SLc (PL tentativePlan)));
  Name PlatoonSergeant says
  prop (SOME (SLc (PSG initiateMovement)));
  Name PlatoonLeader says prop (SOME (SLc (PL report1)))]

```

#### [PlatoonSergeant\_trap\_plCommand\_justified\_lemma]

```

⊢ ∀ NS Out M Oi Os.
  TR (M, Oi, Os)
  (trap
   (inputList
    [Name PlatoonSergeant says
     prop (SOME (SLc (PL plCommand)))])
  (CFG inputOK secContext secContextNull
   ([Name PlatoonSergeant says
     prop (SOME (SLc (PL plCommand)))::ins) s outs)
  (CFG inputOK secContext secContextNull ins
   (NS s
    (trap
     (inputList
      [Name PlatoonSergeant says
       prop (SOME (SLc (PL plCommand)))])
    (Out s
     (trap

```

```

      (inputList
        [Name PlatoonSergeant says
          prop (SOME (SLc (PL plCommand))))]))::
outs))  $\iff$ 
authenticationTest inputOK
  [Name PlatoonSergeant says
    prop (SOME (SLc (PL plCommand)))]  $\wedge$ 
CFGInterpret (M, Oi, Os)
  (CFG inputOK secContext secContextNull
    ([Name PlatoonSergeant says
      prop (SOME (SLc (PL plCommand)))]::ins) s outs)  $\wedge$ 
    (M, Oi, Os) sat prop NONE

[PlatoonSergeant_trap_plCommand_justified_thm]
 $\vdash \forall NS$  Out M Oi Os.
  TR (M, Oi, Os) (trap [SOME (SLc (PL plCommand))])
    (CFG inputOK secContext secContextNull
      ([Name PlatoonSergeant says
        prop (SOME (SLc (PL plCommand)))]::ins) s outs)
      (CFG inputOK secContext secContextNull ins
        (NS s (trap [SOME (SLc (PL plCommand))]))
        (Out s (trap [SOME (SLc (PL plCommand)))]::outs))  $\iff$ 
authenticationTest inputOK
  [Name PlatoonSergeant says
    prop (SOME (SLc (PL plCommand)))]  $\wedge$ 
CFGInterpret (M, Oi, Os)
  (CFG inputOK secContext secContextNull
    ([Name PlatoonSergeant says
      prop (SOME (SLc (PL plCommand)))]::ins) s outs)  $\wedge$ 
    (M, Oi, Os) sat prop NONE

[PlatoonSergeant_trap_plCommand_lemma]
 $\vdash \forall M$  Oi Os.
  CFGInterpret (M, Oi, Os)
    (CFG inputOK secContext secContextNull
      ([Name PlatoonSergeant says
        prop (SOME (SLc (PL plCommand)))]::ins) s outs)  $\Rightarrow$ 
    (M, Oi, Os) sat prop NONE

```

## 16 PlanPBType Theory

**Built:** 10 June 2018

**Parent Theories:** indexedLists, patternMatches

### 16.1 Datatypes

```

plCommand = receiveMission | warno | tentativePlan | recon
             | report1 | completePlan | opoid | supervise | report2
             | complete | plIncomplete | invalidPlCommand

```

```

psgCommand = initiateMovement | psgIncomplete
            | invalidPsgCommand

slCommand = PL plCommand | PSG psgCommand

slOutput = PlanPB | ReceiveMission | Warno | TentativePlan
           | InitiateMovement | Recon | Report1 | CompletePlan
           | Opid | Supervise | Report2 | Complete
           | unAuthenticated | unAuthorized

slState = PLAN_PB | RECEIVE_MISSION | WARNO | TENTATIVE_PLAN
          | INITIATE_MOVEMENT | RECON | REPORT1 | COMPLETE_PLAN
          | OPOID | SUPERVISE | REPORT2 | COMPLETE

stateRole = PlatoonLeader | PlatoonSergeant

```

## 16.2 Theorems

### [plCommand\_distinct\_clauses]

```

⊢ receiveMission ≠ warno ∧ receiveMission ≠ tentativePlan ∧
  receiveMission ≠ recon ∧ receiveMission ≠ report1 ∧
  receiveMission ≠ completePlan ∧ receiveMission ≠ opoid ∧
  receiveMission ≠ supervise ∧ receiveMission ≠ report2 ∧
  receiveMission ≠ complete ∧ receiveMission ≠ plIncomplete ∧
  receiveMission ≠ invalidPlCommand ∧ warno ≠ tentativePlan ∧
  warno ≠ recon ∧ warno ≠ report1 ∧ warno ≠ completePlan ∧
  warno ≠ opoid ∧ warno ≠ supervise ∧ warno ≠ report2 ∧
  warno ≠ complete ∧ warno ≠ plIncomplete ∧
  warno ≠ invalidPlCommand ∧ tentativePlan ≠ recon ∧
  tentativePlan ≠ report1 ∧ tentativePlan ≠ completePlan ∧
  tentativePlan ≠ opoid ∧ tentativePlan ≠ supervise ∧
  tentativePlan ≠ report2 ∧ tentativePlan ≠ complete ∧
  tentativePlan ≠ plIncomplete ∧
  tentativePlan ≠ invalidPlCommand ∧ recon ≠ report1 ∧
  recon ≠ completePlan ∧ recon ≠ opoid ∧ recon ≠ supervise ∧
  recon ≠ report2 ∧ recon ≠ complete ∧ recon ≠ plIncomplete ∧
  recon ≠ invalidPlCommand ∧ report1 ≠ completePlan ∧
  report1 ≠ opoid ∧ report1 ≠ supervise ∧ report1 ≠ report2 ∧
  report1 ≠ complete ∧ report1 ≠ plIncomplete ∧
  report1 ≠ invalidPlCommand ∧ completePlan ≠ opoid ∧
  completePlan ≠ supervise ∧ completePlan ≠ report2 ∧
  completePlan ≠ complete ∧ completePlan ≠ plIncomplete ∧
  completePlan ≠ invalidPlCommand ∧ opoid ≠ supervise ∧
  opoid ≠ report2 ∧ opoid ≠ complete ∧ opoid ≠ plIncomplete ∧
  opoid ≠ invalidPlCommand ∧ supervise ≠ report2 ∧
  supervise ≠ complete ∧ supervise ≠ plIncomplete ∧
  supervise ≠ invalidPlCommand ∧ report2 ≠ complete ∧
  report2 ≠ plIncomplete ∧ report2 ≠ invalidPlCommand ∧
  complete ≠ plIncomplete ∧ complete ≠ invalidPlCommand ∧
  plIncomplete ≠ invalidPlCommand

```

**[psgCommand\_distinct\_clauses]**

$$\vdash \text{initiateMovement} \neq \text{psgIncomplete} \wedge \\ \text{initiateMovement} \neq \text{invalidPsgCommand} \wedge \\ \text{psgIncomplete} \neq \text{invalidPsgCommand}$$
**[slCommand\_distinct\_clauses]**

$$\vdash \forall a' a. \text{PL } a \neq \text{PSG } a'$$
**[slCommand\_one\_one]**

$$\vdash (\forall a a'. (\text{PL } a = \text{PL } a') \iff (a = a')) \wedge \\ \forall a a'. (\text{PSG } a = \text{PSG } a') \iff (a = a')$$
**[slOutput\_distinct\_clauses]**

$$\vdash \text{PlanPB} \neq \text{ReceiveMission} \wedge \text{PlanPB} \neq \text{Warno} \wedge \\ \text{PlanPB} \neq \text{TentativePlan} \wedge \text{PlanPB} \neq \text{InitiateMovement} \wedge \\ \text{PlanPB} \neq \text{Recon} \wedge \text{PlanPB} \neq \text{Report1} \wedge \text{PlanPB} \neq \text{CompletePlan} \wedge \\ \text{PlanPB} \neq \text{Opoid} \wedge \text{PlanPB} \neq \text{Supervise} \wedge \text{PlanPB} \neq \text{Report2} \wedge \\ \text{PlanPB} \neq \text{Complete} \wedge \text{PlanPB} \neq \text{unAuthenticated} \wedge \\ \text{PlanPB} \neq \text{unAuthorized} \wedge \text{ReceiveMission} \neq \text{Warno} \wedge \\ \text{ReceiveMission} \neq \text{TentativePlan} \wedge \\ \text{ReceiveMission} \neq \text{InitiateMovement} \wedge \text{ReceiveMission} \neq \text{Recon} \wedge \\ \text{ReceiveMission} \neq \text{Report1} \wedge \text{ReceiveMission} \neq \text{CompletePlan} \wedge \\ \text{ReceiveMission} \neq \text{Opoid} \wedge \text{ReceiveMission} \neq \text{Supervise} \wedge \\ \text{ReceiveMission} \neq \text{Report2} \wedge \text{ReceiveMission} \neq \text{Complete} \wedge \\ \text{ReceiveMission} \neq \text{unAuthenticated} \wedge \\ \text{ReceiveMission} \neq \text{unAuthorized} \wedge \text{Warno} \neq \text{TentativePlan} \wedge \\ \text{Warno} \neq \text{InitiateMovement} \wedge \text{Warno} \neq \text{Recon} \wedge \text{Warno} \neq \text{Report1} \wedge \\ \text{Warno} \neq \text{CompletePlan} \wedge \text{Warno} \neq \text{Opoid} \wedge \text{Warno} \neq \text{Supervise} \wedge \\ \text{Warno} \neq \text{Report2} \wedge \text{Warno} \neq \text{Complete} \wedge \\ \text{Warno} \neq \text{unAuthenticated} \wedge \text{Warno} \neq \text{unAuthorized} \wedge \\ \text{TentativePlan} \neq \text{InitiateMovement} \wedge \text{TentativePlan} \neq \text{Recon} \wedge \\ \text{TentativePlan} \neq \text{Report1} \wedge \text{TentativePlan} \neq \text{CompletePlan} \wedge \\ \text{TentativePlan} \neq \text{Opoid} \wedge \text{TentativePlan} \neq \text{Supervise} \wedge \\ \text{TentativePlan} \neq \text{Report2} \wedge \text{TentativePlan} \neq \text{Complete} \wedge \\ \text{TentativePlan} \neq \text{unAuthenticated} \wedge \\ \text{TentativePlan} \neq \text{unAuthorized} \wedge \text{InitiateMovement} \neq \text{Recon} \wedge \\ \text{InitiateMovement} \neq \text{Report1} \wedge \\ \text{InitiateMovement} \neq \text{CompletePlan} \wedge \text{InitiateMovement} \neq \text{Opoid} \wedge \\ \text{InitiateMovement} \neq \text{Supervise} \wedge \text{InitiateMovement} \neq \text{Report2} \wedge \\ \text{InitiateMovement} \neq \text{Complete} \wedge \\ \text{InitiateMovement} \neq \text{unAuthenticated} \wedge \\ \text{InitiateMovement} \neq \text{unAuthorized} \wedge \text{Recon} \neq \text{Report1} \wedge \\ \text{Recon} \neq \text{CompletePlan} \wedge \text{Recon} \neq \text{Opoid} \wedge \text{Recon} \neq \text{Supervise} \wedge \\ \text{Recon} \neq \text{Report2} \wedge \text{Recon} \neq \text{Complete} \wedge \\ \text{Recon} \neq \text{unAuthenticated} \wedge \text{Recon} \neq \text{unAuthorized} \wedge \\ \text{Report1} \neq \text{CompletePlan} \wedge \text{Report1} \neq \text{Opoid} \wedge \\ \text{Report1} \neq \text{Supervise} \wedge \text{Report1} \neq \text{Report2} \wedge \\ \text{Report1} \neq \text{Complete} \wedge \text{Report1} \neq \text{unAuthenticated} \wedge$$



$\text{Report1} \neq \text{unAuthorized} \wedge \text{CompletePlan} \neq \text{Opoid} \wedge$   
 $\text{CompletePlan} \neq \text{Supervise} \wedge \text{CompletePlan} \neq \text{Report2} \wedge$   
 $\text{CompletePlan} \neq \text{Complete} \wedge \text{CompletePlan} \neq \text{unAuthenticated} \wedge$   
 $\text{CompletePlan} \neq \text{unAuthorized} \wedge \text{Opoid} \neq \text{Supervise} \wedge$   
 $\text{Opoid} \neq \text{Report2} \wedge \text{Opoid} \neq \text{Complete} \wedge$   
 $\text{Opoid} \neq \text{unAuthenticated} \wedge \text{Opoid} \neq \text{unAuthorized} \wedge$   
 $\text{Supervise} \neq \text{Report2} \wedge \text{Supervise} \neq \text{Complete} \wedge$   
 $\text{Supervise} \neq \text{unAuthenticated} \wedge \text{Supervise} \neq \text{unAuthorized} \wedge$   
 $\text{Report2} \neq \text{Complete} \wedge \text{Report2} \neq \text{unAuthenticated} \wedge$   
 $\text{Report2} \neq \text{unAuthorized} \wedge \text{Complete} \neq \text{unAuthenticated} \wedge$   
 $\text{Complete} \neq \text{unAuthorized} \wedge \text{unAuthenticated} \neq \text{unAuthorized}$

[slRole\_distinct\_clauses]

$\vdash \text{PlatoonLeader} \neq \text{PlatoonSergeant}$

[slState\_distinct\_clauses]

$\vdash \text{PLAN\_PB} \neq \text{RECEIVE\_MISSION} \wedge \text{PLAN\_PB} \neq \text{WARNO} \wedge$   
 $\text{PLAN\_PB} \neq \text{TENTATIVE\_PLAN} \wedge \text{PLAN\_PB} \neq \text{INITIATE\_MOVEMENT} \wedge$   
 $\text{PLAN\_PB} \neq \text{RECON} \wedge \text{PLAN\_PB} \neq \text{REPORT1} \wedge$   
 $\text{PLAN\_PB} \neq \text{COMPLETE\_PLAN} \wedge \text{PLAN\_PB} \neq \text{OPOID} \wedge$   
 $\text{PLAN\_PB} \neq \text{SUPERVISE} \wedge \text{PLAN\_PB} \neq \text{REPORT2} \wedge$   
 $\text{PLAN\_PB} \neq \text{COMPLETE} \wedge \text{RECEIVE\_MISSION} \neq \text{WARNO} \wedge$   
 $\text{RECEIVE\_MISSION} \neq \text{TENTATIVE\_PLAN} \wedge$   
 $\text{RECEIVE\_MISSION} \neq \text{INITIATE\_MOVEMENT} \wedge$   
 $\text{RECEIVE\_MISSION} \neq \text{RECON} \wedge \text{RECEIVE\_MISSION} \neq \text{REPORT1} \wedge$   
 $\text{RECEIVE\_MISSION} \neq \text{COMPLETE\_PLAN} \wedge \text{RECEIVE\_MISSION} \neq \text{OPOID} \wedge$   
 $\text{RECEIVE\_MISSION} \neq \text{SUPERVISE} \wedge \text{RECEIVE\_MISSION} \neq \text{REPORT2} \wedge$   
 $\text{RECEIVE\_MISSION} \neq \text{COMPLETE} \wedge \text{WARNO} \neq \text{TENTATIVE\_PLAN} \wedge$   
 $\text{WARNO} \neq \text{INITIATE\_MOVEMENT} \wedge \text{WARNO} \neq \text{RECON} \wedge \text{WARNO} \neq \text{REPORT1} \wedge$   
 $\text{WARNO} \neq \text{COMPLETE\_PLAN} \wedge \text{WARNO} \neq \text{OPOID} \wedge \text{WARNO} \neq \text{SUPERVISE} \wedge$   
 $\text{WARNO} \neq \text{REPORT2} \wedge \text{WARNO} \neq \text{COMPLETE} \wedge$   
 $\text{TENTATIVE\_PLAN} \neq \text{INITIATE\_MOVEMENT} \wedge \text{TENTATIVE\_PLAN} \neq \text{RECON} \wedge$   
 $\text{TENTATIVE\_PLAN} \neq \text{REPORT1} \wedge \text{TENTATIVE\_PLAN} \neq \text{COMPLETE\_PLAN} \wedge$   
 $\text{TENTATIVE\_PLAN} \neq \text{OPOID} \wedge \text{TENTATIVE\_PLAN} \neq \text{SUPERVISE} \wedge$   
 $\text{TENTATIVE\_PLAN} \neq \text{REPORT2} \wedge \text{TENTATIVE\_PLAN} \neq \text{COMPLETE} \wedge$   
 $\text{INITIATE\_MOVEMENT} \neq \text{RECON} \wedge \text{INITIATE\_MOVEMENT} \neq \text{REPORT1} \wedge$   
 $\text{INITIATE\_MOVEMENT} \neq \text{COMPLETE\_PLAN} \wedge$   
 $\text{INITIATE\_MOVEMENT} \neq \text{OPOID} \wedge \text{INITIATE\_MOVEMENT} \neq \text{SUPERVISE} \wedge$   
 $\text{INITIATE\_MOVEMENT} \neq \text{REPORT2} \wedge \text{INITIATE\_MOVEMENT} \neq \text{COMPLETE} \wedge$   
 $\text{RECON} \neq \text{REPORT1} \wedge \text{RECON} \neq \text{COMPLETE\_PLAN} \wedge \text{RECON} \neq \text{OPOID} \wedge$   
 $\text{RECON} \neq \text{SUPERVISE} \wedge \text{RECON} \neq \text{REPORT2} \wedge \text{RECON} \neq \text{COMPLETE} \wedge$   
 $\text{REPORT1} \neq \text{COMPLETE\_PLAN} \wedge \text{REPORT1} \neq \text{OPOID} \wedge$   
 $\text{REPORT1} \neq \text{SUPERVISE} \wedge \text{REPORT1} \neq \text{REPORT2} \wedge$   
 $\text{REPORT1} \neq \text{COMPLETE} \wedge \text{COMPLETE\_PLAN} \neq \text{OPOID} \wedge$   
 $\text{COMPLETE\_PLAN} \neq \text{SUPERVISE} \wedge \text{COMPLETE\_PLAN} \neq \text{REPORT2} \wedge$   
 $\text{COMPLETE\_PLAN} \neq \text{COMPLETE} \wedge \text{OPOID} \neq \text{SUPERVISE} \wedge$   
 $\text{OPOID} \neq \text{REPORT2} \wedge \text{OPOID} \neq \text{COMPLETE} \wedge \text{SUPERVISE} \neq \text{REPORT2} \wedge$   
 $\text{SUPERVISE} \neq \text{COMPLETE} \wedge \text{REPORT2} \neq \text{COMPLETE}$

## 17 PlanPBDef Theory

**Built:** 10 June 2018

**Parent Theories:** PlanPBType, acfFoundation, OMNIType

### 17.1 Definitions

[PL\_notWARNO\_Auth\_def]

```
⊢ ∀ cmd.  
  PL_notWARNO_Auth cmd =  
  if cmd = report1 then prop NONE  
  else  
    Name PlatoonLeader says prop (SOME (SLc (PL cmd))) impf  
    Name PlatoonLeader controls prop (SOME (SLc (PL cmd)))
```

[PL\_WARNO\_Auth\_def]

```
⊢ PL_WARNO_Auth =  
  prop (SOME (SLc (PL recon))) impf  
  prop (SOME (SLc (PL tentativePlan))) impf  
  prop (SOME (SLc (PSG initiateMovement))) impf  
  Name PlatoonLeader controls prop (SOME (SLc (PL report1)))
```

[secContext\_def]

```
⊢ ∀ s x.  
  secContext s x =  
  if s = WARNO then  
    if  
      (getRecon x = [SOME (SLc (PL recon))]) ∧  
      (getTentativePlan x = [SOME (SLc (PL tentativePlan))]) ∧  
      (getReport x = [SOME (SLc (PL report1))]) ∧  
      (getInitMove x = [SOME (SLc (PSG initiateMovement))])  
    then  
      [PL_WARNO_Auth;  
       Name PlatoonLeader controls  
         prop (SOME (SLc (PL recon)));  
       Name PlatoonLeader controls  
         prop (SOME (SLc (PL tentativePlan)));  
       Name PlatoonSergeant controls  
         prop (SOME (SLc (PSG initiateMovement)))]  
    else [prop NONE]  
  else if getP1Com x = invalidP1Command then [prop NONE]  
  else [PL_notWARNO_Auth (getP1Com x)]
```

[secContextNull\_def]

```
⊢ ∀ x. secContextNull x = [TT]
```

## 17.2 Theorems

[getInitMove\_def]

$$\begin{aligned}
& \vdash (\text{getInitMove } [] = [\text{NONE}]) \wedge \\
& (\forall xs. \\
& \quad \text{getInitMove} \\
& \quad \quad (\text{Name PlatoonSergeant says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PSG initiateMovement)))::xs}) = \\
& \quad \quad [\text{SOME (SLc (PSG initiateMovement))}] \wedge \\
& \quad (\forall xs. \text{getInitMove (TT::xs)} = \text{getInitMove xs}) \wedge \\
& \quad (\forall xs. \text{getInitMove (FF::xs)} = \text{getInitMove xs}) \wedge \\
& \quad (\forall xs \ v_2. \text{getInitMove (prop } v_2::xs) = \text{getInitMove xs}) \wedge \\
& \quad (\forall xs \ v_3. \text{getInitMove (notf } v_3::xs) = \text{getInitMove xs}) \wedge \\
& \quad (\forall xs \ v_5 \ v_4. \text{getInitMove (v}_4 \text{ andf } v_5::xs) = \text{getInitMove xs}) \wedge \\
& \quad (\forall xs \ v_7 \ v_6. \text{getInitMove (v}_6 \text{ orf } v_7::xs) = \text{getInitMove xs}) \wedge \\
& \quad (\forall xs \ v_9 \ v_8. \text{getInitMove (v}_8 \text{ impf } v_9::xs) = \text{getInitMove xs}) \wedge \\
& \quad (\forall xs \ v_{11} \ v_{10}. \\
& \quad \quad \text{getInitMove (v}_{10} \text{ eqf } v_{11}::xs) = \text{getInitMove xs}) \wedge \\
& \quad (\forall xs \ v_{12}. \text{getInitMove (v}_{12} \text{ says TT::xs)} = \text{getInitMove xs}) \wedge \\
& \quad (\forall xs \ v_{12}. \text{getInitMove (v}_{12} \text{ says FF::xs)} = \text{getInitMove xs}) \wedge \\
& \quad (\forall xs \ v_{134}. \\
& \quad \quad \text{getInitMove (Name } v_{134} \text{ says prop NONE::xs)} = \\
& \quad \quad \text{getInitMove xs}) \wedge \\
& \quad (\forall xs \ v_{144}. \\
& \quad \quad \text{getInitMove} \\
& \quad \quad \quad (\text{Name PlatoonLeader says prop (SOME } v_{144})::xs) = \\
& \quad \quad \quad \text{getInitMove xs}) \wedge \\
& \quad (\forall xs \ v_{146}. \\
& \quad \quad \text{getInitMove} \\
& \quad \quad \quad (\text{Name PlatoonSergeant says prop (SOME (ESCc } v_{146}))::} \\
& \quad \quad \quad \text{xs}) = \\
& \quad \quad \text{getInitMove xs}) \wedge \\
& \quad (\forall xs \ v_{150}. \\
& \quad \quad \text{getInitMove} \\
& \quad \quad \quad (\text{Name PlatoonSergeant says prop (SOME (SLc (PL } v_{150}))::} \\
& \quad \quad \quad \text{xs}) = \\
& \quad \quad \text{getInitMove xs}) \wedge \\
& \quad (\forall xs. \\
& \quad \quad \text{getInitMove} \\
& \quad \quad \quad (\text{Name PlatoonSergeant says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PSG psgIncomplete)))::xs}) = \\
& \quad \quad \text{getInitMove xs}) \wedge \\
& \quad (\forall xs. \\
& \quad \quad \text{getInitMove} \\
& \quad \quad \quad (\text{Name PlatoonSergeant says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PSG invalidPsgCommand)))::xs}) = \\
& \quad \quad \text{getInitMove xs}) \wedge \\
& \quad (\forall xs \ v_{68} \ v_{136} \ v_{135}. \\
& \quad \quad \text{getInitMove (v}_{135} \text{ meet } v_{136} \text{ says prop } v_{68}::xs) =
\end{aligned}$$

---

```

    getInitMove xs) ∧
  (∀ xs v68 v138 v137.
    getInitMove (v137 quoting v138 says prop v68::xs) =
    getInitMove xs) ∧
  (∀ xs v69 v12.
    getInitMove (v12 says notf v69::xs) = getInitMove xs) ∧
  (∀ xs v71 v70 v12.
    getInitMove (v12 says (v70 andf v71)::xs) =
    getInitMove xs) ∧
  (∀ xs v73 v72 v12.
    getInitMove (v12 says (v72 orf v73)::xs) =
    getInitMove xs) ∧
  (∀ xs v75 v74 v12.
    getInitMove (v12 says (v74 impf v75)::xs) =
    getInitMove xs) ∧
  (∀ xs v77 v76 v12.
    getInitMove (v12 says (v76 eqf v77)::xs) =
    getInitMove xs) ∧
  (∀ xs v79 v78 v12.
    getInitMove (v12 says v78 says v79::xs) =
    getInitMove xs) ∧
  (∀ xs v81 v80 v12.
    getInitMove (v12 says v80 speaks_for v81::xs) =
    getInitMove xs) ∧
  (∀ xs v83 v82 v12.
    getInitMove (v12 says v82 controls v83::xs) =
    getInitMove xs) ∧
  (∀ xs v86 v85 v84 v12.
    getInitMove (v12 says reps v84 v85 v86::xs) =
    getInitMove xs) ∧
  (∀ xs v88 v87 v12.
    getInitMove (v12 says v87 domi v88::xs) =
    getInitMove xs) ∧
  (∀ xs v90 v89 v12.
    getInitMove (v12 says v89 eqi v90::xs) = getInitMove xs) ∧
  (∀ xs v92 v91 v12.
    getInitMove (v12 says v91 doms v92::xs) =
    getInitMove xs) ∧
  (∀ xs v94 v93 v12.
    getInitMove (v12 says v93 eqs v94::xs) = getInitMove xs) ∧
  (∀ xs v96 v95 v12.
    getInitMove (v12 says v95 eqn v96::xs) = getInitMove xs) ∧
  (∀ xs v98 v97 v12.
    getInitMove (v12 says v97 lte v98::xs) = getInitMove xs) ∧
  (∀ xs v99 v12 v100.
    getInitMove (v12 says v99 lt v100::xs) = getInitMove xs) ∧
  (∀ xs v15 v14.
    getInitMove (v14 speaks_for v15::xs) = getInitMove xs) ∧
  (∀ xs v17 v16.

```

---

```

    getInitMove (v16 controls v17::xs) = getInitMove xs) ∧
  (∀ xs v20 v19 v18.
    getInitMove (reps v18 v19 v20::xs) = getInitMove xs) ∧
  (∀ xs v22 v21.
    getInitMove (v21 domi v22::xs) = getInitMove xs) ∧
  (∀ xs v24 v23.
    getInitMove (v23 eqi v24::xs) = getInitMove xs) ∧
  (∀ xs v26 v25.
    getInitMove (v25 doms v26::xs) = getInitMove xs) ∧
  (∀ xs v28 v27.
    getInitMove (v27 eqs v28::xs) = getInitMove xs) ∧
  (∀ xs v30 v29.
    getInitMove (v29 eqn v30::xs) = getInitMove xs) ∧
  (∀ xs v32 v31.
    getInitMove (v31 lte v32::xs) = getInitMove xs) ∧
  ∀ xs v34 v33. getInitMove (v33 lt v34::xs) = getInitMove xs

```

[getInitMove\_ind]

```

⊢ ∀ P.
  P [] ∧
  (∀ xs.
    P
      (Name PlatoonSergeant says
        prop (SOME (SLc (PSG initiateMovement)))::xs)) ∧
    (∀ xs. P xs ⇒ P (TT::xs)) ∧ (∀ xs. P xs ⇒ P (FF::xs)) ∧
    (∀ v2 xs. P xs ⇒ P (prop v2::xs)) ∧
    (∀ v3 xs. P xs ⇒ P (notf v3::xs)) ∧
    (∀ v4 v5 xs. P xs ⇒ P (v4 andf v5::xs)) ∧
    (∀ v6 v7 xs. P xs ⇒ P (v6 orf v7::xs)) ∧
    (∀ v8 v9 xs. P xs ⇒ P (v8 impf v9::xs)) ∧
    (∀ v10 v11 xs. P xs ⇒ P (v10 eqf v11::xs)) ∧
    (∀ v12 xs. P xs ⇒ P (v12 says TT::xs)) ∧
    (∀ v12 xs. P xs ⇒ P (v12 says FF::xs)) ∧
    (∀ v134 xs. P xs ⇒ P (Name v134 says prop NONE::xs)) ∧
    (∀ v144 xs.
      P xs ⇒
      P (Name PlatoonLeader says prop (SOME v144)::xs)) ∧
    (∀ v146 xs.
      P xs ⇒
      P
        (Name PlatoonSergeant says prop (SOME (ESCc v146))::
          xs)) ∧
    (∀ v150 xs.
      P xs ⇒
      P
        (Name PlatoonSergeant says
          prop (SOME (SLc (PL v150)))::xs)) ∧
    (∀ xs.
      P xs ⇒

```

$$\begin{aligned}
& P \\
& \quad (\text{Name PlatoonSergeant says} \\
& \quad \quad \text{prop (SOME (SLc (PSG psgIncomplete)))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \quad xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonSergeant says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PSG invalidPsgCommand)))::xs}) \wedge \\
& (\forall v135 v136 v68 xs. \\
& \quad P \quad xs \Rightarrow P \quad (v135 \text{ meet } v136 \text{ says prop } v68::xs)) \wedge \\
& (\forall v137 v138 v68 xs. \\
& \quad P \quad xs \Rightarrow P \quad (v137 \text{ quoting } v138 \text{ says prop } v68::xs)) \wedge \\
& (\forall v12 v69 xs. P \quad xs \Rightarrow P \quad (v12 \text{ says notf } v69::xs)) \wedge \\
& (\forall v12 v70 v71 xs. P \quad xs \Rightarrow P \quad (v12 \text{ says (v70 andf v71)::xs})) \wedge \\
& (\forall v12 v72 v73 xs. P \quad xs \Rightarrow P \quad (v12 \text{ says (v72 orf v73)::xs})) \wedge \\
& (\forall v12 v74 v75 xs. P \quad xs \Rightarrow P \quad (v12 \text{ says (v74 impf v75)::xs})) \wedge \\
& (\forall v12 v76 v77 xs. P \quad xs \Rightarrow P \quad (v12 \text{ says (v76 eqf v77)::xs})) \wedge \\
& (\forall v12 v78 v79 xs. P \quad xs \Rightarrow P \quad (v12 \text{ says } v78 \text{ says } v79::xs)) \wedge \\
& (\forall v12 v80 v81 xs. \\
& \quad P \quad xs \Rightarrow P \quad (v12 \text{ says } v80 \text{ speaks\_for } v81::xs)) \wedge \\
& (\forall v12 v82 v83 xs. \\
& \quad P \quad xs \Rightarrow P \quad (v12 \text{ says } v82 \text{ controls } v83::xs)) \wedge \\
& (\forall v12 v84 v85 v86 xs. \\
& \quad P \quad xs \Rightarrow P \quad (v12 \text{ says reps } v84 v85 v86::xs)) \wedge \\
& (\forall v12 v87 v88 xs. P \quad xs \Rightarrow P \quad (v12 \text{ says } v87 \text{ domi } v88::xs)) \wedge \\
& (\forall v12 v89 v90 xs. P \quad xs \Rightarrow P \quad (v12 \text{ says } v89 \text{ eqi } v90::xs)) \wedge \\
& (\forall v12 v91 v92 xs. P \quad xs \Rightarrow P \quad (v12 \text{ says } v91 \text{ doms } v92::xs)) \wedge \\
& (\forall v12 v93 v94 xs. P \quad xs \Rightarrow P \quad (v12 \text{ says } v93 \text{ eqs } v94::xs)) \wedge \\
& (\forall v12 v95 v96 xs. P \quad xs \Rightarrow P \quad (v12 \text{ says } v95 \text{ eqn } v96::xs)) \wedge \\
& (\forall v12 v97 v98 xs. P \quad xs \Rightarrow P \quad (v12 \text{ says } v97 \text{ lte } v98::xs)) \wedge \\
& (\forall v12 v99 v100 xs. P \quad xs \Rightarrow P \quad (v12 \text{ says } v99 \text{ lt } v100::xs)) \wedge \\
& (\forall v14 v15 xs. P \quad xs \Rightarrow P \quad (v14 \text{ speaks\_for } v15::xs)) \wedge \\
& (\forall v16 v17 xs. P \quad xs \Rightarrow P \quad (v16 \text{ controls } v17::xs)) \wedge \\
& (\forall v18 v19 v20 xs. P \quad xs \Rightarrow P \quad (\text{reps } v18 v19 v20::xs)) \wedge \\
& (\forall v21 v22 xs. P \quad xs \Rightarrow P \quad (v21 \text{ domi } v22::xs)) \wedge \\
& (\forall v23 v24 xs. P \quad xs \Rightarrow P \quad (v23 \text{ eqi } v24::xs)) \wedge \\
& (\forall v25 v26 xs. P \quad xs \Rightarrow P \quad (v25 \text{ doms } v26::xs)) \wedge \\
& (\forall v27 v28 xs. P \quad xs \Rightarrow P \quad (v27 \text{ eqs } v28::xs)) \wedge \\
& (\forall v29 v30 xs. P \quad xs \Rightarrow P \quad (v29 \text{ eqn } v30::xs)) \wedge \\
& (\forall v31 v32 xs. P \quad xs \Rightarrow P \quad (v31 \text{ lte } v32::xs)) \wedge \\
& (\forall v33 v34 xs. P \quad xs \Rightarrow P \quad (v33 \text{ lt } v34::xs)) \Rightarrow \\
& \forall v. P \quad v
\end{aligned}$$

[getPlCom\_def]

$$\begin{aligned}
& \vdash (\text{getPlCom } [] = \text{invalidPlCommand}) \wedge \\
& (\forall xs \text{ cmd}. \\
& \quad \text{getPlCom} \\
& \quad \quad (\text{Name PlatoonLeader says prop (SOME (SLc (PL cmd)))::} \\
& \quad \quad \quad xs) =
\end{aligned}$$

---

```

  cmd) ∧ (∀ xs. getPlCom (TT::xs) = getPlCom xs) ∧
  (∀ xs. getPlCom (FF::xs) = getPlCom xs) ∧
  (∀ xs v2. getPlCom (prop v2::xs) = getPlCom xs) ∧
  (∀ xs v3. getPlCom (notf v3::xs) = getPlCom xs) ∧
  (∀ xs v5 v4. getPlCom (v4 andf v5::xs) = getPlCom xs) ∧
  (∀ xs v7 v6. getPlCom (v6 orf v7::xs) = getPlCom xs) ∧
  (∀ xs v9 v8. getPlCom (v8 impf v9::xs) = getPlCom xs) ∧
  (∀ xs v11 v10. getPlCom (v10 eqf v11::xs) = getPlCom xs) ∧
  (∀ xs v12. getPlCom (v12 says TT::xs) = getPlCom xs) ∧
  (∀ xs v12. getPlCom (v12 says FF::xs) = getPlCom xs) ∧
  (∀ xs v134.
    getPlCom (Name v134 says prop NONE::xs) = getPlCom xs) ∧
  (∀ xs v146.
    getPlCom
      (Name PlatoonLeader says prop (SOME (ESCc v146))::xs) =
      getPlCom xs) ∧
  (∀ xs v151.
    getPlCom
      (Name PlatoonLeader says prop (SOME (SLc (PSG v151)))::
        xs) =
      getPlCom xs) ∧
  (∀ xs v144.
    getPlCom
      (Name PlatoonSergeant says prop (SOME v144)::xs) =
      getPlCom xs) ∧
  (∀ xs v68 v136 v135.
    getPlCom (v135 meet v136 says prop v68::xs) =
    getPlCom xs) ∧
  (∀ xs v68 v138 v137.
    getPlCom (v137 quoting v138 says prop v68::xs) =
    getPlCom xs) ∧
  (∀ xs v69 v12.
    getPlCom (v12 says notf v69::xs) = getPlCom xs) ∧
  (∀ xs v71 v70 v12.
    getPlCom (v12 says (v70 andf v71)::xs) = getPlCom xs) ∧
  (∀ xs v73 v72 v12.
    getPlCom (v12 says (v72 orf v73)::xs) = getPlCom xs) ∧
  (∀ xs v75 v74 v12.
    getPlCom (v12 says (v74 impf v75)::xs) = getPlCom xs) ∧
  (∀ xs v77 v76 v12.
    getPlCom (v12 says (v76 eqf v77)::xs) = getPlCom xs) ∧
  (∀ xs v79 v78 v12.
    getPlCom (v12 says v78 says v79::xs) = getPlCom xs) ∧
  (∀ xs v81 v80 v12.
    getPlCom (v12 says v80 speaks_for v81::xs) =
    getPlCom xs) ∧
  (∀ xs v83 v82 v12.
    getPlCom (v12 says v82 controls v83::xs) = getPlCom xs) ∧
  (∀ xs v86 v85 v84 v12.

```

---

```

    getPlCom (v12 says reps v84 v85 v86::xs) = getPlCom xs) ∧
  (∀ xs v88 v87 v12.
    getPlCom (v12 says v87 domi v88::xs) = getPlCom xs) ∧
  (∀ xs v90 v89 v12.
    getPlCom (v12 says v89 eqi v90::xs) = getPlCom xs) ∧
  (∀ xs v92 v91 v12.
    getPlCom (v12 says v91 doms v92::xs) = getPlCom xs) ∧
  (∀ xs v94 v93 v12.
    getPlCom (v12 says v93 eqs v94::xs) = getPlCom xs) ∧
  (∀ xs v96 v95 v12.
    getPlCom (v12 says v95 eqn v96::xs) = getPlCom xs) ∧
  (∀ xs v98 v97 v12.
    getPlCom (v12 says v97 lte v98::xs) = getPlCom xs) ∧
  (∀ xs v99 v12 v100.
    getPlCom (v12 says v99 lt v100::xs) = getPlCom xs) ∧
  (∀ xs v15 v14.
    getPlCom (v14 speaks_for v15::xs) = getPlCom xs) ∧
  (∀ xs v17 v16.
    getPlCom (v16 controls v17::xs) = getPlCom xs) ∧
  (∀ xs v20 v19 v18.
    getPlCom (reps v18 v19 v20::xs) = getPlCom xs) ∧
  (∀ xs v22 v21. getPlCom (v21 domi v22::xs) = getPlCom xs) ∧
  (∀ xs v24 v23. getPlCom (v23 eqi v24::xs) = getPlCom xs) ∧
  (∀ xs v26 v25. getPlCom (v25 doms v26::xs) = getPlCom xs) ∧
  (∀ xs v28 v27. getPlCom (v27 eqs v28::xs) = getPlCom xs) ∧
  (∀ xs v30 v29. getPlCom (v29 eqn v30::xs) = getPlCom xs) ∧
  (∀ xs v32 v31. getPlCom (v31 lte v32::xs) = getPlCom xs) ∧
  ∀ xs v34 v33. getPlCom (v33 lt v34::xs) = getPlCom xs

```

[getPlCom\_ind]

```

⊢ ∀ P.
  P [] ∧
  (∀ cmd xs.
    P
      (Name PlatoonLeader says prop (SOME (SLc (PL cmd))))::
        xs)) ∧ (∀ xs. P xs ⇒ P (TT::xs)) ∧
  (∀ xs. P xs ⇒ P (FF::xs)) ∧
  (∀ v2 xs. P xs ⇒ P (prop v2::xs)) ∧
  (∀ v3 xs. P xs ⇒ P (notf v3::xs)) ∧
  (∀ v4 v5 xs. P xs ⇒ P (v4 andf v5::xs)) ∧
  (∀ v6 v7 xs. P xs ⇒ P (v6 orf v7::xs)) ∧
  (∀ v8 v9 xs. P xs ⇒ P (v8 impf v9::xs)) ∧
  (∀ v10 v11 xs. P xs ⇒ P (v10 eqf v11::xs)) ∧
  (∀ v12 xs. P xs ⇒ P (v12 says TT::xs)) ∧
  (∀ v12 xs. P xs ⇒ P (v12 says FF::xs)) ∧
  (∀ v134 xs. P xs ⇒ P (Name v134 says prop NONE::xs)) ∧
  (∀ v146 xs.
    P xs ⇒
    P

```



```

(Name PlatoonLeader says prop (SOME (ESCc v146))::
  xs)) ∧
(∀ v151 xs.
  P xs ⇒
  P
    (Name PlatoonLeader says
      prop (SOME (SLc (PSG v151)))::xs)) ∧
(∀ v144 xs.
  P xs ⇒
  P (Name PlatoonSergeant says prop (SOME v144)::xs)) ∧
(∀ v135 v136 v68 xs.
  P xs ⇒ P (v135 meet v136 says prop v68::xs)) ∧
(∀ v137 v138 v68 xs.
  P xs ⇒ P (v137 quoting v138 says prop v68::xs)) ∧
(∀ v12 v69 xs. P xs ⇒ P (v12 says notf v69::xs)) ∧
(∀ v12 v70 v71 xs. P xs ⇒ P (v12 says (v70 andf v71)::xs)) ∧
(∀ v12 v72 v73 xs. P xs ⇒ P (v12 says (v72 orf v73)::xs)) ∧
(∀ v12 v74 v75 xs. P xs ⇒ P (v12 says (v74 impf v75)::xs)) ∧
(∀ v12 v76 v77 xs. P xs ⇒ P (v12 says (v76 eqf v77)::xs)) ∧
(∀ v12 v78 v79 xs. P xs ⇒ P (v12 says v78 says v79::xs)) ∧
(∀ v12 v80 v81 xs.
  P xs ⇒ P (v12 says v80 speaks_for v81::xs)) ∧
(∀ v12 v82 v83 xs.
  P xs ⇒ P (v12 says v82 controls v83::xs)) ∧
(∀ v12 v84 v85 v86 xs.
  P xs ⇒ P (v12 says reps v84 v85 v86::xs)) ∧
(∀ v12 v87 v88 xs. P xs ⇒ P (v12 says v87 domi v88::xs)) ∧
(∀ v12 v89 v90 xs. P xs ⇒ P (v12 says v89 eqi v90::xs)) ∧
(∀ v12 v91 v92 xs. P xs ⇒ P (v12 says v91 doms v92::xs)) ∧
(∀ v12 v93 v94 xs. P xs ⇒ P (v12 says v93 eqs v94::xs)) ∧
(∀ v12 v95 v96 xs. P xs ⇒ P (v12 says v95 eqn v96::xs)) ∧
(∀ v12 v97 v98 xs. P xs ⇒ P (v12 says v97 lte v98::xs)) ∧
(∀ v12 v99 v100 xs. P xs ⇒ P (v12 says v99 lt v100::xs)) ∧
(∀ v14 v15 xs. P xs ⇒ P (v14 speaks_for v15::xs)) ∧
(∀ v16 v17 xs. P xs ⇒ P (v16 controls v17::xs)) ∧
(∀ v18 v19 v20 xs. P xs ⇒ P (reps v18 v19 v20::xs)) ∧
(∀ v21 v22 xs. P xs ⇒ P (v21 domi v22::xs)) ∧
(∀ v23 v24 xs. P xs ⇒ P (v23 eqi v24::xs)) ∧
(∀ v25 v26 xs. P xs ⇒ P (v25 doms v26::xs)) ∧
(∀ v27 v28 xs. P xs ⇒ P (v27 eqs v28::xs)) ∧
(∀ v29 v30 xs. P xs ⇒ P (v29 eqn v30::xs)) ∧
(∀ v31 v32 xs. P xs ⇒ P (v31 lte v32::xs)) ∧
(∀ v33 v34 xs. P xs ⇒ P (v33 lt v34::xs)) ⇒
∀ v. P v

```

[getPsgCom\_def]

```

⊢ (getPsgCom [] = invalidPsgCommand) ∧
(∀ xs cmd.
  getPsgCom

```

```

(Name PlatoonSergeant says prop (SOME (SLc (PSG cmd))))::
  xs) =
  cmd) ∧ (∀ xs. getPsgCom (TT::xs) = getPsgCom xs) ∧
(∀ xs. getPsgCom (FF::xs) = getPsgCom xs) ∧
(∀ xs v2. getPsgCom (prop v2::xs) = getPsgCom xs) ∧
(∀ xs v3. getPsgCom (notf v3::xs) = getPsgCom xs) ∧
(∀ xs v5 v4. getPsgCom (v4 andf v5::xs) = getPsgCom xs) ∧
(∀ xs v7 v6. getPsgCom (v6 orf v7::xs) = getPsgCom xs) ∧
(∀ xs v9 v8. getPsgCom (v8 impf v9::xs) = getPsgCom xs) ∧
(∀ xs v11 v10. getPsgCom (v10 eqf v11::xs) = getPsgCom xs) ∧
(∀ xs v12. getPsgCom (v12 says TT::xs) = getPsgCom xs) ∧
(∀ xs v12. getPsgCom (v12 says FF::xs) = getPsgCom xs) ∧
(∀ xs v134.
  getPsgCom (Name v134 says prop NONE::xs) = getPsgCom xs) ∧
(∀ xs v144.
  getPsgCom (Name PlatoonLeader says prop (SOME v144)::xs) =
  getPsgCom xs) ∧
(∀ xs v146.
  getPsgCom
    (Name PlatoonSergeant says prop (SOME (ESCc v146))))::
    xs) =
  getPsgCom xs) ∧
(∀ xs v150.
  getPsgCom
    (Name PlatoonSergeant says prop (SOME (SLc (PL v150))))::
    xs) =
  getPsgCom xs) ∧
(∀ xs v68 v136 v135.
  getPsgCom (v135 meet v136 says prop v68::xs) =
  getPsgCom xs) ∧
(∀ xs v68 v138 v137.
  getPsgCom (v137 quoting v138 says prop v68::xs) =
  getPsgCom xs) ∧
(∀ xs v69 v12.
  getPsgCom (v12 says notf v69::xs) = getPsgCom xs) ∧
(∀ xs v71 v70 v12.
  getPsgCom (v12 says (v70 andf v71)::xs) = getPsgCom xs) ∧
(∀ xs v73 v72 v12.
  getPsgCom (v12 says (v72 orf v73)::xs) = getPsgCom xs) ∧
(∀ xs v75 v74 v12.
  getPsgCom (v12 says (v74 impf v75)::xs) = getPsgCom xs) ∧
(∀ xs v77 v76 v12.
  getPsgCom (v12 says (v76 eqf v77)::xs) = getPsgCom xs) ∧
(∀ xs v79 v78 v12.
  getPsgCom (v12 says v78 says v79::xs) = getPsgCom xs) ∧
(∀ xs v81 v80 v12.
  getPsgCom (v12 says v80 speaks_for v81::xs) =
  getPsgCom xs) ∧
(∀ xs v83 v82 v12.

```

```

    getPsgCom (v12 says v82 controls v83::xs) =
    getPsgCom xs) ∧
  (∀ xs v86 v85 v84 v12.
    getPsgCom (v12 says reps v84 v85 v86::xs) =
    getPsgCom xs) ∧
  (∀ xs v88 v87 v12.
    getPsgCom (v12 says v87 domi v88::xs) = getPsgCom xs) ∧
  (∀ xs v90 v89 v12.
    getPsgCom (v12 says v89 eqi v90::xs) = getPsgCom xs) ∧
  (∀ xs v92 v91 v12.
    getPsgCom (v12 says v91 doms v92::xs) = getPsgCom xs) ∧
  (∀ xs v94 v93 v12.
    getPsgCom (v12 says v93 eqs v94::xs) = getPsgCom xs) ∧
  (∀ xs v96 v95 v12.
    getPsgCom (v12 says v95 eqn v96::xs) = getPsgCom xs) ∧
  (∀ xs v98 v97 v12.
    getPsgCom (v12 says v97 lte v98::xs) = getPsgCom xs) ∧
  (∀ xs v99 v12 v100.
    getPsgCom (v12 says v99 lt v100::xs) = getPsgCom xs) ∧
  (∀ xs v15 v14.
    getPsgCom (v14 speaks_for v15::xs) = getPsgCom xs) ∧
  (∀ xs v17 v16.
    getPsgCom (v16 controls v17::xs) = getPsgCom xs) ∧
  (∀ xs v20 v19 v18.
    getPsgCom (reps v18 v19 v20::xs) = getPsgCom xs) ∧
  (∀ xs v22 v21. getPsgCom (v21 domi v22::xs) = getPsgCom xs) ∧
  (∀ xs v24 v23. getPsgCom (v23 eqi v24::xs) = getPsgCom xs) ∧
  (∀ xs v26 v25. getPsgCom (v25 doms v26::xs) = getPsgCom xs) ∧
  (∀ xs v28 v27. getPsgCom (v27 eqs v28::xs) = getPsgCom xs) ∧
  (∀ xs v30 v29. getPsgCom (v29 eqn v30::xs) = getPsgCom xs) ∧
  (∀ xs v32 v31. getPsgCom (v31 lte v32::xs) = getPsgCom xs) ∧
  (∀ xs v34 v33. getPsgCom (v33 lt v34::xs) = getPsgCom xs

```

[getPsgCom\_ind]

```

⊢ ∀ P.
  P [] ∧
  (∀ cmd xs.
    P
      (Name PlatoonSergeant says
        prop (SOME (SLc (PSG cmd)))::xs)) ∧
  (∀ xs. P xs ⇒ P (TT::xs)) ∧ (∀ xs. P xs ⇒ P (FF::xs)) ∧
  (∀ v2 xs. P xs ⇒ P (prop v2::xs)) ∧
  (∀ v3 xs. P xs ⇒ P (notf v3::xs)) ∧
  (∀ v4 v5 xs. P xs ⇒ P (v4 andf v5::xs)) ∧
  (∀ v6 v7 xs. P xs ⇒ P (v6 orf v7::xs)) ∧
  (∀ v8 v9 xs. P xs ⇒ P (v8 impf v9::xs)) ∧
  (∀ v10 v11 xs. P xs ⇒ P (v10 eqf v11::xs)) ∧
  (∀ v12 xs. P xs ⇒ P (v12 says TT::xs)) ∧
  (∀ v12 xs. P xs ⇒ P (v12 says FF::xs)) ∧

```

$$\begin{aligned}
& (\forall v134 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (Name } v134 \text{ says prop NONE::xs)}) \wedge \\
& (\forall v144 \text{ } xs. \\
& \quad P \text{ } xs \Rightarrow \\
& \quad P \text{ (Name PlatoonLeader says prop (SOME } v144)::xs)}) \wedge \\
& (\forall v146 \text{ } xs. \\
& \quad P \text{ } xs \Rightarrow \\
& \quad P \\
& \quad \quad \text{(Name PlatoonSergeant says prop (SOME (ESCC } v146)::xs))} \wedge \\
& (\forall v150 \text{ } xs. \\
& \quad P \text{ } xs \Rightarrow \\
& \quad P \\
& \quad \quad \text{(Name PlatoonSergeant says} \\
& \quad \quad \text{prop (SOME (SLc (PL } v150)))::xs)}) \wedge \\
& (\forall v135 \text{ } v136 \text{ } v68 \text{ } xs. \\
& \quad P \text{ } xs \Rightarrow P \text{ (} v135 \text{ meet } v136 \text{ says prop } v68::xs)}) \wedge \\
& (\forall v137 \text{ } v138 \text{ } v68 \text{ } xs. \\
& \quad P \text{ } xs \Rightarrow P \text{ (} v137 \text{ quoting } v138 \text{ says prop } v68::xs)}) \wedge \\
& (\forall v12 \text{ } v69 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v12 \text{ says notf } v69::xs)}) \wedge \\
& (\forall v12 \text{ } v70 \text{ } v71 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v12 \text{ says (} v70 \text{ andf } v71)::xs)}) \wedge \\
& (\forall v12 \text{ } v72 \text{ } v73 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v12 \text{ says (} v72 \text{ orf } v73)::xs)}) \wedge \\
& (\forall v12 \text{ } v74 \text{ } v75 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v12 \text{ says (} v74 \text{ impf } v75)::xs)}) \wedge \\
& (\forall v12 \text{ } v76 \text{ } v77 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v12 \text{ says (} v76 \text{ eqf } v77)::xs)}) \wedge \\
& (\forall v12 \text{ } v78 \text{ } v79 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v12 \text{ says } v78 \text{ says } v79::xs)}) \wedge \\
& (\forall v12 \text{ } v80 \text{ } v81 \text{ } xs. \\
& \quad P \text{ } xs \Rightarrow P \text{ (} v12 \text{ says } v80 \text{ speaks_for } v81::xs)}) \wedge \\
& (\forall v12 \text{ } v82 \text{ } v83 \text{ } xs. \\
& \quad P \text{ } xs \Rightarrow P \text{ (} v12 \text{ says } v82 \text{ controls } v83::xs)}) \wedge \\
& (\forall v12 \text{ } v84 \text{ } v85 \text{ } v86 \text{ } xs. \\
& \quad P \text{ } xs \Rightarrow P \text{ (} v12 \text{ says reps } v84 \text{ } v85 \text{ } v86::xs)}) \wedge \\
& (\forall v12 \text{ } v87 \text{ } v88 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v12 \text{ says } v87 \text{ domi } v88::xs)}) \wedge \\
& (\forall v12 \text{ } v89 \text{ } v90 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v12 \text{ says } v89 \text{ eqi } v90::xs)}) \wedge \\
& (\forall v12 \text{ } v91 \text{ } v92 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v12 \text{ says } v91 \text{ doms } v92::xs)}) \wedge \\
& (\forall v12 \text{ } v93 \text{ } v94 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v12 \text{ says } v93 \text{ eqs } v94::xs)}) \wedge \\
& (\forall v12 \text{ } v95 \text{ } v96 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v12 \text{ says } v95 \text{ eqn } v96::xs)}) \wedge \\
& (\forall v12 \text{ } v97 \text{ } v98 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v12 \text{ says } v97 \text{ lte } v98::xs)}) \wedge \\
& (\forall v12 \text{ } v99 \text{ } v100 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v12 \text{ says } v99 \text{ lt } v100::xs)}) \wedge \\
& (\forall v14 \text{ } v15 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v14 \text{ speaks_for } v15::xs)}) \wedge \\
& (\forall v16 \text{ } v17 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v16 \text{ controls } v17::xs)}) \wedge \\
& (\forall v18 \text{ } v19 \text{ } v20 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (reps } v18 \text{ } v19 \text{ } v20::xs)}) \wedge \\
& (\forall v21 \text{ } v22 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v21 \text{ domi } v22::xs)}) \wedge \\
& (\forall v23 \text{ } v24 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v23 \text{ eqi } v24::xs)}) \wedge \\
& (\forall v25 \text{ } v26 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v25 \text{ doms } v26::xs)}) \wedge \\
& (\forall v27 \text{ } v28 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v27 \text{ eqs } v28::xs)}) \wedge \\
& (\forall v29 \text{ } v30 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v29 \text{ eqn } v30::xs)}) \wedge \\
& (\forall v31 \text{ } v32 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v31 \text{ lte } v32::xs)}) \wedge \\
& (\forall v33 \text{ } v34 \text{ } xs. P \text{ } xs \Rightarrow P \text{ (} v33 \text{ lt } v34::xs)}) \Rightarrow \\
& \forall v. P \text{ } v
\end{aligned}$$

[getRecon\_def]

$$\begin{aligned}
& \vdash (\text{getRecon } [] = [\text{NONE}]) \wedge \\
& (\forall xs. \\
& \quad \text{getRecon} \\
& \quad \quad (\text{Name PlatoonLeader says prop (SOME (SLc (PL recon))))::} \\
& \quad \quad \quad xs) = \\
& \quad [\text{SOME (SLc (PL recon))}] \wedge \\
& (\forall xs. \text{getRecon } (\text{TT}::xs) = \text{getRecon } xs) \wedge \\
& (\forall xs. \text{getRecon } (\text{FF}::xs) = \text{getRecon } xs) \wedge \\
& (\forall xs \ v_2. \text{getRecon } (\text{prop } v_2::xs) = \text{getRecon } xs) \wedge \\
& (\forall xs \ v_3. \text{getRecon } (\text{notf } v_3::xs) = \text{getRecon } xs) \wedge \\
& (\forall xs \ v_5 \ v_4. \text{getRecon } (v_4 \ \text{andf } v_5::xs) = \text{getRecon } xs) \wedge \\
& (\forall xs \ v_7 \ v_6. \text{getRecon } (v_6 \ \text{orf } v_7::xs) = \text{getRecon } xs) \wedge \\
& (\forall xs \ v_9 \ v_8. \text{getRecon } (v_8 \ \text{impf } v_9::xs) = \text{getRecon } xs) \wedge \\
& (\forall xs \ v_{11} \ v_{10}. \text{getRecon } (v_{10} \ \text{eqf } v_{11}::xs) = \text{getRecon } xs) \wedge \\
& (\forall xs \ v_{12}. \text{getRecon } (v_{12} \ \text{says TT}::xs) = \text{getRecon } xs) \wedge \\
& (\forall xs \ v_{12}. \text{getRecon } (v_{12} \ \text{says FF}::xs) = \text{getRecon } xs) \wedge \\
& (\forall xs \ v_{134}. \\
& \quad \text{getRecon } (\text{Name } v_{134} \ \text{says prop NONE}::xs) = \text{getRecon } xs) \wedge \\
& (\forall xs \ v_{146}. \\
& \quad \text{getRecon} \\
& \quad \quad (\text{Name PlatoonLeader says prop (SOME (ESCc } v_{146}))::xs) = \\
& \quad \quad \text{getRecon } xs) \wedge \\
& (\forall xs. \\
& \quad \text{getRecon} \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL receiveMission))))::xs} = \\
& \quad \quad \text{getRecon } xs) \wedge \\
& (\forall xs. \\
& \quad \text{getRecon} \\
& \quad \quad (\text{Name PlatoonLeader says prop (SOME (SLc (PL warno))))::} \\
& \quad \quad \quad xs) = \\
& \quad \text{getRecon } xs) \wedge \\
& (\forall xs. \\
& \quad \text{getRecon} \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL tentativePlan))))::xs} = \\
& \quad \quad \text{getRecon } xs) \wedge \\
& (\forall xs. \\
& \quad \text{getRecon} \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL report1))))::xs} = \\
& \quad \quad \text{getRecon } xs) \wedge \\
& (\forall xs. \\
& \quad \text{getRecon} \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL completePlan))))::xs} = \\
& \quad \quad \text{getRecon } xs) \wedge \\
& (\forall xs.
\end{aligned}$$

```

    getRecon
      (Name PlatoonLeader says prop (SOME (SLc (PL opoid))))::
        xs) =
    getRecon xs) ∧
  (∀ xs.
    getRecon
      (Name PlatoonLeader says
        prop (SOME (SLc (PL supervise))))::xs) =
    getRecon xs) ∧
  (∀ xs.
    getRecon
      (Name PlatoonLeader says
        prop (SOME (SLc (PL report2))))::xs) =
    getRecon xs) ∧
  (∀ xs.
    getRecon
      (Name PlatoonLeader says
        prop (SOME (SLc (PL complete))))::xs) =
    getRecon xs) ∧
  (∀ xs.
    getRecon
      (Name PlatoonLeader says
        prop (SOME (SLc (PL plIncomplete))))::xs) =
    getRecon xs) ∧
  (∀ xs.
    getRecon
      (Name PlatoonLeader says
        prop (SOME (SLc (PL invalidPlCommand))))::xs) =
    getRecon xs) ∧
  (∀ xs v151.
    getRecon
      (Name PlatoonLeader says prop (SOME (SLc (PSG v151))))::
        xs) =
    getRecon xs) ∧
  (∀ xs v144.
    getRecon
      (Name PlatoonSergeant says prop (SOME v144))::xs) =
    getRecon xs) ∧
  (∀ xs v68 v136 v135.
    getRecon (v135 meet v136 says prop v68::xs) =
    getRecon xs) ∧
  (∀ xs v68 v138 v137.
    getRecon (v137 quoting v138 says prop v68::xs) =
    getRecon xs) ∧
  (∀ xs v69 v12.
    getRecon (v12 says notf v69::xs) = getRecon xs) ∧
  (∀ xs v71 v70 v12.
    getRecon (v12 says (v70 andf v71)::xs) = getRecon xs) ∧
  (∀ xs v73 v72 v12.

```

```

    getRecon (v12 says (v72 orf v73)::xs) = getRecon xs) ∧
  (∀ xs v75 v74 v12.
    getRecon (v12 says (v74 impf v75)::xs) = getRecon xs) ∧
  (∀ xs v77 v76 v12.
    getRecon (v12 says (v76 eqf v77)::xs) = getRecon xs) ∧
  (∀ xs v79 v78 v12.
    getRecon (v12 says v78 says v79::xs) = getRecon xs) ∧
  (∀ xs v81 v80 v12.
    getRecon (v12 says v80 speaks_for v81::xs) =
    getRecon xs) ∧
  (∀ xs v83 v82 v12.
    getRecon (v12 says v82 controls v83::xs) = getRecon xs) ∧
  (∀ xs v86 v85 v84 v12.
    getRecon (v12 says reps v84 v85 v86::xs) = getRecon xs) ∧
  (∀ xs v88 v87 v12.
    getRecon (v12 says v87 domi v88::xs) = getRecon xs) ∧
  (∀ xs v90 v89 v12.
    getRecon (v12 says v89 eqi v90::xs) = getRecon xs) ∧
  (∀ xs v92 v91 v12.
    getRecon (v12 says v91 doms v92::xs) = getRecon xs) ∧
  (∀ xs v94 v93 v12.
    getRecon (v12 says v93 eqs v94::xs) = getRecon xs) ∧
  (∀ xs v96 v95 v12.
    getRecon (v12 says v95 eqn v96::xs) = getRecon xs) ∧
  (∀ xs v98 v97 v12.
    getRecon (v12 says v97 lte v98::xs) = getRecon xs) ∧
  (∀ xs v99 v12 v100.
    getRecon (v12 says v99 lt v100::xs) = getRecon xs) ∧
  (∀ xs v15 v14.
    getRecon (v14 speaks_for v15::xs) = getRecon xs) ∧
  (∀ xs v17 v16.
    getRecon (v16 controls v17::xs) = getRecon xs) ∧
  (∀ xs v20 v19 v18.
    getRecon (reps v18 v19 v20::xs) = getRecon xs) ∧
  (∀ xs v22 v21. getRecon (v21 domi v22::xs) = getRecon xs) ∧
  (∀ xs v24 v23. getRecon (v23 eqi v24::xs) = getRecon xs) ∧
  (∀ xs v26 v25. getRecon (v25 doms v26::xs) = getRecon xs) ∧
  (∀ xs v28 v27. getRecon (v27 eqs v28::xs) = getRecon xs) ∧
  (∀ xs v30 v29. getRecon (v29 eqn v30::xs) = getRecon xs) ∧
  (∀ xs v32 v31. getRecon (v31 lte v32::xs) = getRecon xs) ∧
  ∀ xs v34 v33. getRecon (v33 lt v34::xs) = getRecon xs

```

[getRecon\_ind]

```

⊢ ∀ P.
  P [] ∧
  (∀ xs.
    P
    (Name PlatoonLeader says
      prop (SOME (SLc (PL recon))))::xs)) ∧

```

$$\begin{aligned}
& (\forall xs. P \ xs \Rightarrow P \ (TT::xs)) \wedge (\forall xs. P \ xs \Rightarrow P \ (FF::xs)) \wedge \\
& (\forall v_2 \ xs. P \ xs \Rightarrow P \ (\text{prop } v_2::xs)) \wedge \\
& (\forall v_3 \ xs. P \ xs \Rightarrow P \ (\text{notf } v_3::xs)) \wedge \\
& (\forall v_4 \ v_5 \ xs. P \ xs \Rightarrow P \ (v_4 \ \text{andf } v_5::xs)) \wedge \\
& (\forall v_6 \ v_7 \ xs. P \ xs \Rightarrow P \ (v_6 \ \text{orf } v_7::xs)) \wedge \\
& (\forall v_8 \ v_9 \ xs. P \ xs \Rightarrow P \ (v_8 \ \text{impf } v_9::xs)) \wedge \\
& (\forall v_{10} \ v_{11} \ xs. P \ xs \Rightarrow P \ (v_{10} \ \text{eqf } v_{11}::xs)) \wedge \\
& (\forall v_{12} \ xs. P \ xs \Rightarrow P \ (v_{12} \ \text{says } TT::xs)) \wedge \\
& (\forall v_{12} \ xs. P \ xs \Rightarrow P \ (v_{12} \ \text{says } FF::xs)) \wedge \\
& (\forall v_{134} \ xs. P \ xs \Rightarrow P \ (\text{Name } v_{134} \ \text{says prop NONE::xs})) \wedge \\
& (\forall v_{146} \ xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad (\text{Name PlatoonLeader says prop (SOME (ESCc } v_{146})):: \\
& \quad \quad xs)) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop (SOME (SLc (PL receiveMission)))::xs})) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop (SOME (SLc (PL warno)))::xs})) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop (SOME (SLc (PL tentativePlan)))::xs})) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop (SOME (SLc (PL report1)))::xs})) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop (SOME (SLc (PL completePlan)))::xs})) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop (SOME (SLc (PL opoid)))::xs})) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad (\text{Name PlatoonLeader says}
\end{aligned}$$



$$\begin{aligned}
& \text{prop (SOME (SLc (PL supervise)))::xs))} \wedge \\
(\forall xs. & \\
& P \ xs \Rightarrow \\
& P \\
& \text{(Name PlatoonLeader says} \\
& \text{prop (SOME (SLc (PL report2)))::xs))} \wedge \\
(\forall xs. & \\
& P \ xs \Rightarrow \\
& P \\
& \text{(Name PlatoonLeader says} \\
& \text{prop (SOME (SLc (PL complete)))::xs))} \wedge \\
(\forall xs. & \\
& P \ xs \Rightarrow \\
& P \\
& \text{(Name PlatoonLeader says} \\
& \text{prop (SOME (SLc (PL plIncomplete)))::xs))} \wedge \\
(\forall xs. & \\
& P \ xs \Rightarrow \\
& P \\
& \text{(Name PlatoonLeader says} \\
& \text{prop (SOME (SLc (PL invalidPlCommand)))::xs))} \wedge \\
(\forall v151 \ xs. & \\
& P \ xs \Rightarrow \\
& P \\
& \text{(Name PlatoonLeader says} \\
& \text{prop (SOME (SLc (PSG v151)))::xs))} \wedge \\
(\forall v144 \ xs. & \\
& P \ xs \Rightarrow \\
& P \ \text{(Name PlatoonSergeant says prop (SOME v144)::xs))} \wedge \\
(\forall v135 \ v136 \ v_{68} \ xs. & \\
& P \ xs \Rightarrow P \ (v135 \ \text{meet} \ v136 \ \text{says} \ \text{prop} \ v_{68}::xs)) \wedge \\
(\forall v137 \ v138 \ v_{68} \ xs. & \\
& P \ xs \Rightarrow P \ (v137 \ \text{quoting} \ v138 \ \text{says} \ \text{prop} \ v_{68}::xs)) \wedge \\
(\forall v_{12} \ v_{69} \ xs. & P \ xs \Rightarrow P \ (v_{12} \ \text{says} \ \text{notf} \ v_{69}::xs)) \wedge \\
(\forall v_{12} \ v_{70} \ v_{71} \ xs. & P \ xs \Rightarrow P \ (v_{12} \ \text{says} \ (v_{70} \ \text{andf} \ v_{71})::xs)) \wedge \\
(\forall v_{12} \ v_{72} \ v_{73} \ xs. & P \ xs \Rightarrow P \ (v_{12} \ \text{says} \ (v_{72} \ \text{orf} \ v_{73})::xs)) \wedge \\
(\forall v_{12} \ v_{74} \ v_{75} \ xs. & P \ xs \Rightarrow P \ (v_{12} \ \text{says} \ (v_{74} \ \text{impf} \ v_{75})::xs)) \wedge \\
(\forall v_{12} \ v_{76} \ v_{77} \ xs. & P \ xs \Rightarrow P \ (v_{12} \ \text{says} \ (v_{76} \ \text{eqf} \ v_{77})::xs)) \wedge \\
(\forall v_{12} \ v_{78} \ v_{79} \ xs. & P \ xs \Rightarrow P \ (v_{12} \ \text{says} \ v_{78} \ \text{says} \ v_{79}::xs)) \wedge \\
(\forall v_{12} \ v_{80} \ v_{81} \ xs. & \\
& P \ xs \Rightarrow P \ (v_{12} \ \text{says} \ v_{80} \ \text{speaks\_for} \ v_{81}::xs)) \wedge \\
(\forall v_{12} \ v_{82} \ v_{83} \ xs. & \\
& P \ xs \Rightarrow P \ (v_{12} \ \text{says} \ v_{82} \ \text{controls} \ v_{83}::xs)) \wedge \\
(\forall v_{12} \ v_{84} \ v_{85} \ v_{86} \ xs. & \\
& P \ xs \Rightarrow P \ (v_{12} \ \text{says} \ \text{reps} \ v_{84} \ v_{85} \ v_{86}::xs)) \wedge \\
(\forall v_{12} \ v_{87} \ v_{88} \ xs. & P \ xs \Rightarrow P \ (v_{12} \ \text{says} \ v_{87} \ \text{domi} \ v_{88}::xs)) \wedge \\
(\forall v_{12} \ v_{89} \ v_{90} \ xs. & P \ xs \Rightarrow P \ (v_{12} \ \text{says} \ v_{89} \ \text{eqi} \ v_{90}::xs)) \wedge \\
(\forall v_{12} \ v_{91} \ v_{92} \ xs. & P \ xs \Rightarrow P \ (v_{12} \ \text{says} \ v_{91} \ \text{doms} \ v_{92}::xs)) \wedge \\
(\forall v_{12} \ v_{93} \ v_{94} \ xs. & P \ xs \Rightarrow P \ (v_{12} \ \text{says} \ v_{93} \ \text{eqs} \ v_{94}::xs)) \wedge
\end{aligned}$$

$$\begin{aligned}
& (\forall v_{12} v_{95} v_{96} xs. P xs \Rightarrow P (v_{12} \text{ says } v_{95} \text{ eqn } v_{96} :: xs)) \wedge \\
& (\forall v_{12} v_{97} v_{98} xs. P xs \Rightarrow P (v_{12} \text{ says } v_{97} \text{ lte } v_{98} :: xs)) \wedge \\
& (\forall v_{12} v_{99} v_{100} xs. P xs \Rightarrow P (v_{12} \text{ says } v_{99} \text{ lt } v_{100} :: xs)) \wedge \\
& (\forall v_{14} v_{15} xs. P xs \Rightarrow P (v_{14} \text{ speaks\_for } v_{15} :: xs)) \wedge \\
& (\forall v_{16} v_{17} xs. P xs \Rightarrow P (v_{16} \text{ controls } v_{17} :: xs)) \wedge \\
& (\forall v_{18} v_{19} v_{20} xs. P xs \Rightarrow P (\text{reps } v_{18} v_{19} v_{20} :: xs)) \wedge \\
& (\forall v_{21} v_{22} xs. P xs \Rightarrow P (v_{21} \text{ domi } v_{22} :: xs)) \wedge \\
& (\forall v_{23} v_{24} xs. P xs \Rightarrow P (v_{23} \text{ eqi } v_{24} :: xs)) \wedge \\
& (\forall v_{25} v_{26} xs. P xs \Rightarrow P (v_{25} \text{ doms } v_{26} :: xs)) \wedge \\
& (\forall v_{27} v_{28} xs. P xs \Rightarrow P (v_{27} \text{ eqs } v_{28} :: xs)) \wedge \\
& (\forall v_{29} v_{30} xs. P xs \Rightarrow P (v_{29} \text{ eqn } v_{30} :: xs)) \wedge \\
& (\forall v_{31} v_{32} xs. P xs \Rightarrow P (v_{31} \text{ lte } v_{32} :: xs)) \wedge \\
& (\forall v_{33} v_{34} xs. P xs \Rightarrow P (v_{33} \text{ lt } v_{34} :: xs)) \Rightarrow \\
& \forall v. P v
\end{aligned}$$

[getReport\_def]

$$\begin{aligned}
& \vdash (\text{getReport } [] = [\text{NONE}]) \wedge \\
& (\forall xs. \\
& \quad \text{getReport} \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL report1))) :: xs} = \\
& \quad \quad \quad [\text{SOME (SLc (PL report1))}]) \wedge \\
& \quad (\forall xs. \text{getReport (TT :: xs)} = \text{getReport } xs) \wedge \\
& \quad (\forall xs. \text{getReport (FF :: xs)} = \text{getReport } xs) \wedge \\
& \quad (\forall xs v_2. \text{getReport (prop } v_2 :: xs) = \text{getReport } xs) \wedge \\
& \quad (\forall xs v_3. \text{getReport (notf } v_3 :: xs) = \text{getReport } xs) \wedge \\
& \quad (\forall xs v_5 v_4. \text{getReport (} v_4 \text{ andf } v_5 :: xs) = \text{getReport } xs) \wedge \\
& \quad (\forall xs v_7 v_6. \text{getReport (} v_6 \text{ orf } v_7 :: xs) = \text{getReport } xs) \wedge \\
& \quad (\forall xs v_9 v_8. \text{getReport (} v_8 \text{ impf } v_9 :: xs) = \text{getReport } xs) \wedge \\
& \quad (\forall xs v_{11} v_{10}. \text{getReport (} v_{10} \text{ eqf } v_{11} :: xs) = \text{getReport } xs) \wedge \\
& \quad (\forall xs v_{12}. \text{getReport (} v_{12} \text{ says TT :: xs) = getReport } xs) \wedge \\
& \quad (\forall xs v_{12}. \text{getReport (} v_{12} \text{ says FF :: xs) = getReport } xs) \wedge \\
& \quad (\forall xs v_{134}. \\
& \quad \quad \text{getReport (Name } v_{134} \text{ says prop NONE :: xs) = getReport } xs) \wedge \\
& \quad (\forall xs v_{146}. \\
& \quad \quad \text{getReport} \\
& \quad \quad \quad (\text{Name PlatoonLeader says prop (SOME (ESCc } v_{146})) :: xs} = \\
& \quad \quad \quad \text{getReport } xs) \wedge \\
& \quad (\forall xs. \\
& \quad \quad \text{getReport} \\
& \quad \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \quad \text{prop (SOME (SLc (PL receiveMission))) :: xs} = \\
& \quad \quad \quad \text{getReport } xs) \wedge \\
& \quad (\forall xs. \\
& \quad \quad \text{getReport} \\
& \quad \quad \quad (\text{Name PlatoonLeader says prop (SOME (SLc (PL warno))) ::} \\
& \quad \quad \quad \quad xs) = \\
& \quad \quad \quad \text{getReport } xs) \wedge \\
& \quad (\forall xs.
\end{aligned}$$

```

    getReport
      (Name PlatoonLeader says
        prop (SOME (SLc (PL tentativePlan))))::xs) =
    getReport xs) ∧
  (∀ xs.
    getReport
      (Name PlatoonLeader says prop (SOME (SLc (PL recon))))::
        xs) =
    getReport xs) ∧
  (∀ xs.
    getReport
      (Name PlatoonLeader says
        prop (SOME (SLc (PL completePlan))))::xs) =
    getReport xs) ∧
  (∀ xs.
    getReport
      (Name PlatoonLeader says prop (SOME (SLc (PL opoid))))::
        xs) =
    getReport xs) ∧
  (∀ xs.
    getReport
      (Name PlatoonLeader says
        prop (SOME (SLc (PL supervise))))::xs) =
    getReport xs) ∧
  (∀ xs.
    getReport
      (Name PlatoonLeader says
        prop (SOME (SLc (PL report2))))::xs) =
    getReport xs) ∧
  (∀ xs.
    getReport
      (Name PlatoonLeader says
        prop (SOME (SLc (PL complete))))::xs) =
    getReport xs) ∧
  (∀ xs.
    getReport
      (Name PlatoonLeader says
        prop (SOME (SLc (PL plIncomplete))))::xs) =
    getReport xs) ∧
  (∀ xs.
    getReport
      (Name PlatoonLeader says
        prop (SOME (SLc (PL invalidPlCommand))))::xs) =
    getReport xs) ∧
  (∀ xs v151.
    getReport
      (Name PlatoonLeader says prop (SOME (SLc (PSG v151))))::
        xs) =
    getReport xs) ∧

```

---

```

(∀ xs v144.
  getReport
    (Name PlatoonSergeant says prop (SOME v144))::xs) =
  getReport xs) ∧
(∀ xs v68 v136 v135.
  getReport (v135 meet v136 says prop v68::xs) =
  getReport xs) ∧
(∀ xs v68 v138 v137.
  getReport (v137 quoting v138 says prop v68::xs) =
  getReport xs) ∧
(∀ xs v69 v12.
  getReport (v12 says notf v69::xs) = getReport xs) ∧
(∀ xs v71 v70 v12.
  getReport (v12 says (v70 andf v71))::xs) = getReport xs) ∧
(∀ xs v73 v72 v12.
  getReport (v12 says (v72 orf v73))::xs) = getReport xs) ∧
(∀ xs v75 v74 v12.
  getReport (v12 says (v74 impf v75))::xs) = getReport xs) ∧
(∀ xs v77 v76 v12.
  getReport (v12 says (v76 eqf v77))::xs) = getReport xs) ∧
(∀ xs v79 v78 v12.
  getReport (v12 says v78 says v79::xs) = getReport xs) ∧
(∀ xs v81 v80 v12.
  getReport (v12 says v80 speaks_for v81::xs) =
  getReport xs) ∧
(∀ xs v83 v82 v12.
  getReport (v12 says v82 controls v83::xs) =
  getReport xs) ∧
(∀ xs v86 v85 v84 v12.
  getReport (v12 says reps v84 v85 v86::xs) =
  getReport xs) ∧
(∀ xs v88 v87 v12.
  getReport (v12 says v87 domi v88::xs) = getReport xs) ∧
(∀ xs v90 v89 v12.
  getReport (v12 says v89 eqi v90::xs) = getReport xs) ∧
(∀ xs v92 v91 v12.
  getReport (v12 says v91 doms v92::xs) = getReport xs) ∧
(∀ xs v94 v93 v12.
  getReport (v12 says v93 eqs v94::xs) = getReport xs) ∧
(∀ xs v96 v95 v12.
  getReport (v12 says v95 eqn v96::xs) = getReport xs) ∧
(∀ xs v98 v97 v12.
  getReport (v12 says v97 lte v98::xs) = getReport xs) ∧
(∀ xs v99 v12 v100.
  getReport (v12 says v99 lt v100::xs) = getReport xs) ∧
(∀ xs v15 v14.
  getReport (v14 speaks_for v15::xs) = getReport xs) ∧
(∀ xs v17 v16.
  getReport (v16 controls v17::xs) = getReport xs) ∧

```

---

$$\begin{aligned}
& (\forall xs \ v_{20} \ v_{19} \ v_{18}. \\
& \quad \text{getReport (reps } v_{18} \ v_{19} \ v_{20} :: xs) = \text{getReport } xs) \wedge \\
& (\forall xs \ v_{22} \ v_{21}. \text{getReport (} v_{21} \text{ domi } v_{22} :: xs) = \text{getReport } xs) \wedge \\
& (\forall xs \ v_{24} \ v_{23}. \text{getReport (} v_{23} \text{ eqi } v_{24} :: xs) = \text{getReport } xs) \wedge \\
& (\forall xs \ v_{26} \ v_{25}. \text{getReport (} v_{25} \text{ doms } v_{26} :: xs) = \text{getReport } xs) \wedge \\
& (\forall xs \ v_{28} \ v_{27}. \text{getReport (} v_{27} \text{ eqs } v_{28} :: xs) = \text{getReport } xs) \wedge \\
& (\forall xs \ v_{30} \ v_{29}. \text{getReport (} v_{29} \text{ eqn } v_{30} :: xs) = \text{getReport } xs) \wedge \\
& (\forall xs \ v_{32} \ v_{31}. \text{getReport (} v_{31} \text{ lte } v_{32} :: xs) = \text{getReport } xs) \wedge \\
& \forall xs \ v_{34} \ v_{33}. \text{getReport (} v_{33} \text{ lt } v_{34} :: xs) = \text{getReport } xs
\end{aligned}$$

[getReport\_ind]

$$\begin{aligned}
& \vdash \forall P. \\
& \quad P \ \square \ \wedge \\
& \quad (\forall xs. \\
& \quad \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL report1))) :: xs}) \wedge \\
& \quad \quad (\forall xs. P \ xs \Rightarrow P \ (\text{TT} :: xs)) \wedge (\forall xs. P \ xs \Rightarrow P \ (\text{FF} :: xs)) \wedge \\
& \quad \quad (\forall v_2 \ xs. P \ xs \Rightarrow P \ (\text{prop } v_2 :: xs)) \wedge \\
& \quad \quad (\forall v_3 \ xs. P \ xs \Rightarrow P \ (\text{notf } v_3 :: xs)) \wedge \\
& \quad \quad (\forall v_4 \ v_5 \ xs. P \ xs \Rightarrow P \ (v_4 \ \text{andf } v_5 :: xs)) \wedge \\
& \quad \quad (\forall v_6 \ v_7 \ xs. P \ xs \Rightarrow P \ (v_6 \ \text{orf } v_7 :: xs)) \wedge \\
& \quad \quad (\forall v_8 \ v_9 \ xs. P \ xs \Rightarrow P \ (v_8 \ \text{impf } v_9 :: xs)) \wedge \\
& \quad \quad (\forall v_{10} \ v_{11} \ xs. P \ xs \Rightarrow P \ (v_{10} \ \text{eqf } v_{11} :: xs)) \wedge \\
& \quad \quad (\forall v_{12} \ xs. P \ xs \Rightarrow P \ (v_{12} \ \text{says TT} :: xs)) \wedge \\
& \quad \quad (\forall v_{12} \ xs. P \ xs \Rightarrow P \ (v_{12} \ \text{says FF} :: xs)) \wedge \\
& \quad \quad (\forall v_{134} \ xs. P \ xs \Rightarrow P \ (\text{Name } v_{134} \ \text{says prop NONE} :: xs)) \wedge \\
& \quad \quad (\forall v_{146} \ xs. \\
& \quad \quad \quad P \ xs \Rightarrow \\
& \quad \quad \quad P \\
& \quad \quad \quad (\text{Name PlatoonLeader says prop (SOME (ESCc } v_{146})) :: \\
& \quad \quad \quad \quad xs)) \wedge \\
& \quad \quad (\forall xs. \\
& \quad \quad \quad P \ xs \Rightarrow \\
& \quad \quad \quad P \\
& \quad \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \quad \text{prop (SOME (SLc (PL receiveMission))) :: xs}) \wedge \\
& \quad \quad (\forall xs. \\
& \quad \quad \quad P \ xs \Rightarrow \\
& \quad \quad \quad P \\
& \quad \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \quad \text{prop (SOME (SLc (PL warno))) :: xs}) \wedge \\
& \quad \quad (\forall xs. \\
& \quad \quad \quad P \ xs \Rightarrow \\
& \quad \quad \quad P \\
& \quad \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \quad \text{prop (SOME (SLc (PL tentativePlan))) :: xs}) \wedge \\
& \quad \quad (\forall xs. \\
& \quad \quad \quad P \ xs \Rightarrow
\end{aligned}$$

$$\begin{aligned}
& P \\
& \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop (SOME (SLc (PL recon)))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL completePlan)))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL opoid)))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL supervise)))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL report2)))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL complete)))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL plIncomplete)))::xs}) \wedge \\
& (\forall v151 \ xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PSG v151)))::xs}) \wedge \\
& (\forall v144 \ xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \ (\text{Name PlatoonSergeant says prop (SOME v144)::xs}) \wedge \\
& (\forall v135 \ v136 \ v_{68} \ xs. \\
& \quad P \ xs \Rightarrow P \ (v135 \text{ meet } v136 \text{ says prop } v_{68}::xs) \wedge \\
& (\forall v137 \ v138 \ v_{68} \ xs.
\end{aligned}$$

$$\begin{aligned}
& P \text{ } xs \Rightarrow P \text{ } (v137 \text{ quoting } v138 \text{ says prop } v68::xs)) \wedge \\
& (\forall v_{12} \ v_{69} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says notf } v_{69}::xs)) \wedge \\
& (\forall v_{12} \ v_{70} \ v_{71} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } (v_{70} \text{ andf } v_{71})::xs)) \wedge \\
& (\forall v_{12} \ v_{72} \ v_{73} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } (v_{72} \text{ orf } v_{73})::xs)) \wedge \\
& (\forall v_{12} \ v_{74} \ v_{75} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } (v_{74} \text{ impf } v_{75})::xs)) \wedge \\
& (\forall v_{12} \ v_{76} \ v_{77} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } (v_{76} \text{ eqf } v_{77})::xs)) \wedge \\
& (\forall v_{12} \ v_{78} \ v_{79} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{78} \text{ says } v_{79}::xs)) \wedge \\
& (\forall v_{12} \ v_{80} \ v_{81} \ xs. \\
& \quad P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{80} \text{ speaks\_for } v_{81}::xs)) \wedge \\
& (\forall v_{12} \ v_{82} \ v_{83} \ xs. \\
& \quad P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{82} \text{ controls } v_{83}::xs)) \wedge \\
& (\forall v_{12} \ v_{84} \ v_{85} \ v_{86} \ xs. \\
& \quad P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says reps } v_{84} \ v_{85} \ v_{86}::xs)) \wedge \\
& (\forall v_{12} \ v_{87} \ v_{88} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{87} \text{ domi } v_{88}::xs)) \wedge \\
& (\forall v_{12} \ v_{89} \ v_{90} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{89} \text{ eqi } v_{90}::xs)) \wedge \\
& (\forall v_{12} \ v_{91} \ v_{92} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{91} \text{ doms } v_{92}::xs)) \wedge \\
& (\forall v_{12} \ v_{93} \ v_{94} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{93} \text{ eqs } v_{94}::xs)) \wedge \\
& (\forall v_{12} \ v_{95} \ v_{96} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{95} \text{ eqn } v_{96}::xs)) \wedge \\
& (\forall v_{12} \ v_{97} \ v_{98} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{97} \text{ lte } v_{98}::xs)) \wedge \\
& (\forall v_{12} \ v_{99} \ v100 \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{99} \text{ lt } v100::xs)) \wedge \\
& (\forall v_{14} \ v_{15} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{14} \text{ speaks\_for } v_{15}::xs)) \wedge \\
& (\forall v_{16} \ v_{17} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{16} \text{ controls } v_{17}::xs)) \wedge \\
& (\forall v_{18} \ v_{19} \ v_{20} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{18} \text{ reps } v_{19} \ v_{20}::xs)) \wedge \\
& (\forall v_{21} \ v_{22} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{21} \text{ domi } v_{22}::xs)) \wedge \\
& (\forall v_{23} \ v_{24} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{23} \text{ eqi } v_{24}::xs)) \wedge \\
& (\forall v_{25} \ v_{26} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{25} \text{ doms } v_{26}::xs)) \wedge \\
& (\forall v_{27} \ v_{28} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{27} \text{ eqs } v_{28}::xs)) \wedge \\
& (\forall v_{29} \ v_{30} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{29} \text{ eqn } v_{30}::xs)) \wedge \\
& (\forall v_{31} \ v_{32} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{31} \text{ lte } v_{32}::xs)) \wedge \\
& (\forall v_{33} \ v_{34} \ xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{33} \text{ lt } v_{34}::xs)) \Rightarrow \\
& \forall v. \ P \ v
\end{aligned}$$

[getTenativePlan\_def]

$$\begin{aligned}
& \vdash (\text{getTenativePlan } [] = [\text{NONE}]) \wedge \\
& (\forall xs. \\
& \quad \text{getTenativePlan} \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL tentativePlan)))::xs}) = \\
& \quad \quad \text{[SOME (SLc (PL tentativePlan))])} \wedge \\
& (\forall xs. \text{getTenativePlan } (\text{TT}::xs) = \text{getTenativePlan } xs) \wedge \\
& (\forall xs. \text{getTenativePlan } (\text{FF}::xs) = \text{getTenativePlan } xs) \wedge \\
& (\forall xs \ v_2. \\
& \quad \text{getTenativePlan } (\text{prop } v_2::xs) = \text{getTenativePlan } xs) \wedge \\
& (\forall xs \ v_3. \\
& \quad \text{getTenativePlan } (\text{notf } v_3::xs) = \text{getTenativePlan } xs) \wedge \\
& (\forall xs \ v_5 \ v_4. \\
& \quad \text{getTenativePlan } (v_4 \text{ andf } v_5::xs) = \text{getTenativePlan } xs) \wedge \\
& (\forall xs \ v_7 \ v_6. \\
& \quad \text{getTenativePlan } (v_6 \text{ orf } v_7::xs) = \text{getTenativePlan } xs) \wedge
\end{aligned}$$

---

```

(∀ xs v9 v8.
  getTenativePlan (v8 impf v9::xs) = getTenativePlan xs) ∧
(∀ xs v11 v10.
  getTenativePlan (v10 eqf v11::xs) = getTenativePlan xs) ∧
(∀ xs v12.
  getTenativePlan (v12 says TT::xs) = getTenativePlan xs) ∧
(∀ xs v12.
  getTenativePlan (v12 says FF::xs) = getTenativePlan xs) ∧
(∀ xs v134.
  getTenativePlan (Name v134 says prop NONE::xs) =
  getTenativePlan xs) ∧
(∀ xs v146.
  getTenativePlan
    (Name PlatoonLeader says prop (SOME (ESCc v146))::xs) =
  getTenativePlan xs) ∧
(∀ xs.
  getTenativePlan
    (Name PlatoonLeader says
      prop (SOME (SLc (PL receiveMission)))::xs) =
  getTenativePlan xs) ∧
(∀ xs.
  getTenativePlan
    (Name PlatoonLeader says prop (SOME (SLc (PL warno)))::
      xs) =
  getTenativePlan xs) ∧
(∀ xs.
  getTenativePlan
    (Name PlatoonLeader says prop (SOME (SLc (PL recon)))::
      xs) =
  getTenativePlan xs) ∧
(∀ xs.
  getTenativePlan
    (Name PlatoonLeader says
      prop (SOME (SLc (PL report1)))::xs) =
  getTenativePlan xs) ∧
(∀ xs.
  getTenativePlan
    (Name PlatoonLeader says
      prop (SOME (SLc (PL completePlan)))::xs) =
  getTenativePlan xs) ∧
(∀ xs.
  getTenativePlan
    (Name PlatoonLeader says prop (SOME (SLc (PL opoid)))::
      xs) =
  getTenativePlan xs) ∧
(∀ xs.
  getTenativePlan
    (Name PlatoonLeader says
      prop (SOME (SLc (PL supervise)))::xs) =

```

---



---

```

    getTenativePlan xs) ∧
(∀ xs.
  getTenativePlan
    (Name PlatoonLeader says
      prop (SOME (SLc (PL report2))))::xs) =
  getTenativePlan xs) ∧
(∀ xs.
  getTenativePlan
    (Name PlatoonLeader says
      prop (SOME (SLc (PL complete))))::xs) =
  getTenativePlan xs) ∧
(∀ xs.
  getTenativePlan
    (Name PlatoonLeader says
      prop (SOME (SLc (PL plIncomplete))))::xs) =
  getTenativePlan xs) ∧
(∀ xs.
  getTenativePlan
    (Name PlatoonLeader says
      prop (SOME (SLc (PL invalidPlCommand))))::xs) =
  getTenativePlan xs) ∧
(∀ xs v151.
  getTenativePlan
    (Name PlatoonLeader says prop (SOME (SLc (PSG v151))))::
    xs) =
  getTenativePlan xs) ∧
(∀ xs v144.
  getTenativePlan
    (Name PlatoonSergeant says prop (SOME v144))::xs) =
  getTenativePlan xs) ∧
(∀ xs v68 v136 v135.
  getTenativePlan (v135 meet v136 says prop v68::xs) =
  getTenativePlan xs) ∧
(∀ xs v68 v138 v137.
  getTenativePlan (v137 quoting v138 says prop v68::xs) =
  getTenativePlan xs) ∧
(∀ xs v69 v12.
  getTenativePlan (v12 says notf v69::xs) =
  getTenativePlan xs) ∧
(∀ xs v71 v70 v12.
  getTenativePlan (v12 says (v70 andf v71)::xs) =
  getTenativePlan xs) ∧
(∀ xs v73 v72 v12.
  getTenativePlan (v12 says (v72 orf v73)::xs) =
  getTenativePlan xs) ∧
(∀ xs v75 v74 v12.
  getTenativePlan (v12 says (v74 impf v75)::xs) =
  getTenativePlan xs) ∧
(∀ xs v77 v76 v12.

```

---

```

    getTentativePlan (v12 says (v76 eqf v77 :: xs) =
    getTentativePlan xs) ∧
  (∀ xs v79 v78 v12.
    getTentativePlan (v12 says v78 says v79 :: xs) =
    getTentativePlan xs) ∧
  (∀ xs v81 v80 v12.
    getTentativePlan (v12 says v80 speaks_for v81 :: xs) =
    getTentativePlan xs) ∧
  (∀ xs v83 v82 v12.
    getTentativePlan (v12 says v82 controls v83 :: xs) =
    getTentativePlan xs) ∧
  (∀ xs v86 v85 v84 v12.
    getTentativePlan (v12 says reps v84 v85 v86 :: xs) =
    getTentativePlan xs) ∧
  (∀ xs v88 v87 v12.
    getTentativePlan (v12 says v87 domi v88 :: xs) =
    getTentativePlan xs) ∧
  (∀ xs v90 v89 v12.
    getTentativePlan (v12 says v89 eqi v90 :: xs) =
    getTentativePlan xs) ∧
  (∀ xs v92 v91 v12.
    getTentativePlan (v12 says v91 doms v92 :: xs) =
    getTentativePlan xs) ∧
  (∀ xs v94 v93 v12.
    getTentativePlan (v12 says v93 eqs v94 :: xs) =
    getTentativePlan xs) ∧
  (∀ xs v96 v95 v12.
    getTentativePlan (v12 says v95 eqn v96 :: xs) =
    getTentativePlan xs) ∧
  (∀ xs v98 v97 v12.
    getTentativePlan (v12 says v97 lte v98 :: xs) =
    getTentativePlan xs) ∧
  (∀ xs v99 v12 v100.
    getTentativePlan (v12 says v99 lt v100 :: xs) =
    getTentativePlan xs) ∧
  (∀ xs v15 v14.
    getTentativePlan (v14 speaks_for v15 :: xs) =
    getTentativePlan xs) ∧
  (∀ xs v17 v16.
    getTentativePlan (v16 controls v17 :: xs) =
    getTentativePlan xs) ∧
  (∀ xs v20 v19 v18.
    getTentativePlan (reps v18 v19 v20 :: xs) =
    getTentativePlan xs) ∧
  (∀ xs v22 v21.
    getTentativePlan (v21 domi v22 :: xs) = getTentativePlan xs) ∧
  (∀ xs v24 v23.
    getTentativePlan (v23 eqi v24 :: xs) = getTentativePlan xs) ∧
  (∀ xs v26 v25.

```

```

    getTenativePlan (v25 doms v26::xs) = getTenativePlan xs) ∧
(∀ xs v28 v27.
    getTenativePlan (v27 eqs v28::xs) = getTenativePlan xs) ∧
(∀ xs v30 v29.
    getTenativePlan (v29 eqn v30::xs) = getTenativePlan xs) ∧
(∀ xs v32 v31.
    getTenativePlan (v31 lte v32::xs) = getTenativePlan xs) ∧
∀ xs v34 v33.
    getTenativePlan (v33 lt v34::xs) = getTenativePlan xs

```

[getTenativePlan\_ind]

```

⊢ ∀ P.
  P [] ∧
  (∀ xs.
    P
      (Name PlatoonLeader says
        prop (SOME (SLc (PL tentativePlan)))::xs)) ∧
    (∀ xs. P xs ⇒ P (TT::xs)) ∧ (∀ xs. P xs ⇒ P (FF::xs)) ∧
    (∀ v2 xs. P xs ⇒ P (prop v2::xs)) ∧
    (∀ v3 xs. P xs ⇒ P (notf v3::xs)) ∧
    (∀ v4 v5 xs. P xs ⇒ P (v4 andf v5::xs)) ∧
    (∀ v6 v7 xs. P xs ⇒ P (v6 orf v7::xs)) ∧
    (∀ v8 v9 xs. P xs ⇒ P (v8 impf v9::xs)) ∧
    (∀ v10 v11 xs. P xs ⇒ P (v10 eqf v11::xs)) ∧
    (∀ v12 xs. P xs ⇒ P (v12 says TT::xs)) ∧
    (∀ v12 xs. P xs ⇒ P (v12 says FF::xs)) ∧
    (∀ v134 xs. P xs ⇒ P (Name v134 says prop NONE::xs)) ∧
    (∀ v146 xs.
      P xs ⇒
      P
        (Name PlatoonLeader says prop (SOME (ESCc v146))::
          xs)) ∧
  (∀ xs.
    P xs ⇒
    P
      (Name PlatoonLeader says
        prop (SOME (SLc (PL receiveMission)))::xs)) ∧
  (∀ xs.
    P xs ⇒
    P
      (Name PlatoonLeader says
        prop (SOME (SLc (PL warno)))::xs)) ∧
  (∀ xs.
    P xs ⇒
    P
      (Name PlatoonLeader says
        prop (SOME (SLc (PL recon)))::xs)) ∧
  (∀ xs.
    P xs ⇒

```

$$\begin{aligned}
& P \\
& \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop (SOME (SLc (PL report1))))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \quad xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL completePlan))))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \quad xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL opoid))))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \quad xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL supervise))))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \quad xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL report2))))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \quad xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL complete))))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \quad xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL plIncomplete))))::xs}) \wedge \\
& (\forall v151 \quad xs. \\
& \quad P \quad xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PL invalidPlCommand))))::xs}) \wedge \\
& (\forall v151 \quad xs. \\
& \quad P \quad xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop (SOME (SLc (PSG v151))))::xs}) \wedge \\
& (\forall v144 \quad xs. \\
& \quad P \quad xs \Rightarrow \\
& \quad P \quad (\text{Name PlatoonSergeant says prop (SOME v144)::xs}) \wedge \\
& (\forall v135 \quad v136 \quad v_{68} \quad xs. \\
& \quad P \quad xs \Rightarrow P \quad (v135 \text{ meet } v136 \text{ says prop } v_{68}::xs) \wedge \\
& (\forall v137 \quad v138 \quad v_{68} \quad xs.
\end{aligned}$$

$$\begin{aligned}
& P \text{ } xs \Rightarrow P \text{ } (v137 \text{ quoting } v138 \text{ says prop } v68::xs)) \wedge \\
& (\forall v_{12} v_{69} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says notf } v_{69}::xs)) \wedge \\
& (\forall v_{12} v_{70} v_{71} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } (v_{70} \text{ andf } v_{71})::xs)) \wedge \\
& (\forall v_{12} v_{72} v_{73} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } (v_{72} \text{ orf } v_{73})::xs)) \wedge \\
& (\forall v_{12} v_{74} v_{75} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } (v_{74} \text{ impf } v_{75})::xs)) \wedge \\
& (\forall v_{12} v_{76} v_{77} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } (v_{76} \text{ eqf } v_{77})::xs)) \wedge \\
& (\forall v_{12} v_{78} v_{79} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{78} \text{ says } v_{79}::xs)) \wedge \\
& (\forall v_{12} v_{80} v_{81} \text{ } xs. \\
& \quad P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{80} \text{ speaks\_for } v_{81}::xs)) \wedge \\
& (\forall v_{12} v_{82} v_{83} \text{ } xs. \\
& \quad P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{82} \text{ controls } v_{83}::xs)) \wedge \\
& (\forall v_{12} v_{84} v_{85} v_{86} \text{ } xs. \\
& \quad P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says reps } v_{84} v_{85} v_{86}::xs)) \wedge \\
& (\forall v_{12} v_{87} v_{88} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{87} \text{ domi } v_{88}::xs)) \wedge \\
& (\forall v_{12} v_{89} v_{90} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{89} \text{ eqi } v_{90}::xs)) \wedge \\
& (\forall v_{12} v_{91} v_{92} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{91} \text{ doms } v_{92}::xs)) \wedge \\
& (\forall v_{12} v_{93} v_{94} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{93} \text{ eqs } v_{94}::xs)) \wedge \\
& (\forall v_{12} v_{95} v_{96} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{95} \text{ eqn } v_{96}::xs)) \wedge \\
& (\forall v_{12} v_{97} v_{98} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{97} \text{ lte } v_{98}::xs)) \wedge \\
& (\forall v_{12} v_{99} v100 \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{99} \text{ lt } v100::xs)) \wedge \\
& (\forall v_{14} v_{15} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{14} \text{ speaks\_for } v_{15}::xs)) \wedge \\
& (\forall v_{16} v_{17} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{16} \text{ controls } v_{17}::xs)) \wedge \\
& (\forall v_{18} v_{19} v_{20} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{18} \text{ reps } v_{19} v_{20}::xs)) \wedge \\
& (\forall v_{21} v_{22} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{21} \text{ domi } v_{22}::xs)) \wedge \\
& (\forall v_{23} v_{24} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{23} \text{ eqi } v_{24}::xs)) \wedge \\
& (\forall v_{25} v_{26} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{25} \text{ doms } v_{26}::xs)) \wedge \\
& (\forall v_{27} v_{28} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{27} \text{ eqs } v_{28}::xs)) \wedge \\
& (\forall v_{29} v_{30} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{29} \text{ eqn } v_{30}::xs)) \wedge \\
& (\forall v_{31} v_{32} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{31} \text{ lte } v_{32}::xs)) \wedge \\
& (\forall v_{33} v_{34} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{33} \text{ lt } v_{34}::xs)) \Rightarrow \\
& \forall v. P \text{ } v
\end{aligned}$$



## Index

### **ConductORPType Theory**, 33

Datatypes, 33

Theorems, 34

plCommand\_distinct\_clauses, 34

psgCommand\_distinct\_clauses, 34

slCommand\_distinct\_clauses, 34

slCommand\_one\_one, 34

slOutput\_distinct\_clauses, 34

slRole\_distinct\_clauses, 34

slState\_distinct\_clauses, 34

### **ConductPBType Theory**, 40

Datatypes, 40

Theorems, 40

plCommandPB\_distinct\_clauses, 40

psgCommandPB\_distinct\_clauses, 40

slCommand\_distinct\_clauses, 41

slCommand\_one\_one, 41

slOutput\_distinct\_clauses, 41

slRole\_distinct\_clauses, 41

slState\_distinct\_clauses, 41

### **MoveToORPType Theory**, 46

Datatypes, 46

Theorems, 46

slCommand\_distinct\_clauses, 46

slOutput\_distinct\_clauses, 46

slState\_distinct\_clauses, 47

### **MoveToPBType Theory**, 51

Datatypes, 51

Theorems, 51

slCommand\_distinct\_clauses, 52

slOutput\_distinct\_clauses, 52

slState\_distinct\_clauses, 52

### **OMNIType Theory**, 3

Datatypes, 3

Theorems, 3

command\_distinct\_clauses, 3

command\_one\_one, 3

escCommand\_distinct\_clauses, 3

escOutput\_distinct\_clauses, 3

escState\_distinct\_clauses, 3

output\_distinct\_clauses, 4

output\_one\_one, 4

principal\_one\_one, 4

state\_distinct\_clauses, 4

state\_one\_one, 4

### **PBIntegratedDef Theory**, 23

Definitions, 23

secAuthorization\_def, 23

secHelper\_def, 23

Theorems, 24

getOmniCommand\_def, 24

getOmniCommand\_ind, 26

secContext\_def, 27

secContext\_ind, 28

### **PBTypeIntegrated Theory**, 21

Datatypes, 21

Theorems, 22

omniCommand\_distinct\_clauses, 22

plCommand\_distinct\_clauses, 22

slCommand\_distinct\_clauses, 22

slCommand\_one\_one, 22

slOutput\_distinct\_clauses, 23

slState\_distinct\_clauses, 23

stateRole\_distinct\_clauses, 23

### **PlanPBDef Theory**, 66

Definitions, 66

PL\_notWARNO\_Auth\_def, 66

PL\_WARNO\_Auth\_def, 66

secContext\_def, 66

secContextNull\_def, 66

Theorems, 67

getInitMove\_def, 67

getInitMove\_ind, 69

getPlCom\_def, 70

getPlCom\_ind, 72

getPsgCom\_def, 73

getPsgCom\_ind, 75

getRecon\_def, 77

getRecon\_ind, 79

- getReport\_def, 82
- getReport\_ind, 85
- getTenativePlan\_def, 87
- getTenativePlan\_ind, 91
- PlanPBType Theory**, 62
  - Datatypes, 62
  - Theorems, 63
    - plCommand\_distinct\_clauses, 63
    - psgCommand\_distinct\_clauses, 64
    - slCommand\_distinct\_clauses, 64
    - slCommand\_one\_one, 64
    - slOutput\_distinct\_clauses, 64
    - slRole\_distinct\_clauses, 65
    - slState\_distinct\_clauses, 65
- satList Theory**, 21
  - Definitions, 21
    - satList\_def, 21
  - Theorems, 21
    - satList\_conj, 21
    - satList\_CONS, 21
    - satList\_nil, 21
- ssm Theory**, 11
  - Datatypes, 11
  - Definitions, 12
    - authenticationTest\_def, 12
    - commandList\_def, 12
    - inputList\_def, 12
    - propCommandList\_def, 12
    - TR\_def, 12
  - Theorems, 13
    - CFGInterpret\_def, 13
    - CFGInterpret\_ind, 13
    - configuration\_one\_one, 13
    - extractCommand\_def, 13
    - extractCommand\_ind, 13
    - extractInput\_def, 14
    - extractInput\_ind, 14
    - extractPropCommand\_def, 15
    - extractPropCommand\_ind, 15
    - TR\_cases, 16
    - TR\_discard\_cmd\_rule, 17
    - TR\_EQ\_rules\_thm, 17
    - TR\_exec\_cmd\_rule, 17
    - TR\_ind, 18
    - TR\_rules, 18
    - TR\_strongind, 19
    - TR\_trap\_cmd\_rule, 20
    - TRrule0, 20
    - TRrule1, 20
    - trType\_distinct\_clauses, 20
    - trType\_one\_one, 21
- ssm11 Theory**, 4
  - Datatypes, 4
  - Definitions, 4
    - TR\_def, 4
  - Theorems, 5
    - CFGInterpret\_def, 5
    - CFGInterpret\_ind, 6
    - configuration\_one\_one, 6
    - order\_distinct\_clauses, 6
    - order\_one\_one, 6
    - TR\_cases, 6
    - TR\_discard\_cmd\_rule, 7
    - TR\_EQ\_rules\_thm, 7
    - TR\_exec\_cmd\_rule, 8
    - TR\_ind, 8
    - TR\_rules, 9
    - TR\_strongind, 9
    - TR\_trap\_cmd\_rule, 10
    - TRrule0, 10
    - TRrule1, 11
    - trType\_distinct\_clauses, 11
    - trType\_one\_one, 11
- ssmConductORP Theory**, 28
  - Definitions, 28
    - secContextConductORP\_def, 28
    - ssmConductORPStateInterp\_def, 28
  - Theorems, 29
    - authTestConductORP\_cmd\_reject\_lemma, 29
    - authTestConductORP\_def, 29
    - authTestConductORP\_ind, 30
    - conductORPNS\_def, 30
    - conductORPNS\_ind, 31
    - conductORPOut\_def, 31
    - conductORPOut\_ind, 31



PlatoonLeader\_exec\_plCommand\_justified.thm, 32  
 PlatoonLeader\_plCommand.lemma, 32  
 PlatoonSergeant\_exec\_psgCommand\_justified.thm, 33  
 PlatoonSergeant\_psgCommand.lemma, 33  
**ssmConductPB Theory**, 35  
   Definitions, 35  
     secContextConductPB\_def, 35  
     ssmConductPBStateInterp\_def, 35  
   Theorems, 35  
     authTestConductPB\_cmd\_reject.lemma, 35  
     authTestConductPB\_def, 35  
     authTestConductPB\_ind, 36  
     conductPBNS\_def, 37  
     conductPBNS\_ind, 37  
     conductPBOut\_def, 38  
     conductPBOut\_ind, 38  
     PlatoonLeader\_exec\_plCommandPB\_justified.thm, 39  
     PlatoonLeader\_plCommandPB.lemma, 39  
     PlatoonSergeant\_exec\_psgCommandPB\_justified.thm, 39  
     PlatoonSergeant\_psgCommandPB.lemma, 40  
**ssmMoveToORP Theory**, 41  
   Definitions, 41  
     secContextMoveToORP\_def, 41  
     ssmMoveToORPStateInterp\_def, 41  
   Theorems, 42  
     authTestMoveToORP\_cmd\_reject.lemma, 42  
     authTestMoveToORP\_def, 42  
     authTestMoveToORP\_ind, 42  
     moveToORPNS\_def, 43  
     moveToORPNS\_ind, 43  
     moveToORPOut\_def, 44  
     moveToORPOut\_ind, 44  
     PlatoonLeader\_exec\_slCommand\_justified.thm, 45  
     PlatoonLeader\_slCommand.lemma, 46  
**ssmMoveToPB Theory**, 47  
   Definitions, 47  
     secContextMoveToPB\_def, 47  
     ssmMoveToPBStateInterp\_def, 47  
   Theorems, 47  
     authTestMoveToPB\_cmd\_reject.lemma, 47  
     authTestMoveToPB\_def, 47  
     authTestMoveToPB\_ind, 48  
     moveToPBNS\_def, 49  
     moveToPBNS\_ind, 49  
     moveToPBOut\_def, 50  
     moveToPBOut\_ind, 50  
     PlatoonLeader\_exec\_slCommand\_justified.thm, 51  
     PlatoonLeader\_slCommand.lemma, 51  
**ssmPlanPB Theory**, 52  
   Theorems, 52  
     inputOK\_def, 52  
     inputOK\_ind, 53  
     planPBNS\_def, 54  
     planPBNS\_ind, 54  
     planPBOut\_def, 55  
     planPBOut\_ind, 55  
     PlatoonLeader\_notWARNO\_notreport1\_exec\_plCommand\_justified.lemma, 55  
     PlatoonLeader\_notWARNO\_notreport1\_exec\_plCommand\_justified.thm, 56  
     PlatoonLeader\_notWARNO\_notreport1\_exec\_plCommand.lemma, 57  
     PlatoonLeader\_psgCommand\_notDiscard.thm, 57  
     PlatoonLeader\_trap\_psgCommand\_justified.lemma, 57  
     PlatoonLeader\_trap\_psgCommand.lemma, 58  
     PlatoonLeader\_WARNO\_exec\_report1\_justified.lemma, 58  
     PlatoonLeader\_WARNO\_exec\_report1\_justified.thm, 60  
     PlatoonLeader\_WARNO\_exec\_report1\_lemma, 61

PlatoonSergeant\_trap\_plCommand\_justified\_lemma, 61  
PlatoonSergeant\_trap\_plCommand\_justified\_thm, 62  
PlatoonSergeant\_trap\_plCommand\_lemma, 62