

Contents

1	OMNIType Theory	3
1.1	Datatypes	3
1.2	Theorems	3
2	ssm11 Theory	4
2.1	Datatypes	4
2.2	Definitions	4
2.3	Theorems	5
3	ssm Theory	11
3.1	Datatypes	11
3.2	Definitions	12
3.3	Theorems	13
4	satList Theory	21
4.1	Definitions	21
4.2	Theorems	21
5	PBTypeIntegrated Theory	21
5.1	Datatypes	21
5.2	Theorems	22
6	PBIntegratedDef Theory	23
6.1	Definitions	23
6.2	Theorems	24
7	ssmPBIntegrated Theory	28
7.1	Theorems	28
8	ssmConductORP Theory	34
8.1	Definitions	34
8.2	Theorems	35
9	ConductORPType Theory	39
9.1	Datatypes	40
9.2	Theorems	40
10	ssmConductPB Theory	41
10.1	Definitions	41
10.2	Theorems	41

11 ConductPBType Theory	46
11.1 Datatypes	46
11.2 Theorems	47
12 ssmMoveToORP Theory	47
12.1 Definitions	48
12.2 Theorems	48
13 MoveToORPType Theory	52
13.1 Datatypes	52
13.2 Theorems	52
14 ssmMoveToPB Theory	53
14.1 Definitions	53
14.2 Theorems	53
15 MoveToPBType Theory	57
15.1 Datatypes	58
15.2 Theorems	58
16 ssmPlanPB Theory	58
16.1 Theorems	59
17 PlanPBType Theory	68
17.1 Datatypes	69
17.2 Theorems	69
18 PlanPBDef Theory	72
18.1 Definitions	72
18.2 Theorems	73

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]

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

```

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]

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

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

[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]

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

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

[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]

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

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

[TR_trap_cmd_rule]

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

[TRrule0]

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

```

(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]

```

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

```

[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]

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

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

3.3 Theorems

[CFGInterpret_def]

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

[CFGInterpret_ind]

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

[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]

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

[extractCommand_ind]

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

$$\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]

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

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]

$$\begin{aligned}
&\vdash \text{PlanPB} \neq \text{MoveToORP} \wedge \text{PlanPB} \neq \text{ConductORP} \wedge \\
&\quad \text{PlanPB} \neq \text{MoveToPB} \wedge \text{PlanPB} \neq \text{ConductPB} \wedge \\
&\quad \text{PlanPB} \neq \text{CompletePB} \wedge \text{PlanPB} \neq \text{unAuthenticated} \wedge \\
&\quad \text{PlanPB} \neq \text{unAuthorized} \wedge \text{MoveToORP} \neq \text{ConductORP} \wedge \\
&\quad \text{MoveToORP} \neq \text{MoveToPB} \wedge \text{MoveToORP} \neq \text{ConductPB} \wedge \\
&\quad \text{MoveToORP} \neq \text{CompletePB} \wedge \text{MoveToORP} \neq \text{unAuthenticated} \wedge \\
&\quad \text{MoveToORP} \neq \text{unAuthorized} \wedge \text{ConductORP} \neq \text{MoveToPB} \wedge \\
&\quad \text{ConductORP} \neq \text{ConductPB} \wedge \text{ConductORP} \neq \text{CompletePB} \wedge \\
&\quad \text{ConductORP} \neq \text{unAuthenticated} \wedge \text{ConductORP} \neq \text{unAuthorized} \wedge \\
&\quad \text{MoveToPB} \neq \text{ConductPB} \wedge \text{MoveToPB} \neq \text{CompletePB} \wedge \\
&\quad \text{MoveToPB} \neq \text{unAuthenticated} \wedge \text{MoveToPB} \neq \text{unAuthorized} \wedge \\
&\quad \text{ConductPB} \neq \text{CompletePB} \wedge \text{ConductPB} \neq \text{unAuthenticated} \wedge \\
&\quad \text{ConductPB} \neq \text{unAuthorized} \wedge \text{CompletePB} \neq \text{unAuthenticated} \wedge \\
&\quad \text{CompletePB} \neq \text{unAuthorized} \wedge \text{unAuthenticated} \neq \text{unAuthorized}
\end{aligned}$$
[slState_distinct_clauses]

$$\begin{aligned}
&\vdash \text{PLAN_PB} \neq \text{MOVE_TO_ORP} \wedge \text{PLAN_PB} \neq \text{CONDUCT_ORP} \wedge \\
&\quad \text{PLAN_PB} \neq \text{MOVE_TO_PB} \wedge \text{PLAN_PB} \neq \text{CONDUCT_PB} \wedge \\
&\quad \text{PLAN_PB} \neq \text{COMPLETE_PB} \wedge \text{MOVE_TO_ORP} \neq \text{CONDUCT_ORP} \wedge \\
&\quad \text{MOVE_TO_ORP} \neq \text{MOVE_TO_PB} \wedge \text{MOVE_TO_ORP} \neq \text{CONDUCT_PB} \wedge \\
&\quad \text{MOVE_TO_ORP} \neq \text{COMPLETE_PB} \wedge \text{CONDUCT_ORP} \neq \text{MOVE_TO_PB} \wedge \\
&\quad \text{CONDUCT_ORP} \neq \text{CONDUCT_PB} \wedge \text{CONDUCT_ORP} \neq \text{COMPLETE_PB} \wedge \\
&\quad \text{MOVE_TO_PB} \neq \text{CONDUCT_PB} \wedge \text{MOVE_TO_PB} \neq \text{COMPLETE_PB} \wedge \\
&\quad \text{CONDUCT_PB} \neq \text{COMPLETE_PB}
\end{aligned}$$
[stateRole_distinct_clauses]

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

6 PBIntegratedDef Theory

Built: 10 June 2018**Parent Theories:** PBTypeIntegrated, aclfoundation

6.1 Definitions

[secAuthorization_def]

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

$$\begin{aligned}
&\vdash (\forall xs. \\
&\quad \text{secContext PLAN_PB } xs = \\
&\quad \text{if } \text{getOmniCommand } xs = \text{ssmPlanPBComplete} \text{ then} \\
&\quad \quad [\text{prop } (\text{SOME } (\text{SLc } (\text{OMNI ssmPlanPBComplete}))) \text{ impf} \\
&\quad \quad \quad \text{Name PlatoonLeader controls} \\
&\quad \quad \text{prop } (\text{SOME } (\text{SLc } (\text{PL crossLD})))])
\end{aligned}$$

```

    else [prop NONE]) ∧
  (∀ xs.
    secContext MOVE_TO_ORP xs =
    if getOmniCommand xs = ssmMoveToORPComplete then
      [prop (SOME (SLc (OMNI ssmMoveToORPComplete))) impf
        Name PlatoonLeader controls
        prop (SOME (SLc (PL conductORP)))])
    else [prop NONE]) ∧
  (∀ xs.
    secContext CONDUCT_ORP xs =
    if getOmniCommand xs = ssmConductORPComplete then
      [prop (SOME (SLc (OMNI ssmConductORPComplete))) impf
        Name PlatoonLeader controls
        prop (SOME (SLc (PL moveToPB)))])
    else [prop NONE]) ∧
  (∀ xs.
    secContext MOVE_TO_PB xs =
    if getOmniCommand xs = ssmConductORPComplete then
      [prop (SOME (SLc (OMNI ssmMoveToPBComplete))) impf
        Name PlatoonLeader controls
        prop (SOME (SLc (PL conductPB)))])
    else [prop NONE]) ∧
  ∀ xs.
    secContext CONDUCT_PB xs =
    if getOmniCommand xs = ssmConductPBComplete then
      [prop (SOME (SLc (OMNI ssmConductPBComplete))) impf
        Name PlatoonLeader controls
        prop (SOME (SLc (PL completePB)))])
    else [prop NONE]

```

[secHelper_def]

```

⊢ ∀ cmd.
  secHelper cmd =
  [Name Omni controls prop (SOME (SLc (OMNI cmd)))]

```

6.2 Theorems

[getOmniCommand_def]

```

⊢ (getOmniCommand [] = invalidOmniCommand) ∧
  (∀ xs cmd.
    getOmniCommand
      (Name Omni says prop (SOME (SLc (OMNI cmd))))::xs =
      cmd) ∧
  (∀ xs. getOmniCommand (TT::xs) = getOmniCommand xs) ∧
  (∀ xs. getOmniCommand (FF::xs) = getOmniCommand xs) ∧
  (∀ xs v2. getOmniCommand (prop v2::xs) = getOmniCommand xs) ∧
  (∀ xs v3. getOmniCommand (notf v3::xs) = getOmniCommand xs) ∧
  (∀ xs v5 v4.

```



```

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

```

```

    getOmniCommand xs) ∧
  (∀ xs v81 v80 v12.
    getOmniCommand (v12 says v80 speaks_for v81::xs) =
    getOmniCommand xs) ∧
  (∀ xs v83 v82 v12.
    getOmniCommand (v12 says v82 controls v83::xs) =
    getOmniCommand xs) ∧
  (∀ xs v86 v85 v84 v12.
    getOmniCommand (v12 says reps v84 v85 v86::xs) =
    getOmniCommand xs) ∧
  (∀ xs v88 v87 v12.
    getOmniCommand (v12 says v87 domi v88::xs) =
    getOmniCommand xs) ∧
  (∀ xs v90 v89 v12.
    getOmniCommand (v12 says v89 eqi v90::xs) =
    getOmniCommand xs) ∧
  (∀ xs v92 v91 v12.
    getOmniCommand (v12 says v91 doms v92::xs) =
    getOmniCommand xs) ∧
  (∀ xs v94 v93 v12.
    getOmniCommand (v12 says v93 eqs v94::xs) =
    getOmniCommand xs) ∧
  (∀ xs v96 v95 v12.
    getOmniCommand (v12 says v95 eqn v96::xs) =
    getOmniCommand xs) ∧
  (∀ xs v98 v97 v12.
    getOmniCommand (v12 says v97 lte v98::xs) =
    getOmniCommand xs) ∧
  (∀ xs v99 v12 v100.
    getOmniCommand (v12 says v99 lt v100::xs) =
    getOmniCommand xs) ∧
  (∀ xs v15 v14.
    getOmniCommand (v14 speaks_for v15::xs) =
    getOmniCommand xs) ∧
  (∀ xs v17 v16.
    getOmniCommand (v16 controls v17::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.

```

$\text{getOmniCommand } (v_{29} \text{ eqn } v_{30}::xs) = \text{getOmniCommand } xs) \wedge$
 $(\forall xs \ v_{32} \ v_{31}.$
 $\text{getOmniCommand } (v_{31} \text{ lte } v_{32}::xs) = \text{getOmniCommand } xs) \wedge$
 $\forall xs \ v_{34} \ v_{33}.$
 $\text{getOmniCommand } (v_{33} \text{ lt } v_{34}::xs) = \text{getOmniCommand } xs$

[getOmniCommand_ind]

$\vdash \forall P.$
 $P \ [] \wedge$
 $(\forall cmd \ xs.$
 $P \ (\text{Name Omni says prop (SOME (SLc (OMNI cmd)))::xs})) \wedge$
 $(\forall xs. P \ xs \Rightarrow P \ (\text{TT::xs})) \wedge (\forall xs. P \ xs \Rightarrow P \ (\text{FF::xs})) \wedge$
 $(\forall v_2 \ xs. P \ xs \Rightarrow P \ (\text{prop } v_2::xs)) \wedge$
 $(\forall v_3 \ xs. P \ xs \Rightarrow P \ (\text{notf } v_3::xs)) \wedge$
 $(\forall v_4 \ v_5 \ xs. P \ xs \Rightarrow P \ (v_4 \text{ andf } v_5::xs)) \wedge$
 $(\forall v_6 \ v_7 \ xs. P \ xs \Rightarrow P \ (v_6 \text{ orf } v_7::xs)) \wedge$
 $(\forall v_8 \ v_9 \ xs. P \ xs \Rightarrow P \ (v_8 \text{ impf } v_9::xs)) \wedge$
 $(\forall v_{10} \ v_{11} \ xs. P \ xs \Rightarrow P \ (v_{10} \text{ eqf } v_{11}::xs)) \wedge$
 $(\forall v_{12} \ xs. P \ xs \Rightarrow P \ (v_{12} \text{ says TT::xs})) \wedge$
 $(\forall v_{12} \ xs. P \ xs \Rightarrow P \ (v_{12} \text{ says FF::xs})) \wedge$
 $(\forall v_{134} \ xs. P \ xs \Rightarrow P \ (\text{Name } v_{134} \text{ says prop NONE::xs})) \wedge$
 $(\forall v_{144} \ xs.$
 $P \ xs \Rightarrow$
 $P \ (\text{Name PlatoonLeader says prop (SOME } v_{144}::xs)) \wedge$
 $(\forall v_{146} \ xs.$
 $P \ xs \Rightarrow P \ (\text{Name Omni says prop (SOME (ESCc } v_{146}::xs))} \wedge$
 $(\forall v_{150} \ xs.$
 $P \ xs \Rightarrow$
 $P \ (\text{Name Omni says prop (SOME (SLc (PL } v_{150})))::xs)) \wedge$
 $(\forall v_{135} \ v_{136} \ v_{68} \ xs.$
 $P \ xs \Rightarrow P \ (v_{135} \text{ meet } v_{136} \text{ says prop } v_{68}::xs)) \wedge$
 $(\forall v_{137} \ v_{138} \ v_{68} \ xs.$
 $P \ xs \Rightarrow P \ (v_{137} \text{ quoting } v_{138} \text{ says prop } v_{68}::xs)) \wedge$
 $(\forall v_{12} \ v_{69} \ xs. P \ xs \Rightarrow P \ (v_{12} \text{ says notf } v_{69}::xs)) \wedge$
 $(\forall v_{12} \ v_{70} \ v_{71} \ xs. P \ xs \Rightarrow P \ (v_{12} \text{ says } (v_{70} \text{ andf } v_{71})::xs)) \wedge$
 $(\forall v_{12} \ v_{72} \ v_{73} \ xs. P \ xs \Rightarrow P \ (v_{12} \text{ says } (v_{72} \text{ orf } v_{73})::xs)) \wedge$
 $(\forall v_{12} \ v_{74} \ v_{75} \ xs. P \ xs \Rightarrow P \ (v_{12} \text{ says } (v_{74} \text{ impf } v_{75})::xs)) \wedge$
 $(\forall v_{12} \ v_{76} \ v_{77} \ xs. P \ xs \Rightarrow P \ (v_{12} \text{ says } (v_{76} \text{ eqf } v_{77})::xs)) \wedge$
 $(\forall v_{12} \ v_{78} \ v_{79} \ xs. P \ xs \Rightarrow P \ (v_{12} \text{ says } v_{78} \text{ says } v_{79}::xs)) \wedge$
 $(\forall v_{12} \ v_{80} \ v_{81} \ xs.$
 $P \ xs \Rightarrow P \ (v_{12} \text{ says } v_{80} \text{ speaks_for } v_{81}::xs)) \wedge$
 $(\forall v_{12} \ v_{82} \ v_{83} \ xs.$
 $P \ xs \Rightarrow P \ (v_{12} \text{ says } v_{82} \text{ controls } v_{83}::xs)) \wedge$
 $(\forall v_{12} \ v_{84} \ v_{85} \ v_{86} \ xs.$
 $P \ xs \Rightarrow P \ (v_{12} \text{ says reps } v_{84} \ v_{85} \ v_{86}::xs)) \wedge$
 $(\forall v_{12} \ v_{87} \ v_{88} \ xs. P \ xs \Rightarrow P \ (v_{12} \text{ says } v_{87} \text{ domi } v_{88}::xs)) \wedge$
 $(\forall v_{12} \ v_{89} \ v_{90} \ xs. P \ xs \Rightarrow P \ (v_{12} \text{ says } v_{89} \text{ eqi } v_{90}::xs)) \wedge$
 $(\forall v_{12} \ v_{91} \ v_{92} \ xs. P \ xs \Rightarrow P \ (v_{12} \text{ says } v_{91} \text{ doms } v_{92}::xs)) \wedge$
 $(\forall v_{12} \ v_{93} \ v_{94} \ xs. P \ xs \Rightarrow P \ (v_{12} \text{ says } v_{93} \text{ eqs } v_{94}::xs)) \wedge$

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

7 ssmPBIntegrated Theory

Built: 10 June 2018

Parent Theories: PBIntegratedDef, ssm

7.1 Theorems

[inputOK_def]

$$\begin{aligned}
& \vdash (\text{inputOK } (\text{Name PlatoonLeader says prop } cmd) \iff T) \wedge \\
& (\text{inputOK } (\text{Name Omni 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
\end{aligned}$$

```

(inputOK (v10 says v97 lt v98)  $\iff$  F)  $\wedge$ 
(inputOK (v12 speaks_for v13)  $\iff$  F)  $\wedge$ 
(inputOK (v14 controls v15)  $\iff$  F)  $\wedge$ 
(inputOK (reps v16 v17 v18)  $\iff$  F)  $\wedge$ 
(inputOK (v19 domi v20)  $\iff$  F)  $\wedge$ 
(inputOK (v21 eqi v22)  $\iff$  F)  $\wedge$ 
(inputOK (v23 doms v24)  $\iff$  F)  $\wedge$ 
(inputOK (v25 eqs v26)  $\iff$  F)  $\wedge$  (inputOK (v27 eqn v28)  $\iff$  F)  $\wedge$ 
(inputOK (v29 lte v30)  $\iff$  F)  $\wedge$  (inputOK (v31 lt v32)  $\iff$  F)

```

[inputOK_ind]

$\vdash \forall P.$

```

( $\forall$  cmd. P (Name PlatoonLeader says prop cmd))  $\wedge$ 
( $\forall$  cmd. P (Name Omni says prop cmd))  $\wedge$  P TT  $\wedge$  P FF  $\wedge$ 
( $\forall$  v. P (prop v))  $\wedge$  ( $\forall$  v1. P (notf v1))  $\wedge$ 
( $\forall$  v2 v3. P (v2 andf v3))  $\wedge$  ( $\forall$  v4 v5. P (v4 orf v5))  $\wedge$ 
( $\forall$  v6 v7. P (v6 impf v7))  $\wedge$  ( $\forall$  v8 v9. P (v8 eqf v9))  $\wedge$ 
( $\forall$  v10. P (v10 says TT))  $\wedge$  ( $\forall$  v10. P (v10 says FF))  $\wedge$ 
( $\forall$  v133 v134 v66. P (v133 meet v134 says prop v66))  $\wedge$ 
( $\forall$  v135 v136 v66. P (v135 quoting v136 says prop v66))  $\wedge$ 
( $\forall$  v10 v67. P (v10 says notf v67))  $\wedge$ 
( $\forall$  v10 v68 v69. P (v10 says (v68 andf v69)))  $\wedge$ 
( $\forall$  v10 v70 v71. P (v10 says (v70 orf v71)))  $\wedge$ 
( $\forall$  v10 v72 v73. P (v10 says (v72 impf v73)))  $\wedge$ 
( $\forall$  v10 v74 v75. P (v10 says (v74 eqf v75)))  $\wedge$ 
( $\forall$  v10 v76 v77. P (v10 says v76 says v77))  $\wedge$ 
( $\forall$  v10 v78 v79. P (v10 says v78 speaks_for v79))  $\wedge$ 
( $\forall$  v10 v80 v81. P (v10 says v80 controls v81))  $\wedge$ 
( $\forall$  v10 v82 v83 v84. P (v10 says reps v82 v83 v84))  $\wedge$ 
( $\forall$  v10 v85 v86. P (v10 says v85 domi v86))  $\wedge$ 
( $\forall$  v10 v87 v88. P (v10 says v87 eqi v88))  $\wedge$ 
( $\forall$  v10 v89 v90. P (v10 says v89 doms v90))  $\wedge$ 
( $\forall$  v10 v91 v92. P (v10 says v91 eqs v92))  $\wedge$ 
( $\forall$  v10 v93 v94. P (v10 says v93 eqn v94))  $\wedge$ 
( $\forall$  v10 v95 v96. P (v10 says v95 lte v96))  $\wedge$ 
( $\forall$  v10 v97 v98. P (v10 says v97 lt v98))  $\wedge$ 
( $\forall$  v12 v13. P (v12 speaks_for v13))  $\wedge$ 
( $\forall$  v14 v15. P (v14 controls v15))  $\wedge$ 
( $\forall$  v16 v17 v18. P (reps v16 v17 v18))  $\wedge$ 
( $\forall$  v19 v20. P (v19 domi v20))  $\wedge$ 
( $\forall$  v21 v22. P (v21 eqi v22))  $\wedge$ 
( $\forall$  v23 v24. P (v23 doms v24))  $\wedge$ 
( $\forall$  v25 v26. P (v25 eqs v26))  $\wedge$  ( $\forall$  v27 v28. P (v27 eqn v28))  $\wedge$ 
( $\forall$  v29 v30. P (v29 lte v30))  $\wedge$  ( $\forall$  v31 v32. P (v31 lt v32))  $\Rightarrow$ 
 $\forall v. P v$ 

```

[PBNS_def]

\vdash (PBNS PLAN_PB (exec [SOME (SLc (PL crossLD))]) =
MOVE_TO_ORP) \wedge

(PBNS MOVE_TO_ORP (exec [SOME (SLc (PL conductORP))]) =
 CONDUCT_ORP) \wedge
 (PBNS CONDUCT_ORP (exec [SOME (SLc (PL moveToPB))]) =
 MOVE_TO_PB) \wedge
 (PBNS MOVE_TO_PB (exec [SOME (SLc (PL conductPB))]) =
 CONDUCT_PB) \wedge
 (PBNS CONDUCT_PB (exec [SOME (SLc (PL completePB))]) =
 COMPLETE_PB) \wedge (PBNS s (trap v_0) = s) \wedge
 (PBNS s (discard v_1) = s)

[PBNS_ind]

$\vdash \forall P.$

P PLAN_PB (exec [SOME (SLc (PL crossLD))]) \wedge
 P MOVE_TO_ORP (exec [SOME (SLc (PL conductORP))]) \wedge
 P CONDUCT_ORP (exec [SOME (SLc (PL moveToPB))]) \wedge
 P MOVE_TO_PB (exec [SOME (SLc (PL conductPB))]) \wedge
 P CONDUCT_PB (exec [SOME (SLc (PL completePB))]) \wedge
 $(\forall s \ v_0. P \ s \ (\text{trap } v_0)) \wedge (\forall s \ v_1. P \ s \ (\text{discard } v_1)) \wedge$
 $(\forall v_8. P \ v_8 \ (\text{exec } [])) \wedge$
 $(\forall v_{11} \ v_{10}. P \ v_{11} \ (\text{exec } (\text{NONE}::v_{10}))) \wedge$
 $(\forall v_{16} \ v_{13} \ v_{15}. P \ v_{16} \ (\text{exec } (\text{SOME } (\text{ESCc } v_{13})::v_{15}))) \wedge$
 P MOVE_TO_ORP (exec [SOME (SLc (PL crossLD))]) \wedge
 P CONDUCT_ORP (exec [SOME (SLc (PL crossLD))]) \wedge
 P MOVE_TO_PB (exec [SOME (SLc (PL crossLD))]) \wedge
 P CONDUCT_PB (exec [SOME (SLc (PL crossLD))]) \wedge
 P COMPLETE_PB (exec [SOME (SLc (PL crossLD))]) \wedge
 P PLAN_PB (exec [SOME (SLc (PL conductORP))]) \wedge
 P CONDUCT_ORP (exec [SOME (SLc (PL conductORP))]) \wedge
 P MOVE_TO_PB (exec [SOME (SLc (PL conductORP))]) \wedge
 P CONDUCT_PB (exec [SOME (SLc (PL conductORP))]) \wedge
 P COMPLETE_PB (exec [SOME (SLc (PL conductORP))]) \wedge
 P PLAN_PB (exec [SOME (SLc (PL moveToPB))]) \wedge
 P MOVE_TO_ORP (exec [SOME (SLc (PL moveToPB))]) \wedge
 P MOVE_TO_PB (exec [SOME (SLc (PL moveToPB))]) \wedge
 P CONDUCT_PB (exec [SOME (SLc (PL moveToPB))]) \wedge
 P COMPLETE_PB (exec [SOME (SLc (PL moveToPB))]) \wedge
 P PLAN_PB (exec [SOME (SLc (PL conductPB))]) \wedge
 P MOVE_TO_ORP (exec [SOME (SLc (PL conductPB))]) \wedge
 P CONDUCT_ORP (exec [SOME (SLc (PL conductPB))]) \wedge
 P CONDUCT_PB (exec [SOME (SLc (PL conductPB))]) \wedge
 P COMPLETE_PB (exec [SOME (SLc (PL conductPB))]) \wedge
 P PLAN_PB (exec [SOME (SLc (PL completePB))]) \wedge
 P MOVE_TO_ORP (exec [SOME (SLc (PL completePB))]) \wedge
 P CONDUCT_ORP (exec [SOME (SLc (PL completePB))]) \wedge
 P MOVE_TO_PB (exec [SOME (SLc (PL completePB))]) \wedge
 P COMPLETE_PB (exec [SOME (SLc (PL completePB))]) \wedge
 $(\forall v_{24}. P \ v_{24} \ (\text{exec } [\text{SOME } (\text{SLc } (\text{PL incomplete}))])) \wedge$
 $(\forall v_{26} \ v_{25} \ v_{22} \ v_{23}.$
 $P \ v_{26} \ (\text{exec } (\text{SOME } (\text{SLc } (\text{PL } v_{25}))::v_{22}::v_{23}))) \wedge$

$$(\forall v_{28} v_{19} v_{27}. P v_{28} (\text{exec } (\text{SOME } (\text{SLc } (\text{OMNI } v_{19}))) :: v_{27})) \Rightarrow \\ \forall v v_1. P v v_1$$

[PBOut_def]

$$\begin{aligned} &\vdash (\text{PBOut PLAN_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL crossLD}))])) = \\ &\quad \text{MoveToORP}) \wedge \\ &(\text{PBOut MOVE_TO_ORP } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL conductORP}))])) = \\ &\quad \text{ConductORP}) \wedge \\ &(\text{PBOut CONDUCT_ORP } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL moveToPB}))])) = \\ &\quad \text{MoveToPB}) \wedge \\ &(\text{PBOut MOVE_TO_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL conductPB}))])) = \\ &\quad \text{ConductPB}) \wedge \\ &(\text{PBOut CONDUCT_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL completePB}))])) = \\ &\quad \text{CompletePB}) \wedge (\text{PBOut } s (\text{trap } v_0) = \text{unAuthorized}) \wedge \\ &(\text{PBOut } s (\text{discard } v_1) = \text{unAuthenticated}) \end{aligned}$$

[PBOut_ind]

$$\begin{aligned} &\vdash \forall P. \\ &\quad P \text{ PLAN_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL crossLD}))])) \wedge \\ &\quad P \text{ MOVE_TO_ORP } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL conductORP}))])) \wedge \\ &\quad P \text{ CONDUCT_ORP } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL moveToPB}))])) \wedge \\ &\quad P \text{ MOVE_TO_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL conductPB}))])) \wedge \\ &\quad P \text{ CONDUCT_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL completePB}))])) \wedge \\ &\quad (\forall s v_0. P s (\text{trap } v_0)) \wedge (\forall s v_1. P s (\text{discard } v_1)) \wedge \\ &\quad (\forall v_8. P v_8 (\text{exec } [])) \wedge \\ &\quad (\forall v_{11} v_{10}. P v_{11} (\text{exec } (\text{NONE} :: v_{10}))) \wedge \\ &\quad (\forall v_{16} v_{13} v_{15}. P v_{16} (\text{exec } (\text{SOME } (\text{ESCc } v_{13}) :: v_{15}))) \wedge \\ &\quad P \text{ MOVE_TO_ORP } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL crossLD}))])) \wedge \\ &\quad P \text{ CONDUCT_ORP } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL crossLD}))])) \wedge \\ &\quad P \text{ MOVE_TO_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL crossLD}))])) \wedge \\ &\quad P \text{ CONDUCT_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL crossLD}))])) \wedge \\ &\quad P \text{ COMPLETE_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL crossLD}))])) \wedge \\ &\quad P \text{ PLAN_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL conductORP}))])) \wedge \\ &\quad P \text{ CONDUCT_ORP } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL conductORP}))])) \wedge \\ &\quad P \text{ MOVE_TO_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL conductORP}))])) \wedge \\ &\quad P \text{ CONDUCT_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL conductORP}))])) \wedge \\ &\quad P \text{ COMPLETE_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL conductORP}))])) \wedge \\ &\quad P \text{ PLAN_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL moveToPB}))])) \wedge \\ &\quad P \text{ MOVE_TO_ORP } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL moveToPB}))])) \wedge \\ &\quad P \text{ MOVE_TO_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL moveToPB}))])) \wedge \\ &\quad P \text{ CONDUCT_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL moveToPB}))])) \wedge \\ &\quad P \text{ COMPLETE_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL moveToPB}))])) \wedge \\ &\quad P \text{ PLAN_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL conductPB}))])) \wedge \\ &\quad P \text{ MOVE_TO_ORP } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL conductPB}))])) \wedge \\ &\quad P \text{ CONDUCT_ORP } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL conductPB}))])) \wedge \\ &\quad P \text{ CONDUCT_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL conductPB}))])) \wedge \\ &\quad P \text{ COMPLETE_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL conductPB}))])) \wedge \\ &\quad P \text{ PLAN_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL completePB}))])) \wedge \\ &\quad P \text{ MOVE_TO_ORP } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL completePB}))])) \wedge \end{aligned}$$

$$\begin{aligned}
& P \text{ CONDUCT_ORP } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL } \text{completePB}))]) \wedge \\
& P \text{ MOVE_TO_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL } \text{completePB}))]) \wedge \\
& P \text{ COMPLETE_PB } (\text{exec } [\text{SOME } (\text{SLc } (\text{PL } \text{completePB}))]) \wedge \\
& (\forall v_{24}. P \ v_{24} \ (\text{exec } [\text{SOME } (\text{SLc } (\text{PL } \text{incomplete}))])) \wedge \\
& (\forall v_{26} \ v_{25} \ v_{22} \ v_{23}. \\
& \quad P \ v_{26} \ (\text{exec } (\text{SOME } (\text{SLc } (\text{PL } v_{25})))::v_{22}::v_{23})) \wedge \\
& (\forall v_{28} \ v_{19} \ v_{27}. P \ v_{28} \ (\text{exec } (\text{SOME } (\text{SLc } (\text{OMNI } v_{19})))::v_{27})) \Rightarrow \\
& \forall v \ v_1. P \ v \ v_1
\end{aligned}$$

[PlatoonLeader_Omni_notDiscard_slCommand_thm]

$$\begin{aligned}
& \vdash \forall NS \ Out \ M \ Oi \ Os. \\
& \quad \neg \text{TR } (M, Oi, Os) \\
& \quad (\text{discard} \\
& \quad \quad [\text{SOME } (\text{SLc } (\text{PL } plCommand)); \\
& \quad \quad \quad \text{SOME } (\text{SLc } (\text{OMNI } omniCommand))]) \\
& \quad (\text{CFG inputOK secContext secAuthorization} \\
& \quad \quad ([\text{Name Omni says prop } (\text{SOME } (\text{SLc } (\text{PL } plCommand)))); \\
& \quad \quad \quad \text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop } (\text{SOME } (\text{SLc } (\text{OMNI } omniCommand)))]::ins) \text{ PLAN_PB} \\
& \quad \quad outs) \\
& \quad (\text{CFG inputOK secContext secAuthorization ins} \\
& \quad \quad (NS \text{ PLAN_PB} \\
& \quad \quad \quad (\text{discard} \\
& \quad \quad \quad \quad [\text{SOME } (\text{SLc } (\text{PL } plCommand)); \\
& \quad \quad \quad \quad \quad \text{SOME } (\text{SLc } (\text{OMNI } omniCommand))])) \\
& \quad \quad (Out \text{ PLAN_PB} \\
& \quad \quad \quad (\text{discard} \\
& \quad \quad \quad \quad [\text{SOME } (\text{SLc } (\text{PL } plCommand)); \\
& \quad \quad \quad \quad \quad \text{SOME } (\text{SLc } (\text{OMNI } omniCommand))]]::outs))
\end{aligned}$$

[PlatoonLeader_PLAN_PB_exec_lemma]

$$\begin{aligned}
& \vdash \forall M \ Oi \ Os. \\
& \quad \text{CFGInterpret } (M, Oi, Os) \\
& \quad (\text{CFG inputOK secContext secAuthorization} \\
& \quad \quad ([\text{Name Omni says} \\
& \quad \quad \quad \text{prop } (\text{SOME } (\text{SLc } (\text{OMNI } ssmPlanPBComplete}))]; \\
& \quad \quad \quad \text{Name PlatoonLeader says} \\
& \quad \quad \quad \text{prop } (\text{SOME } (\text{SLc } (\text{PL } crossLD)))]::ins) \text{ PLAN_PB} \\
& \quad \quad outs) \Rightarrow \\
& \quad (M, Oi, Os) \text{ satList} \\
& \quad \text{propCommandList} \\
& \quad \quad [\text{Name Omni says} \\
& \quad \quad \quad \text{prop } (\text{SOME } (\text{SLc } (\text{OMNI } ssmPlanPBComplete}))]; \\
& \quad \quad \quad \text{Name PlatoonLeader says prop } (\text{SOME } (\text{SLc } (\text{PL } crossLD)))]
\end{aligned}$$

[PlatoonLeader_PLAN_PB_trap_justified_lemma]

$$\begin{aligned}
& \vdash omniCommand \neq ssmPlanPBComplete \Rightarrow \\
& \quad (s = \text{PLAN_PB}) \Rightarrow
\end{aligned}$$


```

 $\forall NS \text{ Out } M \text{ } Oi \text{ } Os.$ 
TR (M, Oi, Os)
  (trap
    (inputList
      [Name Omni says
        prop (SOME (SLc (OMNI omniCommand)));
        Name PlatoonLeader says
        prop (SOME (SLc (PL crossLD)))]))
    (CFG inputOK secContext secAuthorization
      ([Name Omni says prop (SOME (SLc (OMNI omniCommand)));
        Name PlatoonLeader says
        prop (SOME (SLc (PL crossLD)))]::ins) PLAN_PB outs)
    (CFG inputOK secContext secAuthorization ins
      (NS PLAN_PB
        (trap
          (inputList
            [Name Omni says
              prop (SOME (SLc (OMNI omniCommand)));
              Name PlatoonLeader says
              prop (SOME (SLc (PL crossLD)))])))
        (Out PLAN_PB
          (trap
            (inputList
              [Name Omni says
                prop (SOME (SLc (OMNI omniCommand)));
                Name PlatoonLeader says
                prop (SOME (SLc (PL crossLD)))]))::outs))  $\iff$ 
          authenticationTest inputOK
            [Name Omni says prop (SOME (SLc (OMNI omniCommand)));
              Name PlatoonLeader says
              prop (SOME (SLc (PL crossLD)))]  $\wedge$ 
            CFGInterpret (M, Oi, Os)
              (CFG inputOK secContext secAuthorization
                ([Name Omni says prop (SOME (SLc (OMNI omniCommand)));
                  Name PlatoonLeader says
                  prop (SOME (SLc (PL crossLD)))]::ins) PLAN_PB
                outs)  $\wedge$  (M, Oi, Os) sat prop NONE

```

[PlatoonLeader_PLAN_PB_trap_justified_thm]

```

 $\vdash \text{ omniCommand } \neq \text{ ssmPlanPBComplete } \Rightarrow$ 
  (s = PLAN_PB)  $\Rightarrow$ 
 $\forall NS \text{ Out } M \text{ } Oi \text{ } Os.$ 
  TR (M, Oi, Os)
    (trap
      [SOME (SLc (OMNI omniCommand));
        SOME (SLc (PL crossLD))])
    (CFG inputOK secContext secAuthorization
      ([Name Omni says prop (SOME (SLc (OMNI omniCommand)));
        Name PlatoonLeader says

```

```

      prop (SOME (SLc (PL crossLD))))]::ins) PLAN_PB outs)
(CFG inputOK secContext secAuthorization ins
  (NS PLAN_PB
    (trap
      [SOME (SLc (OMNI omniCommand));
       SOME (SLc (PL crossLD))]))
  (Out PLAN_PB
    (trap
      [SOME (SLc (OMNI omniCommand));
       SOME (SLc (PL crossLD))]::outs))  $\iff$ 
authenticationTest inputOK
  [Name Omni says prop (SOME (SLc (OMNI omniCommand))));
   Name PlatoonLeader says
   prop (SOME (SLc (PL crossLD)))]  $\wedge$ 
CFGInterpret (M, Oi, Os)
  (CFG inputOK secContext secAuthorization
    ([Name Omni says prop (SOME (SLc (OMNI omniCommand))));
     Name PlatoonLeader says
     prop (SOME (SLc (PL crossLD)))]::ins) PLAN_PB
    outs)  $\wedge$  (M, Oi, Os) sat prop NONE

```

[PlatoonLeader_PLAN_PB_trap_lemma]

```

 $\vdash$  omniCommand  $\neq$  ssmPlanPBComplete  $\Rightarrow$ 
  (s = PLAN_PB)  $\Rightarrow$ 
 $\forall M \ Oi \ Os.$ 
  CFGInterpret (M, Oi, Os)
    (CFG inputOK secContext secAuthorization
      ([Name Omni says prop (SOME (SLc (OMNI omniCommand))));
       Name PlatoonLeader says
       prop (SOME (SLc (PL crossLD)))]::ins) PLAN_PB
      outs)  $\Rightarrow$ 
    (M, Oi, Os) sat prop NONE

```

8 ssmConductORP Theory

Built: 10 June 2018

Parent Theories: ConductORPType, ssm11, OMNIType

8.1 Definitions

[secContextConductORP_def]

```

 $\vdash \forall plcnd \ psgcmd \ incomplete.$ 
secContextConductORP plcnd psgcmd incomplete =
  [Name PlatoonLeader controls prop (SOME (SLc (PL plcnd))));
   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 plcnd))) impf prop NONE]

```

[ssmConductORPStateInterp_def]

$\vdash \forall slState. \text{ssmConductORPStateInterp } slState = \text{TT}$

8.2 Theorems

[authTestConductORP_cmd_reject_lemma]

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

[authTestConductORP_def]

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

$$\begin{aligned}
& (\text{authTestConductORP } (v_{21} \text{ eqi } v_{22}) \iff F) \wedge \\
& (\text{authTestConductORP } (v_{23} \text{ doms } v_{24}) \iff F) \wedge \\
& (\text{authTestConductORP } (v_{25} \text{ eqs } v_{26}) \iff F) \wedge \\
& (\text{authTestConductORP } (v_{27} \text{ eqn } v_{28}) \iff F) \wedge \\
& (\text{authTestConductORP } (v_{29} \text{ lte } v_{30}) \iff F) \wedge \\
& (\text{authTestConductORP } (v_{31} \text{ lt } v_{32}) \iff F)
\end{aligned}$$

[authTestConductORP_ind]

$\vdash \forall P.$

$$\begin{aligned}
& (\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 \\
& (\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}$$

[conductORPNS_def]

$$\begin{aligned}
& \vdash (\text{conductORPNS CONDUCT_ORP (exec (PL secure))} = \text{SECURE}) \wedge \\
& (\text{conductORPNS CONDUCT_ORP (exec (PL plIncomplete))} = \\
& \quad \text{CONDUCT_ORP}) \wedge \\
& (\text{conductORPNS SECURE (exec (PSG actionsIn))} = \text{ACTIONS_IN}) \wedge \\
& (\text{conductORPNS SECURE (exec (PSG psgIncomplete))} = \text{SECURE}) \wedge
\end{aligned}$$

$$\begin{aligned}
& (\text{conductORPNS ACTIONS_IN (exec (PL withdraw))} = \text{WITHDRAW}) \wedge \\
& (\text{conductORPNS ACTIONS_IN (exec (PL plIncomplete))} = \\
& \quad \text{ACTIONS_IN}) \wedge \\
& (\text{conductORPNS WITHDRAW (exec (PL complete))} = \text{COMPLETE}) \wedge \\
& (\text{conductORPNS WITHDRAW (exec (PL plIncomplete))} = \text{WITHDRAW}) \wedge \\
& (\text{conductORPNS } s \text{ (trap (PL cmd'))} = s) \wedge \\
& (\text{conductORPNS } s \text{ (trap (PSG cmd))} = s) \wedge \\
& (\text{conductORPNS } s \text{ (discard (PL cmd'))} = s) \wedge \\
& (\text{conductORPNS } s \text{ (discard (PSG cmd))} = s)
\end{aligned}$$

[conductORPNS_ind]

$$\begin{aligned}
& \vdash \forall P. \\
& \quad P \text{ CONDUCT_ORP (exec (PL secure))} \wedge \\
& \quad P \text{ CONDUCT_ORP (exec (PL plIncomplete))} \wedge \\
& \quad P \text{ SECURE (exec (PSG actionsIn))} \wedge \\
& \quad P \text{ SECURE (exec (PSG psgIncomplete))} \wedge \\
& \quad P \text{ ACTIONS_IN (exec (PL withdraw))} \wedge \\
& \quad P \text{ ACTIONS_IN (exec (PL plIncomplete))} \wedge \\
& \quad P \text{ WITHDRAW (exec (PL complete))} \wedge \\
& \quad P \text{ WITHDRAW (exec (PL plIncomplete))} \wedge \\
& \quad (\forall s \text{ cmd. } P \text{ } s \text{ (trap (PL cmd'))}) \wedge \\
& \quad (\forall s \text{ cmd. } P \text{ } s \text{ (trap (PSG cmd))}) \wedge \\
& \quad (\forall s \text{ cmd. } P \text{ } s \text{ (discard (PL cmd'))}) \wedge \\
& \quad (\forall s \text{ cmd. } P \text{ } s \text{ (discard (PSG cmd))}) \wedge \\
& \quad P \text{ CONDUCT_ORP (exec (PL withdraw))} \wedge \\
& \quad P \text{ CONDUCT_ORP (exec (PL complete))} \wedge \\
& \quad (\forall v_{11}. P \text{ CONDUCT_ORP (exec (PSG } v_{11}))}) \wedge \\
& \quad (\forall v_{13}. P \text{ SECURE (exec (PL } v_{13}))}) \wedge \\
& \quad P \text{ ACTIONS_IN (exec (PL secure))} \wedge \\
& \quad P \text{ ACTIONS_IN (exec (PL complete))} \wedge \\
& \quad (\forall v_{17}. P \text{ ACTIONS_IN (exec (PSG } v_{17}))}) \wedge \\
& \quad P \text{ WITHDRAW (exec (PL secure))} \wedge \\
& \quad P \text{ WITHDRAW (exec (PL withdraw))} \wedge \\
& \quad (\forall v_{20}. P \text{ WITHDRAW (exec (PSG } v_{20}))}) \wedge \\
& \quad (\forall v_{21}. P \text{ COMPLETE (exec } v_{21})) \Rightarrow \\
& \quad \forall v \text{ } v_1. P \text{ } v \text{ } v_1
\end{aligned}$$

[conductORPOut_def]

$$\begin{aligned}
& \vdash (\text{conductORPOut CONDUCT_ORP (exec (PL secure))} = \text{Secure}) \wedge \\
& (\text{conductORPOut CONDUCT_ORP (exec (PL plIncomplete))} = \\
& \quad \text{ConductORP}) \wedge \\
& (\text{conductORPOut SECURE (exec (PSG actionsIn))} = \text{ActionsIn}) \wedge \\
& (\text{conductORPOut SECURE (exec (PSG psgIncomplete))} = \text{Secure}) \wedge \\
& (\text{conductORPOut ACTIONS_IN (exec (PL withdraw))} = \text{Withdraw}) \wedge \\
& (\text{conductORPOut ACTIONS_IN (exec (PL plIncomplete))} = \\
& \quad \text{ActionsIn}) \wedge \\
& (\text{conductORPOut WITHDRAW (exec (PL complete))} = \text{Complete}) \wedge \\
& (\text{conductORPOut WITHDRAW (exec (PL plIncomplete))} = \\
& \quad \text{Withdraw}) \wedge
\end{aligned}$$

$(\text{conductORPOut } s \text{ (trap (PL } cmd')) = \text{unAuthorized}) \wedge$
 $(\text{conductORPOut } s \text{ (trap (PSG } cmd)) = \text{unAuthorized}) \wedge$
 $(\text{conductORPOut } s \text{ (discard (PL } cmd')) = \text{unAuthenticated}) \wedge$
 $(\text{conductORPOut } s \text{ (discard (PSG } cmd)) = \text{unAuthenticated})$

[conductORPOut_ind]

$\vdash \forall P.$
 $P \text{ CONDUCT_ORP (exec (PL secure))} \wedge$
 $P \text{ CONDUCT_ORP (exec (PL plIncomplete))} \wedge$
 $P \text{ SECURE (exec (PSG actionsIn))} \wedge$
 $P \text{ SECURE (exec (PSG psgIncomplete))} \wedge$
 $P \text{ ACTIONS_IN (exec (PL withdraw))} \wedge$
 $P \text{ ACTIONS_IN (exec (PL plIncomplete))} \wedge$
 $P \text{ WITHDRAW (exec (PL complete))} \wedge$
 $P \text{ WITHDRAW (exec (PL plIncomplete))} \wedge$
 $(\forall s \text{ cmd. } P \text{ s (trap (PL } cmd))) \wedge$
 $(\forall s \text{ cmd. } P \text{ s (trap (PSG } cmd))) \wedge$
 $(\forall s \text{ cmd. } P \text{ s (discard (PL } cmd))) \wedge$
 $(\forall s \text{ cmd. } P \text{ s (discard (PSG } cmd))) \wedge$
 $P \text{ CONDUCT_ORP (exec (PL withdraw))} \wedge$
 $P \text{ CONDUCT_ORP (exec (PL complete))} \wedge$
 $(\forall v_{11}. P \text{ CONDUCT_ORP (exec (PSG } v_{11}))) \wedge$
 $(\forall v_{13}. P \text{ SECURE (exec (PL } v_{13}))) \wedge$
 $P \text{ ACTIONS_IN (exec (PL secure))} \wedge$
 $P \text{ ACTIONS_IN (exec (PL complete))} \wedge$
 $(\forall v_{17}. P \text{ ACTIONS_IN (exec (PSG } v_{17}))) \wedge$
 $P \text{ WITHDRAW (exec (PL secure))} \wedge$
 $P \text{ WITHDRAW (exec (PL withdraw))} \wedge$
 $(\forall v_{20}. P \text{ WITHDRAW (exec (PSG } v_{20}))) \wedge$
 $(\forall v_{21}. P \text{ COMPLETE (exec } v_{21})) \Rightarrow$
 $\forall v \text{ v}_1. P \text{ v } v_1$

[PlatoonLeader_exec_plCommand_justified_thm]

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

(Name PlatoonLeader says
 prop (SOME (SLc (PL *plCommand*)))::ins) s outs) \wedge
 (*M, Oi, Os*) sat prop (SOME (SLc (PL *plCommand*)))

[PlatoonLeader_plCommand_lemma]

\vdash CFGInterpret (*M, Oi, Os*)
 (CFG authTestConductORP ssmConductORPStateInterp
 (secContextConductORP *plCommand psgCommand incomplete*)
 (Name PlatoonLeader says
 prop (SOME (SLc (PL *plCommand*)))::ins) s outs) \Rightarrow
 (*M, Oi, Os*) sat prop (SOME (SLc (PL *plCommand*)))

[PlatoonSergeant_exec_psgCommand_justified_thm]

$\vdash \forall NS$ 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)) \iff
 authTestConductORP
 (Name PlatoonSergeant says
 prop (SOME (SLc (PSG *psgCommand*)))) \wedge
 CFGInterpret (*M, Oi, Os*)
 (CFG authTestConductORP ssmConductORPStateInterp
 (secContextConductORP *plCommand psgCommand incomplete*)
 (Name PlatoonSergeant says
 prop (SOME (SLc (PSG *psgCommand*)))::ins) s outs) \wedge
 (*M, Oi, Os*) sat prop (SOME (SLc (PSG *psgCommand*)))

[PlatoonSergeant_psgCommand_lemma]

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

9 ConductORPType Theory

Built: 10 June 2018

Parent Theories: indexedLists, patternMatches

9.1 Datatypes

$plCommand = \text{secure} \mid \text{withdraw} \mid \text{complete} \mid plIncomplete$

$psgCommand = \text{actionsIn} \mid psgIncomplete$

$slCommand =$
 $PL \text{ ConductORPType\$plCommand}$
 $\mid PSG \text{ ConductORPType\$psgCommand}$

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

$slState = \text{CONDUCT_ORP} \mid \text{SECURE} \mid \text{ACTIONS_IN} \mid \text{WITHDRAW}$
 $\mid \text{COMPLETE}$

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

9.2 Theorems

[$plCommand_distinct_clauses$]

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

[$psgCommand_distinct_clauses$]

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

[$slCommand_distinct_clauses$]

$\vdash \forall a' a. PL\ a \neq PSG\ a'$

[$slCommand_one_one$]

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

[$slOutput_distinct_clauses$]

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

[slRole_distinct_clauses]

⊢ PlatoonLeader ≠ PlatoonSergeant

[slState_distinct_clauses]

⊢ CONDUCT_ORP ≠ SECURE ∧ CONDUCT_ORP ≠ ACTIONS_IN ∧
 CONDUCT_ORP ≠ WITHDRAW ∧ CONDUCT_ORP ≠ COMPLETE ∧
 SECURE ≠ ACTIONS_IN ∧ SECURE ≠ WITHDRAW ∧ SECURE ≠ COMPLETE ∧
 ACTIONS_IN ≠ WITHDRAW ∧ ACTIONS_IN ≠ COMPLETE ∧
 WITHDRAW ≠ COMPLETE

10 ssmConductPB Theory

Built: 10 June 2018

Parent Theories: ConductPBType, ssm11, OMNIType

10.1 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

10.2 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) ∧

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

$[\text{authTestConductPB_ind}]$

$\vdash \forall P.$

$(\forall \text{cmd}. P (\text{Name PlatoonLeader says prop cmd})) \wedge$
 $(\forall \text{cmd}. P (\text{Name PlatoonSergeant says prop cmd})) \wedge P \text{ TT} \wedge$
 $P \text{ FF} \wedge (\forall v. P (\text{prop } v)) \wedge (\forall v_1. P (\text{notf } v_1)) \wedge$
 $(\forall v_2 \text{ } v_3. P (v_2 \text{ andf } v_3)) \wedge (\forall v_4 \text{ } v_5. P (v_4 \text{ orf } v_5)) \wedge$
 $(\forall v_6 \text{ } v_7. P (v_6 \text{ impf } v_7)) \wedge (\forall v_8 \text{ } v_9. P (v_8 \text{ eqf } v_9)) \wedge$
 $(\forall v_{10}. P (v_{10} \text{ says TT})) \wedge (\forall v_{10}. P (v_{10} \text{ says FF})) \wedge$
 $(\forall v_{133} \text{ } v_{134} \text{ } v_{66}. P (v_{133} \text{ meet } v_{134} \text{ says prop } v_{66})) \wedge$
 $(\forall v_{135} \text{ } v_{136} \text{ } v_{66}. P (v_{135} \text{ quoting } v_{136} \text{ says prop } v_{66})) \wedge$
 $(\forall v_{10} \text{ } v_{67}. P (v_{10} \text{ says notf } v_{67})) \wedge$
 $(\forall v_{10} \text{ } v_{68} \text{ } v_{69}. P (v_{10} \text{ says } (v_{68} \text{ andf } v_{69}))) \wedge$
 $(\forall v_{10} \text{ } v_{70} \text{ } v_{71}. P (v_{10} \text{ says } (v_{70} \text{ orf } v_{71}))) \wedge$
 $(\forall v_{10} \text{ } v_{72} \text{ } v_{73}. P (v_{10} \text{ says } (v_{72} \text{ impf } v_{73}))) \wedge$
 $(\forall v_{10} \text{ } v_{74} \text{ } v_{75}. P (v_{10} \text{ says } (v_{74} \text{ eqf } v_{75}))) \wedge$
 $(\forall v_{10} \text{ } v_{76} \text{ } v_{77}. P (v_{10} \text{ says } v_{76} \text{ says } v_{77})) \wedge$
 $(\forall v_{10} \text{ } v_{78} \text{ } v_{79}. P (v_{10} \text{ says } v_{78} \text{ speaks_for } v_{79})) \wedge$

$$\begin{aligned}
& (\forall v_{10} v_{80} v_{81}. P (v_{10} \text{ says } v_{80} \text{ controls } v_{81})) \wedge \\
& (\forall v_{10} v_{82} v_{83} v_{84}. P (v_{10} \text{ says reps } v_{82} v_{83} v_{84})) \wedge \\
& (\forall v_{10} v_{85} v_{86}. P (v_{10} \text{ says } v_{85} \text{ domi } v_{86})) \wedge \\
& (\forall v_{10} v_{87} v_{88}. P (v_{10} \text{ says } v_{87} \text{ eqi } v_{88})) \wedge \\
& (\forall v_{10} v_{89} v_{90}. P (v_{10} \text{ says } v_{89} \text{ doms } v_{90})) \wedge \\
& (\forall v_{10} v_{91} v_{92}. P (v_{10} \text{ says } v_{91} \text{ eqs } v_{92})) \wedge \\
& (\forall v_{10} v_{93} v_{94}. P (v_{10} \text{ says } v_{93} \text{ eqn } v_{94})) \wedge \\
& (\forall v_{10} v_{95} v_{96}. P (v_{10} \text{ says } v_{95} \text{ lte } v_{96})) \wedge \\
& (\forall v_{10} v_{97} v_{98}. P (v_{10} \text{ says } v_{97} \text{ lt } v_{98})) \wedge \\
& (\forall v_{12} v_{13}. P (v_{12} \text{ speaks_for } v_{13})) \wedge \\
& (\forall v_{14} v_{15}. P (v_{14} \text{ controls } v_{15})) \wedge \\
& (\forall v_{16} v_{17} v_{18}. P (\text{reps } v_{16} v_{17} v_{18})) \wedge \\
& (\forall v_{19} v_{20}. P (v_{19} \text{ domi } v_{20})) \wedge \\
& (\forall v_{21} v_{22}. P (v_{21} \text{ eqi } v_{22})) \wedge \\
& (\forall v_{23} v_{24}. P (v_{23} \text{ doms } v_{24})) \wedge \\
& (\forall v_{25} v_{26}. P (v_{25} \text{ eqs } v_{26})) \wedge (\forall v_{27} v_{28}. P (v_{27} \text{ eqn } v_{28})) \wedge \\
& (\forall v_{29} v_{30}. P (v_{29} \text{ lte } v_{30})) \wedge (\forall v_{31} v_{32}. P (v_{31} \text{ lt } v_{32})) \Rightarrow \\
& \forall v. P v
\end{aligned}$$

[conductPBNS_def]

$$\begin{aligned}
& \vdash (\text{conductPBNS CONDUCT_PB (exec (PL securePB))} = \text{SECURE_PB}) \wedge \\
& (\text{conductPBNS CONDUCT_PB (exec (PL plIncompletePB))} = \\
& \quad \text{CONDUCT_PB}) \wedge \\
& (\text{conductPBNS SECURE_PB (exec (PSG actionsInPB))} = \\
& \quad \text{ACTIONS_IN_PB}) \wedge \\
& (\text{conductPBNS SECURE_PB (exec (PSG psgIncompletePB))} = \\
& \quad \text{SECURE_PB}) \wedge \\
& (\text{conductPBNS ACTIONS_IN_PB (exec (PL withdrawPB))} = \\
& \quad \text{WITHDRAW_PB}) \wedge \\
& (\text{conductPBNS ACTIONS_IN_PB (exec (PL plIncompletePB))} = \\
& \quad \text{ACTIONS_IN_PB}) \wedge \\
& (\text{conductPBNS WITHDRAW_PB (exec (PL completePB))} = \\
& \quad \text{COMPLETE_PB}) \wedge \\
& (\text{conductPBNS WITHDRAW_PB (exec (PL plIncompletePB))} = \\
& \quad \text{WITHDRAW_PB}) \wedge (\text{conductPBNS } s \text{ (trap (PL cmd'))} = s) \wedge \\
& (\text{conductPBNS } s \text{ (trap (PSG cmd))} = s) \wedge \\
& (\text{conductPBNS } s \text{ (discard (PL cmd'))} = s) \wedge \\
& (\text{conductPBNS } s \text{ (discard (PSG cmd))} = s)
\end{aligned}$$

[conductPBNS_ind]

$$\begin{aligned}
& \vdash \forall P. \\
& \quad P \text{ CONDUCT_PB (exec (PL securePB))} \wedge \\
& \quad P \text{ CONDUCT_PB (exec (PL plIncompletePB))} \wedge \\
& \quad P \text{ SECURE_PB (exec (PSG actionsInPB))} \wedge \\
& \quad P \text{ SECURE_PB (exec (PSG psgIncompletePB))} \wedge \\
& \quad P \text{ ACTIONS_IN_PB (exec (PL withdrawPB))} \wedge \\
& \quad P \text{ ACTIONS_IN_PB (exec (PL plIncompletePB))} \wedge \\
& \quad P \text{ WITHDRAW_PB (exec (PL completePB))} \wedge \\
& \quad P \text{ WITHDRAW_PB (exec (PL plIncompletePB))} \wedge
\end{aligned}$$

$$\begin{aligned}
& (\forall s \text{ cmd}. P \ s \ (\text{trap} \ (\text{PL} \ \text{cmd}))) \wedge \\
& (\forall s \text{ cmd}. P \ s \ (\text{trap} \ (\text{PSG} \ \text{cmd}))) \wedge \\
& (\forall s \text{ cmd}. P \ s \ (\text{discard} \ (\text{PL} \ \text{cmd}))) \wedge \\
& (\forall s \text{ cmd}. P \ s \ (\text{discard} \ (\text{PSG} \ \text{cmd}))) \wedge \\
& P \ \text{CONDUCT_PB} \ (\text{exec} \ (\text{PL} \ \text{withdrawPB})) \wedge \\
& P \ \text{CONDUCT_PB} \ (\text{exec} \ (\text{PL} \ \text{completePB})) \wedge \\
& (\forall v_{11}. P \ \text{CONDUCT_PB} \ (\text{exec} \ (\text{PSG} \ v_{11}))) \wedge \\
& (\forall v_{13}. P \ \text{SECURE_PB} \ (\text{exec} \ (\text{PL} \ v_{13}))) \wedge \\
& P \ \text{ACTIONS_IN_PB} \ (\text{exec} \ (\text{PL} \ \text{securePB})) \wedge \\
& P \ \text{ACTIONS_IN_PB} \ (\text{exec} \ (\text{PL} \ \text{completePB})) \wedge \\
& (\forall v_{17}. P \ \text{ACTIONS_IN_PB} \ (\text{exec} \ (\text{PSG} \ v_{17}))) \wedge \\
& P \ \text{WITHDRAW_PB} \ (\text{exec} \ (\text{PL} \ \text{securePB})) \wedge \\
& P \ \text{WITHDRAW_PB} \ (\text{exec} \ (\text{PL} \ \text{withdrawPB})) \wedge \\
& (\forall v_{20}. P \ \text{WITHDRAW_PB} \ (\text{exec} \ (\text{PSG} \ v_{20}))) \wedge \\
& (\forall v_{21}. P \ \text{COMPLETE_PB} \ (\text{exec} \ v_{21})) \Rightarrow \\
& \forall v \ v_1. P \ v \ v_1
\end{aligned}$$

[conductPBOut_def]

$$\begin{aligned}
& \vdash (\text{conductPBOut} \ \text{CONDUCT_PB} \ (\text{exec} \ (\text{PL} \ \text{securePB}))) = \text{ConductPB} \wedge \\
& (\text{conductPBOut} \ \text{CONDUCT_PB} \ (\text{exec} \ (\text{PL} \ \text{plIncompletePB}))) = \\
& \quad \text{ConductPB} \wedge \\
& (\text{conductPBOut} \ \text{SECURE_PB} \ (\text{exec} \ (\text{PSG} \ \text{actionsInPB}))) = \\
& \quad \text{SecurePB} \wedge \\
& (\text{conductPBOut} \ \text{SECURE_PB} \ (\text{exec} \ (\text{PSG} \ \text{psgIncompletePB}))) = \\
& \quad \text{SecurePB} \wedge \\
& (\text{conductPBOut} \ \text{ACTIONS_IN_PB} \ (\text{exec} \ (\text{PL} \ \text{withdrawPB}))) = \\
& \quad \text{ActionsInPB} \wedge \\
& (\text{conductPBOut} \ \text{ACTIONS_IN_PB} \ (\text{exec} \ (\text{PL} \ \text{plIncompletePB}))) = \\
& \quad \text{ActionsInPB} \wedge \\
& (\text{conductPBOut} \ \text{WITHDRAW_PB} \ (\text{exec} \ (\text{PL} \ \text{completePB}))) = \\
& \quad \text{WithdrawPB} \wedge \\
& (\text{conductPBOut} \ \text{WITHDRAW_PB} \ (\text{exec} \ (\text{PL} \ \text{plIncompletePB}))) = \\
& \quad \text{WithdrawPB} \wedge \\
& (\text{conductPBOut} \ s \ (\text{trap} \ (\text{PL} \ \text{cmd}')) = \text{unAuthorized}) \wedge \\
& (\text{conductPBOut} \ s \ (\text{trap} \ (\text{PSG} \ \text{cmd}))) = \text{unAuthorized} \wedge \\
& (\text{conductPBOut} \ s \ (\text{discard} \ (\text{PL} \ \text{cmd}')) = \text{unAuthenticated}) \wedge \\
& (\text{conductPBOut} \ s \ (\text{discard} \ (\text{PSG} \ \text{cmd}))) = \text{unAuthenticated}
\end{aligned}$$

[conductPBOut_ind]

$$\begin{aligned}
& \vdash \forall P. \\
& \quad P \ \text{CONDUCT_PB} \ (\text{exec} \ (\text{PL} \ \text{securePB})) \wedge \\
& \quad P \ \text{CONDUCT_PB} \ (\text{exec} \ (\text{PL} \ \text{plIncompletePB})) \wedge \\
& \quad P \ \text{SECURE_PB} \ (\text{exec} \ (\text{PSG} \ \text{actionsInPB})) \wedge \\
& \quad P \ \text{SECURE_PB} \ (\text{exec} \ (\text{PSG} \ \text{psgIncompletePB})) \wedge \\
& \quad P \ \text{ACTIONS_IN_PB} \ (\text{exec} \ (\text{PL} \ \text{withdrawPB})) \wedge \\
& \quad P \ \text{ACTIONS_IN_PB} \ (\text{exec} \ (\text{PL} \ \text{plIncompletePB})) \wedge \\
& \quad P \ \text{WITHDRAW_PB} \ (\text{exec} \ (\text{PL} \ \text{completePB})) \wedge \\
& \quad P \ \text{WITHDRAW_PB} \ (\text{exec} \ (\text{PL} \ \text{plIncompletePB})) \wedge \\
& \quad (\forall s \ \text{cmd}. P \ s \ (\text{trap} \ (\text{PL} \ \text{cmd}))) \wedge
\end{aligned}$$

$$\begin{aligned}
& (\forall s \text{ cmd}. P \ s \ (\text{trap} \ (\text{PSG} \ \text{cmd}))) \wedge \\
& (\forall s \text{ cmd}. P \ s \ (\text{discard} \ (\text{PL} \ \text{cmd}))) \wedge \\
& (\forall s \text{ cmd}. P \ s \ (\text{discard} \ (\text{PSG} \ \text{cmd}))) \wedge \\
& P \ \text{CONDUCT_PB} \ (\text{exec} \ (\text{PL} \ \text{withdrawPB})) \wedge \\
& P \ \text{CONDUCT_PB} \ (\text{exec} \ (\text{PL} \ \text{completePB})) \wedge \\
& (\forall v_{11}. P \ \text{CONDUCT_PB} \ (\text{exec} \ (\text{PSG} \ v_{11}))) \wedge \\
& (\forall v_{13}. P \ \text{SECURE_PB} \ (\text{exec} \ (\text{PL} \ v_{13}))) \wedge \\
& P \ \text{ACTIONS_IN_PB} \ (\text{exec} \ (\text{PL} \ \text{securePB})) \wedge \\
& P \ \text{ACTIONS_IN_PB} \ (\text{exec} \ (\text{PL} \ \text{completePB})) \wedge \\
& (\forall v_{17}. P \ \text{ACTIONS_IN_PB} \ (\text{exec} \ (\text{PSG} \ v_{17}))) \wedge \\
& P \ \text{WITHDRAW_PB} \ (\text{exec} \ (\text{PL} \ \text{securePB})) \wedge \\
& P \ \text{WITHDRAW_PB} \ (\text{exec} \ (\text{PL} \ \text{withdrawPB})) \wedge \\
& (\forall v_{20}. P \ \text{WITHDRAW_PB} \ (\text{exec} \ (\text{PSG} \ v_{20}))) \wedge \\
& (\forall v_{21}. P \ \text{COMPLETE_PB} \ (\text{exec} \ v_{21})) \Rightarrow \\
& \forall v \ v_1. P \ v \ v_1
\end{aligned}$$

[PlatoonLeader_exec_plCommandPB_justified_thm]

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

[PlatoonLeader_plCommandPB_lemma]

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

[PlatoonSergeant_exec_psgCommandPB_justified_thm]

$$\begin{aligned}
& \vdash \forall NS \ Out \ M \ Oi \ Os. \\
& \quad \text{TR} \ (M, Oi, Os) \ (\text{exec} \ (\text{SLc} \ (\text{PSG} \ psgCommand)))
\end{aligned}$$

```

(CFG authTestConductPB ssmConductPBStateInterp
  (secContextConductPB plCommand psgCommand incomplete)
  (Name PlatoonSergeant says
    prop (SOME (SLc (PSG psgCommand))))::ins) s outs)
(CFG authTestConductPB ssmConductPBStateInterp
  (secContextConductPB plCommand psgCommand incomplete)
  ins (NS s (exec (SLc (PSG psgCommand))))
  (Out s (exec (SLc (PSG psgCommand))))::outs))  $\iff$ 
authTestConductPB
  (Name PlatoonSergeant says
    prop (SOME (SLc (PSG psgCommand))))  $\wedge$ 
CFGInterpret (M, Oi, Os)
  (CFG authTestConductPB ssmConductPBStateInterp
    (secContextConductPB plCommand psgCommand incomplete)
    (Name PlatoonSergeant says
      prop (SOME (SLc (PSG psgCommand))))::ins) s outs)  $\wedge$ 
  (M, Oi, Os) sat prop (SOME (SLc (PSG psgCommand))))

```

[PlatoonSergeant_psgCommandPB_lemma]

```

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

```

11 ConductPBType Theory

Built: 10 June 2018

Parent Theories: indexedLists, patternMatches

11.1 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

```

11.2 Theorems

[plCommandPB_distinct_clauses]

$$\vdash \text{securePB} \neq \text{withdrawPB} \wedge \text{securePB} \neq \text{completePB} \wedge \\ \text{securePB} \neq \text{plIncompletePB} \wedge \text{withdrawPB} \neq \text{completePB} \wedge \\ \text{withdrawPB} \neq \text{plIncompletePB} \wedge \text{completePB} \neq \text{plIncompletePB}$$

[psgCommandPB_distinct_clauses]

$$\vdash \text{actionsInPB} \neq \text{psgIncompletePB}$$

[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]

$$\vdash \text{CONDUCT_PB} \neq \text{SECURE_PB} \wedge \text{CONDUCT_PB} \neq \text{ACTIONS_IN_PB} \wedge \\ \text{CONDUCT_PB} \neq \text{WITHDRAW_PB} \wedge \text{CONDUCT_PB} \neq \text{COMPLETE_PB} \wedge \\ \text{SECURE_PB} \neq \text{ACTIONS_IN_PB} \wedge \text{SECURE_PB} \neq \text{WITHDRAW_PB} \wedge \\ \text{SECURE_PB} \neq \text{COMPLETE_PB} \wedge \text{ACTIONS_IN_PB} \neq \text{WITHDRAW_PB} \wedge \\ \text{ACTIONS_IN_PB} \neq \text{COMPLETE_PB} \wedge \text{WITHDRAW_PB} \neq \text{COMPLETE_PB}$$

12 ssmMoveToORP Theory

Built: 10 June 2018

Parent Theories: MoveToORPType, ssm11, OMNIType

12.1 Definitions

[secContextMoveToORP_def]

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

[ssmMoveToORPStateInterp_def]

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

12.2 Theorems

[authTestMoveToORP_cmd_reject_lemma]

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

[authTestMoveToORP_def]

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

$(\text{authTestMoveToORP } (v_{21} \text{ eqi } v_{22}) \iff F) \wedge$
 $(\text{authTestMoveToORP } (v_{23} \text{ doms } v_{24}) \iff F) \wedge$
 $(\text{authTestMoveToORP } (v_{25} \text{ eqs } v_{26}) \iff F) \wedge$
 $(\text{authTestMoveToORP } (v_{27} \text{ eqn } v_{28}) \iff F) \wedge$
 $(\text{authTestMoveToORP } (v_{29} \text{ lte } v_{30}) \iff F) \wedge$
 $(\text{authTestMoveToORP } (v_{31} \text{ lt } v_{32}) \iff F)$

[authTestMoveToORP_ind]

$\vdash \forall P.$

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

[moveToORPNS_def]

$\vdash (\text{moveToORPNS MOVE_TO_ORP } (\text{exec } (\text{SLc pltForm})) = \text{PLT_FORM}) \wedge$
 $(\text{moveToORPNS MOVE_TO_ORP } (\text{exec } (\text{SLc incomplete})) =$
 $\text{MOVE_TO_ORP}) \wedge$
 $(\text{moveToORPNS PLT_FORM } (\text{exec } (\text{SLc pltMove})) = \text{PLT_MOVE}) \wedge$
 $(\text{moveToORPNS PLT_FORM } (\text{exec } (\text{SLc incomplete})) = \text{PLT_FORM}) \wedge$
 $(\text{moveToORPNS PLT_MOVE } (\text{exec } (\text{SLc pltSecureHalt})) =$

$\text{PLT_SECURE_HALT}) \wedge$
 $(\text{moveToORPNS PLT_MOVE (exec (SLc incomplete))} = \text{PLT_MOVE}) \wedge$
 $(\text{moveToORPNS PLT_SECURE_HALT (exec (SLc complete))} =$
 $\text{COMPLETE}) \wedge$
 $(\text{moveToORPNS PLT_SECURE_HALT (exec (SLc incomplete))} =$
 $\text{PLT_SECURE_HALT}) \wedge (\text{moveToORPNS } s \text{ (trap (SLc cmd))} = s) \wedge$
 $(\text{moveToORPNS } s \text{ (discard (SLc cmd))} = s)$

[moveToORPNS_ind]

$\vdash \forall P.$
 $P \text{ MOVE_TO_ORP (exec (SLc pltForm))} \wedge$
 $P \text{ MOVE_TO_ORP (exec (SLc incomplete))} \wedge$
 $P \text{ PLT_FORM (exec (SLc pltMove))} \wedge$
 $P \text{ PLT_FORM (exec (SLc incomplete))} \wedge$
 $P \text{ PLT_MOVE (exec (SLc pltSecureHalt))} \wedge$
 $P \text{ PLT_MOVE (exec (SLc incomplete))} \wedge$
 $P \text{ PLT_SECURE_HALT (exec (SLc complete))} \wedge$
 $P \text{ PLT_SECURE_HALT (exec (SLc incomplete))} \wedge$
 $(\forall s \text{ cmd. } P \text{ } s \text{ (trap (SLc cmd))}) \wedge$
 $(\forall s \text{ cmd. } P \text{ } s \text{ (discard (SLc cmd))}) \wedge$
 $(\forall s \text{ } v_6. P \text{ } s \text{ (discard (ESCc } v_6))) \wedge$
 $(\forall s \text{ } v_9. P \text{ } s \text{ (trap (ESCc } v_9))) \wedge$
 $(\forall v_{12}. P \text{ MOVE_TO_ORP (exec (ESCc } v_{12}))) \wedge$
 $P \text{ MOVE_TO_ORP (exec (SLc pltMove))} \wedge$
 $P \text{ MOVE_TO_ORP (exec (SLc pltSecureHalt))} \wedge$
 $P \text{ MOVE_TO_ORP (exec (SLc complete))} \wedge$
 $(\forall v_{15}. P \text{ PLT_FORM (exec (ESCc } v_{15}))) \wedge$
 $P \text{ PLT_FORM (exec (SLc pltForm))} \wedge$
 $P \text{ PLT_FORM (exec (SLc pltSecureHalt))} \wedge$
 $P \text{ PLT_FORM (exec (SLc complete))} \wedge$
 $(\forall v_{18}. P \text{ PLT_MOVE (exec (ESCc } v_{18}))) \wedge$
 $P \text{ PLT_MOVE (exec (SLc pltForm))} \wedge$
 $P \text{ PLT_MOVE (exec (SLc pltMove))} \wedge$
 $P \text{ PLT_MOVE (exec (SLc complete))} \wedge$
 $(\forall v_{21}. P \text{ PLT_SECURE_HALT (exec (ESCc } v_{21}))) \wedge$
 $P \text{ PLT_SECURE_HALT (exec (SLc pltForm))} \wedge$
 $P \text{ PLT_SECURE_HALT (exec (SLc pltMove))} \wedge$
 $P \text{ PLT_SECURE_HALT (exec (SLc pltSecureHalt))} \wedge$
 $(\forall v_{23}. P \text{ COMPLETE (exec } v_{23})) \Rightarrow$
 $\forall v \text{ } v_1. P \text{ } v \text{ } v_1$

[moveToORPOut_def]

$\vdash (\text{moveToORPOut MOVE_TO_ORP (exec (SLc pltForm))} = \text{PLTForm}) \wedge$
 $(\text{moveToORPOut MOVE_TO_ORP (exec (SLc incomplete))} =$
 $\text{MoveToORP}) \wedge$
 $(\text{moveToORPOut PLT_FORM (exec (SLc pltMove))} = \text{PLTMove}) \wedge$
 $(\text{moveToORPOut PLT_FORM (exec (SLc incomplete))} = \text{PLTForm}) \wedge$
 $(\text{moveToORPOut PLT_MOVE (exec (SLc pltSecureHalt))} =$
 $\text{PLTSecureHalt}) \wedge$

```

(moveToORPOut PLT_MOVE (exec (SLc incomplete)) = PLTMove) ∧
(moveToORPOut PLT_SECURE_HALT (exec (SLc complete)) =
  Complete) ∧
(moveToORPOut PLT_SECURE_HALT (exec (SLc incomplete)) =
  PLTSecureHalt) ∧
(moveToORPOut s (trap (SLc cmd)) = unauthorized) ∧
(moveToORPOut s (discard (SLc cmd)) = unauthenticated)

```

[moveToORPOut_ind]

```

⊢ ∀ P.
  P MOVE_TO_ORP (exec (SLc pltForm)) ∧
  P MOVE_TO_ORP (exec (SLc incomplete)) ∧
  P PLT_FORM (exec (SLc pltMove)) ∧
  P PLT_FORM (exec (SLc incomplete)) ∧
  P PLT_MOVE (exec (SLc pltSecureHalt)) ∧
  P PLT_MOVE (exec (SLc incomplete)) ∧
  P PLT_SECURE_HALT (exec (SLc complete)) ∧
  P PLT_SECURE_HALT (exec (SLc incomplete)) ∧
  (∀ s cmd. P s (trap (SLc cmd))) ∧
  (∀ s cmd. P s (discard (SLc cmd))) ∧
  (∀ s v6. P s (discard (ESCc v6))) ∧
  (∀ s v9. P s (trap (ESCc v9))) ∧
  (∀ v12. P MOVE_TO_ORP (exec (ESCc v12))) ∧
  P MOVE_TO_ORP (exec (SLc pltMove)) ∧
  P MOVE_TO_ORP (exec (SLc pltSecureHalt)) ∧
  P MOVE_TO_ORP (exec (SLc complete)) ∧
  (∀ v15. P PLT_FORM (exec (ESCc v15))) ∧
  P PLT_FORM (exec (SLc pltForm)) ∧
  P PLT_FORM (exec (SLc pltSecureHalt)) ∧
  P PLT_FORM (exec (SLc complete)) ∧
  (∀ v18. P PLT_MOVE (exec (ESCc v18))) ∧
  P PLT_MOVE (exec (SLc pltForm)) ∧
  P PLT_MOVE (exec (SLc pltMove)) ∧
  P PLT_MOVE (exec (SLc complete)) ∧
  (∀ v21. P PLT_SECURE_HALT (exec (ESCc v21))) ∧
  P PLT_SECURE_HALT (exec (SLc pltForm)) ∧
  P PLT_SECURE_HALT (exec (SLc pltMove)) ∧
  P PLT_SECURE_HALT (exec (SLc pltSecureHalt)) ∧
  (∀ v23. P COMPLETE (exec v23)) ⇒
  ∀ v v1. P v v1

```

[PlatoonLeader_exec_slCommand_justified_thm]

```

⊢ ∀ NS Out M Oi Os.
  TR (M, Oi, Os) (exec (SLc slCommand))
  (CFG authTestMoveToORP ssmMoveToORPStateInterp
    (secContextMoveToORP slCommand)
    (Name PlatoonLeader says prop (SOME (SLc slCommand)) ::
      ins) s outs)
  (CFG authTestMoveToORP ssmMoveToORPStateInterp

```

```

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

```

[PlatoonLeader_slCommand_lemma]

```

 $\vdash$  CFGInterpret (M, Oi, Os)
(CFG authTestMoveToORP ssmMoveToORPStateInterp
(secContextMoveToORP slCommand)
(Name PlatoonLeader says prop (SOME (SLc slCommand))::
ins) s outs)  $\Rightarrow$ 
(M, Oi, Os) sat prop (SOME (SLc slCommand))

```

13 MoveToORPType Theory

Built: 10 June 2018

Parent Theories: indexedLists, patternMatches

13.1 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

```

13.2 Theorems

[slCommand_distinct_clauses]

```

 $\vdash$  pltForm  $\neq$  pltMove  $\wedge$  pltForm  $\neq$  pltSecureHalt  $\wedge$ 
pltForm  $\neq$  complete  $\wedge$  pltForm  $\neq$  incomplete  $\wedge$ 
pltMove  $\neq$  pltSecureHalt  $\wedge$  pltMove  $\neq$  complete  $\wedge$ 
pltMove  $\neq$  incomplete  $\wedge$  pltSecureHalt  $\neq$  complete  $\wedge$ 
pltSecureHalt  $\neq$  incomplete  $\wedge$  complete  $\neq$  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

```

14 ssmMoveToPB Theory

Built: 10 June 2018

Parent Theories: MoveToPBType, ssm11, OMNIType

14.1 Definitions

[secContextMoveToPB_def]

```

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

```

[ssmMoveToPBStateInterp_def]

```

⊢ ∀ state. ssmMoveToPBStateInterp state = TT

```

14.2 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) ∧

```

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

$[\text{authTestMoveToPB_ind}]$

$\vdash \forall P.$

$(\forall \text{cmd}. P (\text{Name PlatoonLeader says prop cmd})) \wedge P \text{ TT} \wedge$
 $P \text{ FF} \wedge (\forall v. P (\text{prop } v)) \wedge (\forall v_1. P (\text{notf } v_1)) \wedge$
 $(\forall v_2 v_3. P (v_2 \text{ andf } v_3)) \wedge (\forall v_4 v_5. P (v_4 \text{ orf } v_5)) \wedge$
 $(\forall v_6 v_7. P (v_6 \text{ impf } v_7)) \wedge (\forall v_8 v_9. P (v_8 \text{ eqf } v_9)) \wedge$
 $(\forall v_{10}. P (v_{10} \text{ says TT})) \wedge (\forall v_{10}. P (v_{10} \text{ says FF})) \wedge$
 $(\forall v_{133} v_{134} v_{66}. P (v_{133} \text{ meet } v_{134} \text{ says prop } v_{66})) \wedge$
 $(\forall v_{135} v_{136} v_{66}. P (v_{135} \text{ quoting } v_{136} \text{ says prop } v_{66})) \wedge$
 $(\forall v_{10} v_{67}. P (v_{10} \text{ says notf } v_{67})) \wedge$
 $(\forall v_{10} v_{68} v_{69}. P (v_{10} \text{ says } (v_{68} \text{ andf } v_{69}))) \wedge$
 $(\forall v_{10} v_{70} v_{71}. P (v_{10} \text{ says } (v_{70} \text{ orf } v_{71}))) \wedge$
 $(\forall v_{10} v_{72} v_{73}. P (v_{10} \text{ says } (v_{72} \text{ impf } v_{73}))) \wedge$
 $(\forall v_{10} v_{74} v_{75}. P (v_{10} \text{ says } (v_{74} \text{ eqf } v_{75}))) \wedge$
 $(\forall v_{10} v_{76} v_{77}. P (v_{10} \text{ says } v_{76} \text{ says } v_{77})) \wedge$

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

[moveToPBNS_def]

$$\begin{aligned}
& \vdash (\text{moveToPBNS MOVE_TO_PB (exec (SLc pltForm))} = \text{PLT_FORM}) \wedge \\
& (\text{moveToPBNS MOVE_TO_PB (exec (SLc incomplete))} = \\
& \quad \text{MOVE_TO_PB}) \wedge \\
& (\text{moveToPBNS PLT_FORM (exec (SLc pltMove))} = \text{PLT_MOVE}) \wedge \\
& (\text{moveToPBNS PLT_FORM (exec (SLc incomplete))} = \text{PLT_FORM}) \wedge \\
& (\text{moveToPBNS PLT_MOVE (exec (SLc pltHalt))} = \text{PLT_HALT}) \wedge \\
& (\text{moveToPBNS PLT_MOVE (exec (SLc incomplete))} = \text{PLT_MOVE}) \wedge \\
& (\text{moveToPBNS PLT_HALT (exec (SLc complete))} = \text{COMPLETE}) \wedge \\
& (\text{moveToPBNS PLT_HALT (exec (SLc incomplete))} = \text{PLT_HALT}) \wedge \\
& (\text{moveToPBNS } s \text{ (trap (SLc cmd))} = s) \wedge \\
& (\text{moveToPBNS } s \text{ (discard (SLc cmd))} = s)
\end{aligned}$$

[moveToPBNS_ind]

$$\begin{aligned}
& \vdash \forall P. \\
& \quad P \text{ MOVE_TO_PB (exec (SLc pltForm))} \wedge \\
& \quad P \text{ MOVE_TO_PB (exec (SLc incomplete))} \wedge \\
& \quad P \text{ PLT_FORM (exec (SLc pltMove))} \wedge \\
& \quad P \text{ PLT_FORM (exec (SLc incomplete))} \wedge \\
& \quad P \text{ PLT_MOVE (exec (SLc pltHalt))} \wedge \\
& \quad P \text{ PLT_MOVE (exec (SLc incomplete))} \wedge \\
& \quad P \text{ PLT_HALT (exec (SLc complete))} \wedge \\
& \quad P \text{ PLT_HALT (exec (SLc incomplete))} \wedge \\
& \quad (\forall s \text{ cmd}. P s \text{ (trap (SLc cmd))}) \wedge \\
& \quad (\forall s \text{ cmd}. P s \text{ (discard (SLc cmd))}) \wedge \\
& \quad (\forall s v_6. P s \text{ (discard (ESCc } v_6 \text{))}) \wedge \\
& \quad (\forall s v_9. P s \text{ (trap (ESCc } v_9 \text{))}) \wedge \\
& \quad (\forall v_{12}. P \text{ MOVE_TO_PB (exec (ESCc } v_{12} \text{))}) \wedge \\
& \quad P \text{ MOVE_TO_PB (exec (SLc pltMove))} \wedge
\end{aligned}$$

$P \text{ MOVE_TO_PB } (\text{exec } (\text{SLc pltHalt})) \wedge$
 $P \text{ MOVE_TO_PB } (\text{exec } (\text{SLc complete})) \wedge$
 $(\forall v_{15}. P \text{ PLT_FORM } (\text{exec } (\text{ESCc } v_{15}))) \wedge$
 $P \text{ PLT_FORM } (\text{exec } (\text{SLc pltForm})) \wedge$
 $P \text{ PLT_FORM } (\text{exec } (\text{SLc pltHalt})) \wedge$
 $P \text{ PLT_FORM } (\text{exec } (\text{SLc complete})) \wedge$
 $(\forall v_{18}. P \text{ PLT_MOVE } (\text{exec } (\text{ESCc } v_{18}))) \wedge$
 $P \text{ PLT_MOVE } (\text{exec } (\text{SLc pltForm})) \wedge$
 $P \text{ PLT_MOVE } (\text{exec } (\text{SLc pltMove})) \wedge$
 $P \text{ PLT_MOVE } (\text{exec } (\text{SLc complete})) \wedge$
 $(\forall v_{21}. P \text{ PLT_HALT } (\text{exec } (\text{ESCc } v_{21}))) \wedge$
 $P \text{ PLT_HALT } (\text{exec } (\text{SLc pltForm})) \wedge$
 $P \text{ PLT_HALT } (\text{exec } (\text{SLc pltMove})) \wedge$
 $P \text{ PLT_HALT } (\text{exec } (\text{SLc pltHalt})) \wedge$
 $(\forall v_{23}. P \text{ COMPLETE } (\text{exec } v_{23})) \Rightarrow$
 $\forall v \ v_1. P \ v \ v_1$

[moveToPBOut_def]

$\vdash (\text{moveToPBOut } \text{MOVE_TO_PB } (\text{exec } (\text{SLc pltForm})) = \text{PLTForm}) \wedge$
 $(\text{moveToPBOut } \text{MOVE_TO_PB } (\text{exec } (\text{SLc incomplete})) = \text{MoveToPB}) \wedge$
 $(\text{moveToPBOut } \text{PLT_FORM } (\text{exec } (\text{SLc pltMove})) = \text{PLTMove}) \wedge$
 $(\text{moveToPBOut } \text{PLT_FORM } (\text{exec } (\text{SLc incomplete})) = \text{PLTForm}) \wedge$
 $(\text{moveToPBOut } \text{PLT_MOVE } (\text{exec } (\text{SLc pltHalt})) = \text{PLTHalt}) \wedge$
 $(\text{moveToPBOut } \text{PLT_MOVE } (\text{exec } (\text{SLc incomplete})) = \text{PLTMove}) \wedge$
 $(\text{moveToPBOut } \text{PLT_HALT } (\text{exec } (\text{SLc complete})) = \text{Complete}) \wedge$
 $(\text{moveToPBOut } \text{PLT_HALT } (\text{exec } (\text{SLc incomplete})) = \text{PLTHalt}) \wedge$
 $(\text{moveToPBOut } s \ (\text{trap } (\text{SLc cmd})) = \text{unAuthorized}) \wedge$
 $(\text{moveToPBOut } s \ (\text{discard } (\text{SLc cmd})) = \text{unAuthenticated})$

[moveToPBOut_ind]

$\vdash \forall P.$
 $P \text{ MOVE_TO_PB } (\text{exec } (\text{SLc pltForm})) \wedge$
 $P \text{ MOVE_TO_PB } (\text{exec } (\text{SLc incomplete})) \wedge$
 $P \text{ PLT_FORM } (\text{exec } (\text{SLc pltMove})) \wedge$
 $P \text{ PLT_FORM } (\text{exec } (\text{SLc incomplete})) \wedge$
 $P \text{ PLT_MOVE } (\text{exec } (\text{SLc pltHalt})) \wedge$
 $P \text{ PLT_MOVE } (\text{exec } (\text{SLc incomplete})) \wedge$
 $P \text{ PLT_HALT } (\text{exec } (\text{SLc complete})) \wedge$
 $P \text{ PLT_HALT } (\text{exec } (\text{SLc incomplete})) \wedge$
 $(\forall s \ \text{cmd}. P \ s \ (\text{trap } (\text{SLc cmd}))) \wedge$
 $(\forall s \ \text{cmd}. P \ s \ (\text{discard } (\text{SLc cmd}))) \wedge$
 $(\forall s \ v_6. P \ s \ (\text{discard } (\text{ESCc } v_6))) \wedge$
 $(\forall s \ v_9. P \ s \ (\text{trap } (\text{ESCc } v_9))) \wedge$
 $(\forall v_{12}. P \text{ MOVE_TO_PB } (\text{exec } (\text{ESCc } v_{12}))) \wedge$
 $P \text{ MOVE_TO_PB } (\text{exec } (\text{SLc pltMove})) \wedge$
 $P \text{ MOVE_TO_PB } (\text{exec } (\text{SLc pltHalt})) \wedge$
 $P \text{ MOVE_TO_PB } (\text{exec } (\text{SLc complete})) \wedge$
 $(\forall v_{15}. P \text{ PLT_FORM } (\text{exec } (\text{ESCc } v_{15}))) \wedge$
 $P \text{ PLT_FORM } (\text{exec } (\text{SLc pltForm})) \wedge$

$$\begin{aligned}
 & P \text{ PLT_FORM } (\text{exec } (\text{SLc pltHalt})) \wedge \\
 & P \text{ PLT_FORM } (\text{exec } (\text{SLc complete})) \wedge \\
 & (\forall v_{18}. P \text{ PLT_MOVE } (\text{exec } (\text{ESCc } v_{18}))) \wedge \\
 & P \text{ PLT_MOVE } (\text{exec } (\text{SLc pltForm})) \wedge \\
 & P \text{ PLT_MOVE } (\text{exec } (\text{SLc pltMove})) \wedge \\
 & P \text{ PLT_MOVE } (\text{exec } (\text{SLc complete})) \wedge \\
 & (\forall v_{21}. P \text{ PLT_HALT } (\text{exec } (\text{ESCc } v_{21}))) \wedge \\
 & P \text{ PLT_HALT } (\text{exec } (\text{SLc pltForm})) \wedge \\
 & P \text{ PLT_HALT } (\text{exec } (\text{SLc pltMove})) \wedge \\
 & P \text{ PLT_HALT } (\text{exec } (\text{SLc pltHalt})) \wedge \\
 & (\forall v_{23}. P \text{ COMPLETE } (\text{exec } v_{23})) \Rightarrow \\
 & \forall v \ v_1. P \ v \ v_1
 \end{aligned}$$

[PlatoonLeader_exec_slCommand_justified_thm]

$$\begin{aligned}
 & \vdash \forall NS \text{ Out } M \text{ } Oi \text{ } Os. \\
 & \quad \text{TR } (M, Oi, Os) (\text{exec } (\text{SLc slCommand})) \\
 & \quad (\text{CFG authTestMoveToPB ssmMoveToPBStateInterp} \\
 & \quad \quad (\text{secContextMoveToPB slCommand}) \\
 & \quad \quad (\text{Name PlatoonLeader says prop (SOME (SLc slCommand))} :: \\
 & \quad \quad \quad \text{ins) s outs})) \\
 & \quad (\text{CFG authTestMoveToPB ssmMoveToPBStateInterp} \\
 & \quad \quad (\text{secContextMoveToPB slCommand}) \text{ ins} \\
 & \quad \quad (NS \text{ s } (\text{exec } (\text{SLc slCommand}))) \\
 & \quad \quad (\text{Out s } (\text{exec } (\text{SLc slCommand})) :: \text{outs})) \iff \\
 & \text{authTestMoveToPB} \\
 & \quad (\text{Name PlatoonLeader says prop (SOME (SLc slCommand))}) \wedge \\
 & \text{CFGInterpret } (M, Oi, Os) \\
 & \quad (\text{CFG authTestMoveToPB ssmMoveToPBStateInterp} \\
 & \quad \quad (\text{secContextMoveToPB slCommand}) \\
 & \quad \quad (\text{Name PlatoonLeader says prop (SOME (SLc slCommand))} :: \\
 & \quad \quad \quad \text{ins) s outs})) \wedge \\
 & (M, Oi, Os) \text{ sat prop (SOME (SLc slCommand))}
 \end{aligned}$$

[PlatoonLeader_slCommand_lemma]

$$\begin{aligned}
 & \vdash \text{CFGInterpret } (M, Oi, Os) \\
 & \quad (\text{CFG authTestMoveToPB ssmMoveToPBStateInterp} \\
 & \quad \quad (\text{secContextMoveToPB slCommand}) \\
 & \quad \quad (\text{Name PlatoonLeader says prop (SOME (SLc slCommand))} :: \\
 & \quad \quad \quad \text{ins) s outs})) \Rightarrow \\
 & (M, Oi, Os) \text{ sat prop (SOME (SLc slCommand))}
 \end{aligned}$$

15 MoveToPBType Theory

Built: 10 June 2018

Parent Theories: indexedLists, patternMatches

15.1 Datatypes

$slCommand = \text{pltForm} \mid \text{pltMove} \mid \text{pltHalt} \mid \text{complete} \mid \text{incomplete}$

$slOutput = \text{MoveToPB} \mid \text{PLTForm} \mid \text{PLTMove} \mid \text{PLTHalt} \mid \text{Complete}$
 $\mid \text{unAuthorized} \mid \text{unAuthenticated}$

$slState = \text{MOVE_TO_PB} \mid \text{PLT_FORM} \mid \text{PLT_MOVE} \mid \text{PLT_HALT} \mid \text{COMPLETE}$

$stateRole = \text{PlatoonLeader}$

15.2 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]

$\vdash \text{MOVE_TO_PB} \neq \text{PLT_FORM} \wedge \text{MOVE_TO_PB} \neq \text{PLT_MOVE} \wedge$
 $\text{MOVE_TO_PB} \neq \text{PLT_HALT} \wedge \text{MOVE_TO_PB} \neq \text{COMPLETE} \wedge$
 $\text{PLT_FORM} \neq \text{PLT_MOVE} \wedge \text{PLT_FORM} \neq \text{PLT_HALT} \wedge$
 $\text{PLT_FORM} \neq \text{COMPLETE} \wedge \text{PLT_MOVE} \neq \text{PLT_HALT} \wedge$
 $\text{PLT_MOVE} \neq \text{COMPLETE} \wedge \text{PLT_HALT} \neq \text{COMPLETE}$

16 ssmPlanPB Theory

Built: 10 June 2018

Parent Theories: PlanPBDef, ssm

16.1 Theorems

[inputOK_def]

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

[inputOK_ind]

$$\begin{aligned}
&\vdash \forall P. \\
&\quad (\forall cmd. P (\text{Name PlatoonLeader says prop } cmd)) \wedge \\
&\quad (\forall cmd. P (\text{Name PlatoonSergeant says prop } cmd)) \wedge P \ TT \wedge \\
&\quad P \ FF \wedge (\forall v. P (\text{prop } v)) \wedge (\forall v_1. P (\text{notf } v_1)) \wedge \\
&\quad (\forall v_2 \ v_3. P (v_2 \text{ andf } v_3)) \wedge (\forall v_4 \ v_5. P (v_4 \text{ orf } v_5)) \wedge \\
&\quad (\forall v_6 \ v_7. P (v_6 \text{ impf } v_7)) \wedge (\forall v_8 \ v_9. P (v_8 \text{ eqf } v_9)) \wedge \\
&\quad (\forall v_{10}. P (v_{10} \text{ says } TT)) \wedge (\forall v_{10}. P (v_{10} \text{ says } FF)) \wedge \\
&\quad (\forall v_{133} \ v_{134} \ v_{66}. P (v_{133} \text{ meet } v_{134} \text{ says prop } v_{66})) \wedge \\
&\quad (\forall v_{135} \ v_{136} \ v_{66}. P (v_{135} \text{ quoting } v_{136} \text{ says prop } v_{66})) \wedge \\
&\quad (\forall v_{10} \ v_{67}. P (v_{10} \text{ says notf } v_{67})) \wedge \\
&\quad (\forall v_{10} \ v_{68} \ v_{69}. P (v_{10} \text{ says } (v_{68} \text{ andf } v_{69}))) \wedge
\end{aligned}$$

$$\begin{aligned}
& (\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 } \text{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}$$

[planPBNS_def]

```

⊢ (planPBNS WARN0 (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 WARN0) ∧
(planPBNS PLAN_PB (exec x) =
  if getPlCom x = receiveMission then RECEIVE_MISSION
  else PLAN_PB) ∧
(planPBNS RECEIVE_MISSION (exec x) =
  if getPlCom x = warno then WARN0 else RECEIVE_MISSION) ∧
(planPBNS REPORT1 (exec x) =
  if getPlCom x = completePlan then COMPLETE_PLAN
  else REPORT1) ∧
(planPBNS COMPLETE_PLAN (exec x) =
  if getPlCom x = opoid then OPOID else COMPLETE_PLAN) ∧
(planPBNS OPOID (exec x) =
  if getPlCom x = supervise then SUPERVISE else OPOID) ∧
(planPBNS SUPERVISE (exec x) =
  if getPlCom x = report2 then REPORT2 else SUPERVISE) ∧
(planPBNS REPORT2 (exec x) =

```

if getPlCom x = complete **then** COMPLETE **else** REPORT2) \wedge
 (planPBNS s (trap v_0) = s) \wedge (planPBNS s (discard v_1) = s)

[planPBNS_ind]

$\vdash \forall P.$
 $(\forall x. P \text{ WARN0 (exec } x)) \wedge (\forall x. P \text{ PLAN_PB (exec } x)) \wedge$
 $(\forall x. P \text{ RECEIVE_MISSION (exec } x)) \wedge$
 $(\forall x. P \text{ REPORT1 (exec } x)) \wedge (\forall x. P \text{ COMPLETE_PLAN (exec } x)) \wedge$
 $(\forall x. P \text{ OPOID (exec } x)) \wedge (\forall x. P \text{ SUPERVISE (exec } x)) \wedge$
 $(\forall x. P \text{ REPORT2 (exec } x)) \wedge (\forall s \ v_0. P \ s \ (\text{trap } v_0)) \wedge$
 $(\forall s \ v_1. P \ s \ (\text{discard } v_1)) \wedge$
 $(\forall v_6. P \text{ TENTATIVE_PLAN (exec } v_6)) \wedge$
 $(\forall v_7. P \text{ INITIATE_MOVEMENT (exec } v_7)) \wedge$
 $(\forall v_8. P \text{ RECON (exec } v_8)) \wedge (\forall v_9. P \text{ COMPLETE (exec } v_9)) \Rightarrow$
 $\forall v \ v_1. P \ v \ v_1$

[planPBOut_def]

\vdash (planPBOut WARN0 (exec x) =
if
 (getRecon x = [SOME (SLc (PL recon))]) \wedge
 (getTentativePlan x = [SOME (SLc (PL tentativePlan))]) \wedge
 (getReport x = [SOME (SLc (PL report1))]) \wedge
 (getInitMove x = [SOME (SLc (PSG initiateMovement))])
then
 Report1
else unauthorized) \wedge
 (planPBOut PLAN_PB (exec x) =
if getPlCom x = receiveMission **then** ReceiveMission
else unauthorized) \wedge
 (planPBOut RECEIVE_MISSION (exec x) =
if getPlCom x = warno **then** Warno **else** unauthorized) \wedge
 (planPBOut REPORT1 (exec x) =
if getPlCom x = completePlan **then** CompletePlan
else unauthorized) \wedge
 (planPBOut COMPLETE_PLAN (exec x) =
if getPlCom x = opoid **then** Opoid **else** unauthorized) \wedge
 (planPBOut OPOID (exec x) =
if getPlCom x = supervise **then** Supervise
else unauthorized) \wedge
 (planPBOut SUPERVISE (exec x) =
if getPlCom x = report2 **then** Report2 **else** unauthorized) \wedge
 (planPBOut REPORT2 (exec x) =
if getPlCom x = complete **then** Complete **else** unauthorized) \wedge
 (planPBOut s (trap v_0) = unauthorized) \wedge
 (planPBOut s (discard v_1) = unAuthenticated)

[planPBOut_ind]

$\vdash \forall P.$
 $(\forall x. P \text{ WARN0 (exec } x)) \wedge (\forall x. P \text{ PLAN_PB (exec } x)) \wedge$

$$\begin{aligned}
& (\forall x. P \text{ RECEIVE_MISSION } (\text{exec } x)) \wedge \\
& (\forall x. P \text{ REPORT1 } (\text{exec } x)) \wedge (\forall x. P \text{ COMPLETE_PLAN } (\text{exec } x)) \wedge \\
& (\forall x. P \text{ OPOID } (\text{exec } x)) \wedge (\forall x. P \text{ SUPERVISE } (\text{exec } x)) \wedge \\
& (\forall x. P \text{ REPORT2 } (\text{exec } x)) \wedge (\forall s \ v_0. P \ s \ (\text{trap } v_0)) \wedge \\
& (\forall s \ v_1. P \ s \ (\text{discard } v_1)) \wedge \\
& (\forall v_6. P \text{ TENTATIVE_PLAN } (\text{exec } v_6)) \wedge \\
& (\forall v_7. P \text{ INITIATE_MOVEMENT } (\text{exec } v_7)) \wedge \\
& (\forall v_8. P \text{ RECON } (\text{exec } v_8)) \wedge (\forall v_9. P \text{ COMPLETE } (\text{exec } v_9)) \Rightarrow \\
& \forall v \ v_1. P \ v \ v_1
\end{aligned}$$

[PlatoonLeader_notWARNO_notreport1_exec_plCommand_justified_lemma]

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

[PlatoonLeader_notWARNO_notreport1_exec_plCommand_justified_thm]

$$\begin{aligned}
& \vdash s \neq \text{WARNO} \Rightarrow \\
& \text{plCommand} \neq \text{invalidPlCommand} \Rightarrow
\end{aligned}$$

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

[PlatoonLeader_notWARNO_notreport1_exec_plCommand_lemma]

$\vdash s \neq WARN0 \Rightarrow$
 $plCommand \neq invalidPlCommand \Rightarrow$
 $plCommand \neq report1 \Rightarrow$
 $\forall M \text{ Oi } Os.$
 $CFGInterpret (M, Oi, Os)$
 $(CFG \text{ inputOK } secContext \text{ secContextNull}$
 $([Name \text{ PlatoonLeader says}$
 $\text{prop } (SOME (SLc (PL \text{ plCommand}))) :: ins) \text{ s outs}) \Rightarrow$
 $(M, Oi, Os) \text{ satList}$
 propCommandList
 $[Name \text{ PlatoonLeader says}$
 $\text{prop } (SOME (SLc (PL \text{ plCommand})))]$

[PlatoonLeader_psgCommand_notDiscard_thm]

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

[PlatoonLeader_trap_psgCommand_justified_lemma]

$\vdash \forall NS \text{ Out } M \text{ Oi } Os.$
 $TR (M, Oi, Os)$
 $(\text{trap}$
 inputList

```

      [Name PlatoonLeader says
        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) WARNO outs)
(CFG inputOK secContext secContextNull ins
  (NS WARNO
    (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 WARNO
      (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, 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) WARNO outs)  $\wedge$ 
(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)))]

```

[PlatoonLeader_WARNO_exec_report1_justified_thm]

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

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))  $\iff$ 
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  $(M, Oi, Os)$ 
  (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]

```

 $\vdash \forall NS \text{ Out } M \text{ Oi } Os.$ 
  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)  $\Rightarrow$ 
(M, Oi, Os) sat prop NONE

```

[PlatoonSergeant_trap_plCommand_lemma]

```

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

```

17 PlanPBType Theory

Built: 10 June 2018

Parent Theories: indexedLists, patternMatches

17.1 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
 | Opoid | 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

17.2 Theorems

[plCommand_distinct_clauses]

\vdash receiveMission \neq warno \wedge receiveMission \neq tentativePlan \wedge
 receiveMission \neq recon \wedge receiveMission \neq report1 \wedge
 receiveMission \neq completePlan \wedge receiveMission \neq opoid \wedge
 receiveMission \neq supervise \wedge receiveMission \neq report2 \wedge
 receiveMission \neq complete \wedge receiveMission \neq plIncomplete \wedge
 receiveMission \neq invalidPlCommand \wedge warno \neq tentativePlan \wedge
 warno \neq recon \wedge warno \neq report1 \wedge warno \neq completePlan \wedge
 warno \neq opoid \wedge warno \neq supervise \wedge warno \neq report2 \wedge
 warno \neq complete \wedge warno \neq plIncomplete \wedge
 warno \neq invalidPlCommand \wedge tentativePlan \neq recon \wedge
 tentativePlan \neq report1 \wedge tentativePlan \neq completePlan \wedge
 tentativePlan \neq opoid \wedge tentativePlan \neq supervise \wedge
 tentativePlan \neq report2 \wedge tentativePlan \neq complete \wedge
 tentativePlan \neq plIncomplete \wedge
 tentativePlan \neq invalidPlCommand \wedge recon \neq report1 \wedge
 recon \neq completePlan \wedge recon \neq opoid \wedge recon \neq supervise \wedge
 recon \neq report2 \wedge recon \neq complete \wedge recon \neq plIncomplete \wedge
 recon \neq invalidPlCommand \wedge report1 \neq completePlan \wedge
 report1 \neq opoid \wedge report1 \neq supervise \wedge report1 \neq report2 \wedge
 report1 \neq complete \wedge report1 \neq plIncomplete \wedge
 report1 \neq invalidPlCommand \wedge completePlan \neq opoid \wedge
 completePlan \neq supervise \wedge completePlan \neq report2 \wedge
 completePlan \neq complete \wedge completePlan \neq plIncomplete \wedge
 completePlan \neq invalidPlCommand \wedge opoid \neq supervise \wedge

$$\begin{aligned} & \text{opoid} \neq \text{report2} \wedge \text{opoid} \neq \text{complete} \wedge \text{opoid} \neq \text{plIncomplete} \wedge \\ & \text{opoid} \neq \text{invalidPlCommand} \wedge \text{supervise} \neq \text{report2} \wedge \\ & \text{supervise} \neq \text{complete} \wedge \text{supervise} \neq \text{plIncomplete} \wedge \\ & \text{supervise} \neq \text{invalidPlCommand} \wedge \text{report2} \neq \text{complete} \wedge \\ & \text{report2} \neq \text{plIncomplete} \wedge \text{report2} \neq \text{invalidPlCommand} \wedge \\ & \text{complete} \neq \text{plIncomplete} \wedge \text{complete} \neq \text{invalidPlCommand} \wedge \\ & \text{plIncomplete} \neq \text{invalidPlCommand} \end{aligned}$$

[psgCommand_distinct_clauses]

$$\begin{aligned} & \vdash \text{initiateMovement} \neq \text{psgIncomplete} \wedge \\ & \text{initiateMovement} \neq \text{invalidPsgCommand} \wedge \\ & \text{psgIncomplete} \neq \text{invalidPsgCommand} \end{aligned}$$

[slCommand_distinct_clauses]

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

[slCommand_one_one]

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

[slOutput_distinct_clauses]

$$\begin{aligned} & \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 \end{aligned}$$

InitiateMovement \neq unAuthenticated \wedge
 InitiateMovement \neq unAuthorized \wedge Recon \neq Report1 \wedge
 Recon \neq CompletePlan \wedge Recon \neq Opoid \wedge Recon \neq Supervise \wedge
 Recon \neq Report2 \wedge Recon \neq Complete \wedge
 Recon \neq unAuthenticated \wedge Recon \neq unAuthorized \wedge
 Report1 \neq CompletePlan \wedge Report1 \neq Opoid \wedge
 Report1 \neq Supervise \wedge Report1 \neq Report2 \wedge
 Report1 \neq Complete \wedge Report1 \neq unAuthenticated \wedge
 Report1 \neq unAuthorized \wedge CompletePlan \neq Opoid \wedge
 CompletePlan \neq Supervise \wedge CompletePlan \neq Report2 \wedge
 CompletePlan \neq Complete \wedge CompletePlan \neq unAuthenticated \wedge
 CompletePlan \neq unAuthorized \wedge Opoid \neq Supervise \wedge
 Opoid \neq Report2 \wedge Opoid \neq Complete \wedge
 Opoid \neq unAuthenticated \wedge Opoid \neq unAuthorized \wedge
 Supervise \neq Report2 \wedge Supervise \neq Complete \wedge
 Supervise \neq unAuthenticated \wedge Supervise \neq unAuthorized \wedge
 Report2 \neq Complete \wedge Report2 \neq unAuthenticated \wedge
 Report2 \neq unAuthorized \wedge Complete \neq unAuthenticated \wedge
 Complete \neq unAuthorized \wedge unAuthenticated \neq unAuthorized

[slRole_distinct_clauses]

\vdash PlatoonLeader \neq PlatoonSergeant

[slState_distinct_clauses]

\vdash PLAN_PB \neq RECEIVE_MISSION \wedge PLAN_PB \neq WARNO \wedge
 PLAN_PB \neq TENTATIVE_PLAN \wedge PLAN_PB \neq INITIATE_MOVEMENT \wedge
 PLAN_PB \neq RECON \wedge PLAN_PB \neq REPORT1 \wedge
 PLAN_PB \neq COMPLETE_PLAN \wedge PLAN_PB \neq OPOID \wedge
 PLAN_PB \neq SUPERVISE \wedge PLAN_PB \neq REPORT2 \wedge
 PLAN_PB \neq COMPLETE \wedge RECEIVE_MISSION \neq WARNO \wedge
 RECEIVE_MISSION \neq TENTATIVE_PLAN \wedge
 RECEIVE_MISSION \neq INITIATE_MOVEMENT \wedge
 RECEIVE_MISSION \neq RECON \wedge RECEIVE_MISSION \neq REPORT1 \wedge
 RECEIVE_MISSION \neq COMPLETE_PLAN \wedge RECEIVE_MISSION \neq OPOID \wedge
 RECEIVE_MISSION \neq SUPERVISE \wedge RECEIVE_MISSION \neq REPORT2 \wedge
 RECEIVE_MISSION \neq COMPLETE \wedge WARNO \neq TENTATIVE_PLAN \wedge
 WARNO \neq INITIATE_MOVEMENT \wedge WARNO \neq RECON \wedge WARNO \neq REPORT1 \wedge
 WARNO \neq COMPLETE_PLAN \wedge WARNO \neq OPOID \wedge WARNO \neq SUPERVISE \wedge
 WARNO \neq REPORT2 \wedge WARNO \neq COMPLETE \wedge
 TENTATIVE_PLAN \neq INITIATE_MOVEMENT \wedge TENTATIVE_PLAN \neq RECON \wedge
 TENTATIVE_PLAN \neq REPORT1 \wedge TENTATIVE_PLAN \neq COMPLETE_PLAN \wedge
 TENTATIVE_PLAN \neq OPOID \wedge TENTATIVE_PLAN \neq SUPERVISE \wedge
 TENTATIVE_PLAN \neq REPORT2 \wedge TENTATIVE_PLAN \neq COMPLETE \wedge
 INITIATE_MOVEMENT \neq RECON \wedge INITIATE_MOVEMENT \neq REPORT1 \wedge
 INITIATE_MOVEMENT \neq COMPLETE_PLAN \wedge
 INITIATE_MOVEMENT \neq OPOID \wedge INITIATE_MOVEMENT \neq SUPERVISE \wedge
 INITIATE_MOVEMENT \neq REPORT2 \wedge INITIATE_MOVEMENT \neq COMPLETE \wedge
 RECON \neq REPORT1 \wedge RECON \neq COMPLETE_PLAN \wedge RECON \neq OPOID \wedge
 RECON \neq SUPERVISE \wedge RECON \neq REPORT2 \wedge RECON \neq COMPLETE \wedge

```

REPORT1 ≠ COMPLETE_PLAN ∧ REPORT1 ≠ OPOID ∧
REPORT1 ≠ SUPERVISE ∧ REPORT1 ≠ REPORT2 ∧
REPORT1 ≠ COMPLETE ∧ COMPLETE_PLAN ≠ OPOID ∧
COMPLETE_PLAN ≠ SUPERVISE ∧ COMPLETE_PLAN ≠ REPORT2 ∧
COMPLETE_PLAN ≠ COMPLETE ∧ OPOID ≠ SUPERVISE ∧
OPOID ≠ REPORT2 ∧ OPOID ≠ COMPLETE ∧ SUPERVISE ≠ REPORT2 ∧
SUPERVISE ≠ COMPLETE ∧ REPORT2 ≠ COMPLETE

```

18 PlanPBDef Theory

Built: 10 June 2018

Parent Theories: PlanPBType, acelfoundation, OMNIType

18.1 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 getPlCom x = invalidPlCommand then [prop NONE]
  else [PL_notWARNO_Auth (getPlCom x)]

```

[secContextNull_def]

```

⊢ ∀ x. secContextNull x = [TT]

```

18.2 Theorems

[getInitMove_def]

```

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

```

```

    getInitMove xs) ∧
  (∀ xs.
    getInitMove
      (Name PlatoonSergeant says
        prop (SOME (SLc (PSG invalidPsgCommand))))::xs) =
    getInitMove xs) ∧
  (∀ xs v68 v136 v135.
    getInitMove (v135 meet v136 says prop v68::xs) =
    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))::

```

$$\begin{aligned}
& xs)) \wedge \\
(\forall v150 \ xs. & \\
& P \ xs \Rightarrow \\
& P \\
& \quad (\text{Name PlatoonSergeant says} \\
& \quad \text{prop (SOME (SLc (PL v150)))::xs)) \wedge \\
(\forall xs. & \\
& P \ xs \Rightarrow \\
& P \\
& \quad (\text{Name PlatoonSergeant says} \\
& \quad \text{prop (SOME (SLc (PSG psgIncomplete)))::xs)) \wedge \\
(\forall xs. & \\
& P \ xs \Rightarrow \\
& P \\
& \quad (\text{Name PlatoonSergeant says} \\
& \quad \text{prop (SOME (SLc (PSG invalidPsgCommand)))::xs)) \wedge \\
(\forall v135 \ v136 \ v68 \ xs. & \\
& P \ xs \Rightarrow P \ (v135 \text{ meet } v136 \text{ says prop } v68::xs)) \wedge \\
(\forall v137 \ v138 \ v68 \ xs. & \\
& P \ xs \Rightarrow P \ (v137 \text{ quoting } v138 \text{ says prop } v68::xs)) \wedge \\
(\forall v12 \ v69 \ xs. & P \ xs \Rightarrow P \ (v12 \text{ says notf } v69::xs)) \wedge \\
(\forall v12 \ v70 \ v71 \ xs. & P \ xs \Rightarrow P \ (v12 \text{ says (v70 andf v71)::xs)) \wedge \\
(\forall v12 \ v72 \ v73 \ xs. & P \ xs \Rightarrow P \ (v12 \text{ says (v72 orf v73)::xs)) \wedge \\
(\forall v12 \ v74 \ v75 \ xs. & P \ xs \Rightarrow P \ (v12 \text{ says (v74 impf v75)::xs)) \wedge \\
(\forall v12 \ v76 \ v77 \ xs. & P \ xs \Rightarrow P \ (v12 \text{ says (v76 eqf v77)::xs)) \wedge \\
(\forall v12 \ v78 \ v79 \ xs. & P \ xs \Rightarrow P \ (v12 \text{ says } v78 \text{ says } v79::xs)) \wedge \\
(\forall v12 \ v80 \ v81 \ xs. & \\
& P \ xs \Rightarrow P \ (v12 \text{ says } v80 \text{ speaks_for } v81::xs)) \wedge \\
(\forall v12 \ v82 \ v83 \ xs. & \\
& P \ xs \Rightarrow P \ (v12 \text{ says } v82 \text{ controls } v83::xs)) \wedge \\
(\forall v12 \ v84 \ v85 \ v86 \ xs. & \\
& P \ xs \Rightarrow P \ (v12 \text{ says reps } v84 \ v85 \ v86::xs)) \wedge \\
(\forall v12 \ v87 \ v88 \ xs. & P \ xs \Rightarrow P \ (v12 \text{ says } v87 \text{ domi } v88::xs)) \wedge \\
(\forall v12 \ v89 \ v90 \ xs. & P \ xs \Rightarrow P \ (v12 \text{ says } v89 \text{ eqi } v90::xs)) \wedge \\
(\forall v12 \ v91 \ v92 \ xs. & P \ xs \Rightarrow P \ (v12 \text{ says } v91 \text{ doms } v92::xs)) \wedge \\
(\forall v12 \ v93 \ v94 \ xs. & P \ xs \Rightarrow P \ (v12 \text{ says } v93 \text{ eqs } v94::xs)) \wedge \\
(\forall v12 \ v95 \ v96 \ xs. & P \ xs \Rightarrow P \ (v12 \text{ says } v95 \text{ eqn } v96::xs)) \wedge \\
(\forall v12 \ v97 \ v98 \ xs. & P \ xs \Rightarrow P \ (v12 \text{ says } v97 \text{ lte } v98::xs)) \wedge \\
(\forall v12 \ v99 \ v100 \ xs. & P \ xs \Rightarrow P \ (v12 \text{ says } v99 \text{ lt } v100::xs)) \wedge \\
(\forall v14 \ v15 \ xs. & P \ xs \Rightarrow P \ (v14 \text{ speaks_for } v15::xs)) \wedge \\
(\forall v16 \ v17 \ xs. & P \ xs \Rightarrow P \ (v16 \text{ controls } v17::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
\end{aligned}$$

$$\forall v. P \ v$$

[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) = \\ & \quad \quad \text{cmd}) \wedge (\forall xs. \text{getPlCom (TT::xs)} = \text{getPlCom xs}) \wedge \\ & (\forall xs. \text{getPlCom (FF::xs)} = \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_2. \text{getPlCom (prop } v_2 :: xs) = \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_3. \text{getPlCom (notf } v_3 :: xs) = \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_5 \ v_4. \text{getPlCom (v}_4 \text{ andf } v_5 :: xs) = \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_7 \ v_6. \text{getPlCom (v}_6 \text{ orf } v_7 :: xs) = \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_9 \ v_8. \text{getPlCom (v}_8 \text{ impf } v_9 :: xs) = \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_{11} \ v_{10}. \text{getPlCom (v}_{10} \text{ eqf } v_{11} :: xs) = \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_{12}. \text{getPlCom (v}_{12} \text{ says TT::xs)} = \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_{12}. \text{getPlCom (v}_{12} \text{ says FF::xs)} = \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_{134}. \\ & \quad \text{getPlCom (Name v}_{134} \text{ says prop NONE::xs)} = \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_{146}. \\ & \quad \text{getPlCom} \\ & \quad \quad (\text{Name PlatoonLeader says prop (SOME (ESCc v}_{146})) :: xs) = \\ & \quad \quad \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_{151}. \\ & \quad \text{getPlCom} \\ & \quad \quad (\text{Name PlatoonLeader says prop (SOME (SLc (PSG v}_{151}))}) :: \\ & \quad \quad \quad xs) = \\ & \quad \quad \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_{144}. \\ & \quad \text{getPlCom} \\ & \quad \quad (\text{Name PlatoonSergeant says prop (SOME v}_{144}) :: xs) = \\ & \quad \quad \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_{68} \ v_{136} \ v_{135}. \\ & \quad \text{getPlCom (v}_{135} \text{ meet v}_{136} \text{ says prop v}_{68} :: xs) = \\ & \quad \quad \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_{68} \ v_{138} \ v_{137}. \\ & \quad \text{getPlCom (v}_{137} \text{ quoting v}_{138} \text{ says prop v}_{68} :: xs) = \\ & \quad \quad \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_{69} \ v_{12}. \\ & \quad \text{getPlCom (v}_{12} \text{ says notf v}_{69} :: xs) = \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_{71} \ v_{70} \ v_{12}. \\ & \quad \text{getPlCom (v}_{12} \text{ says (v}_{70} \text{ andf v}_{71}) :: xs) = \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_{73} \ v_{72} \ v_{12}. \\ & \quad \text{getPlCom (v}_{12} \text{ says (v}_{72} \text{ orf v}_{73}) :: xs) = \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_{75} \ v_{74} \ v_{12}. \\ & \quad \text{getPlCom (v}_{12} \text{ says (v}_{74} \text{ impf v}_{75}) :: xs) = \text{getPlCom xs}) \wedge \\ & (\forall xs \ v_{77} \ v_{76} \ v_{12}. \\ & \quad \text{getPlCom (v}_{12} \text{ says (v}_{76} \text{ eqf v}_{77}) :: xs) = \text{getPlCom xs}) \wedge \end{aligned}$$

$$\begin{aligned}
& (\forall xs \ v_{79} \ v_{78} \ v_{12}. \\
& \quad \text{getPlCom } (v_{12} \text{ says } v_{78} \text{ says } v_{79}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{81} \ v_{80} \ v_{12}. \\
& \quad \text{getPlCom } (v_{12} \text{ says } v_{80} \text{ speaks_for } v_{81}::xs) = \\
& \quad \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{83} \ v_{82} \ v_{12}. \\
& \quad \text{getPlCom } (v_{12} \text{ says } v_{82} \text{ controls } v_{83}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{86} \ v_{85} \ v_{84} \ v_{12}. \\
& \quad \text{getPlCom } (v_{12} \text{ says } \text{reps } v_{84} \ v_{85} \ v_{86}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{88} \ v_{87} \ v_{12}. \\
& \quad \text{getPlCom } (v_{12} \text{ says } v_{87} \text{ domi } v_{88}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{90} \ v_{89} \ v_{12}. \\
& \quad \text{getPlCom } (v_{12} \text{ says } v_{89} \text{ eqi } v_{90}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{92} \ v_{91} \ v_{12}. \\
& \quad \text{getPlCom } (v_{12} \text{ says } v_{91} \text{ doms } v_{92}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{94} \ v_{93} \ v_{12}. \\
& \quad \text{getPlCom } (v_{12} \text{ says } v_{93} \text{ eqs } v_{94}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{96} \ v_{95} \ v_{12}. \\
& \quad \text{getPlCom } (v_{12} \text{ says } v_{95} \text{ eqn } v_{96}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{98} \ v_{97} \ v_{12}. \\
& \quad \text{getPlCom } (v_{12} \text{ says } v_{97} \text{ lte } v_{98}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{99} \ v_{12} \ v_{100}. \\
& \quad \text{getPlCom } (v_{12} \text{ says } v_{99} \text{ lt } v_{100}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{15} \ v_{14}. \\
& \quad \text{getPlCom } (v_{14} \text{ speaks_for } v_{15}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{17} \ v_{16}. \\
& \quad \text{getPlCom } (v_{16} \text{ controls } v_{17}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{20} \ v_{19} \ v_{18}. \\
& \quad \text{getPlCom } (\text{reps } v_{18} \ v_{19} \ v_{20}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{22} \ v_{21}. \text{getPlCom } (v_{21} \text{ domi } v_{22}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{24} \ v_{23}. \text{getPlCom } (v_{23} \text{ eqi } v_{24}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{26} \ v_{25}. \text{getPlCom } (v_{25} \text{ doms } v_{26}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{28} \ v_{27}. \text{getPlCom } (v_{27} \text{ eqs } v_{28}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{30} \ v_{29}. \text{getPlCom } (v_{29} \text{ eqn } v_{30}::xs) = \text{getPlCom } xs) \wedge \\
& (\forall xs \ v_{32} \ v_{31}. \text{getPlCom } (v_{31} \text{ lte } v_{32}::xs) = \text{getPlCom } xs) \wedge \\
& \forall xs \ v_{34} \ v_{33}. \text{getPlCom } (v_{33} \text{ lt } v_{34}::xs) = \text{getPlCom } xs
\end{aligned}$$

[getPlCom_ind]

$$\begin{aligned}
& \vdash \forall P. \\
& \quad P \ [] \wedge \\
& \quad (\forall cmd \ xs. \\
& \quad \quad P \\
& \quad \quad \quad (\text{Name PlatoonLeader says prop (SOME (SLc (PL cmd))))::} \\
& \quad \quad \quad \quad xs)) \wedge (\forall xs. P \ xs \Rightarrow P \ (\text{TT}::xs)) \wedge \\
& \quad (\forall xs. P \ xs \Rightarrow P \ (\text{FF}::xs)) \wedge \\
& \quad (\forall v_2 \ xs. P \ xs \Rightarrow P \ (\text{prop } v_2::xs)) \wedge \\
& \quad (\forall v_3 \ xs. P \ xs \Rightarrow P \ (\text{notf } v_3::xs)) \wedge \\
& \quad (\forall v_4 \ v_5 \ xs. P \ xs \Rightarrow P \ (v_4 \text{ andf } v_5::xs)) \wedge \\
& \quad (\forall v_6 \ v_7 \ xs. P \ xs \Rightarrow P \ (v_6 \text{ orf } v_7::xs)) \wedge
\end{aligned}$$

$$\begin{aligned}
& (\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 v_{151} xs. \\
& \quad P xs \Rightarrow \\
& \quad P \\
& \quad (\text{Name PlatoonLeader says} \\
& \quad \text{prop (SOME (SLc (PSG } v_{151})) :: xs)) \wedge \\
& (\forall v_{144} xs. \\
& \quad P xs \Rightarrow \\
& \quad P (\text{Name PlatoonSergeant says prop (SOME } v_{144}) :: xs)) \wedge \\
& (\forall v_{135} v_{136} v_{68} xs. \\
& \quad P xs \Rightarrow P (v_{135} \text{ meet } v_{136} \text{ says prop } v_{68} :: xs)) \wedge \\
& (\forall v_{137} v_{138} v_{68} xs. \\
& \quad P xs \Rightarrow P (v_{137} \text{ quoting } v_{138} \text{ says prop } v_{68} :: xs)) \wedge \\
& (\forall v_{12} v_{69} xs. P xs \Rightarrow P (v_{12} \text{ says 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. \\
& \quad 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. \\
& \quad 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. \\
& \quad P xs \Rightarrow P (v_{12} \text{ says 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 \\
& (\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
\end{aligned}$$

$$\begin{aligned}
& (\forall v_{31} v_{32} xs. P xs \Rightarrow P (v_{31} \text{ lte } v_{32} :: xs)) \wedge \\
& (\forall v_{33} v_{34} xs. P xs \Rightarrow P (v_{33} \text{ lt } v_{34} :: xs)) \Rightarrow \\
& \forall v. P v
\end{aligned}$$

[getPsgCom_def]

$$\begin{aligned}
& \vdash (\text{getPsgCom } [] = \text{invalidPsgCommand}) \wedge \\
& (\forall xs \text{ cmd}. \\
& \quad \text{getPsgCom} \\
& \quad \quad (\text{Name PlatoonSergeant says prop (SOME (SLc (PSG cmd)))) ::} \\
& \quad \quad \quad xs) = \\
& \quad \text{cmd}) \wedge (\forall xs. \text{getPsgCom (TT :: xs)} = \text{getPsgCom xs}) \wedge \\
& (\forall xs. \text{getPsgCom (FF :: xs)} = \text{getPsgCom xs}) \wedge \\
& (\forall xs v_2. \text{getPsgCom (prop } v_2 :: xs) = \text{getPsgCom xs}) \wedge \\
& (\forall xs v_3. \text{getPsgCom (notf } v_3 :: xs) = \text{getPsgCom xs}) \wedge \\
& (\forall xs v_5 v_4. \text{getPsgCom (v}_4 \text{ andf v}_5 :: xs) = \text{getPsgCom xs}) \wedge \\
& (\forall xs v_7 v_6. \text{getPsgCom (v}_6 \text{ orf v}_7 :: xs) = \text{getPsgCom xs}) \wedge \\
& (\forall xs v_9 v_8. \text{getPsgCom (v}_8 \text{ impf v}_9 :: xs) = \text{getPsgCom xs}) \wedge \\
& (\forall xs v_{11} v_{10}. \text{getPsgCom (v}_{10} \text{ eqf v}_{11} :: xs) = \text{getPsgCom xs}) \wedge \\
& (\forall xs v_{12}. \text{getPsgCom (v}_{12} \text{ says TT :: xs)} = \text{getPsgCom xs}) \wedge \\
& (\forall xs v_{12}. \text{getPsgCom (v}_{12} \text{ says FF :: xs)} = \text{getPsgCom xs}) \wedge \\
& (\forall xs v_{134}. \\
& \quad \text{getPsgCom (Name v}_{134} \text{ says prop NONE :: xs)} = \text{getPsgCom xs}) \wedge \\
& (\forall xs v_{144}. \\
& \quad \text{getPsgCom (Name PlatoonLeader says prop (SOME v}_{144}) :: xs) =} \\
& \quad \text{getPsgCom xs}) \wedge \\
& (\forall xs v_{146}. \\
& \quad \text{getPsgCom} \\
& \quad \quad (\text{Name PlatoonSergeant says prop (SOME (ESCc v}_{146}) ::} \\
& \quad \quad \quad xs) = \\
& \quad \text{getPsgCom xs}) \wedge \\
& (\forall xs v_{150}. \\
& \quad \text{getPsgCom} \\
& \quad \quad (\text{Name PlatoonSergeant says prop (SOME (SLc (PL v}_{150}) ::} \\
& \quad \quad \quad xs) = \\
& \quad \text{getPsgCom xs}) \wedge \\
& (\forall xs v_{68} v_{136} v_{135}. \\
& \quad \text{getPsgCom (v}_{135} \text{ meet v}_{136} \text{ says prop v}_{68} :: xs) =} \\
& \quad \text{getPsgCom xs}) \wedge \\
& (\forall xs v_{68} v_{138} v_{137}. \\
& \quad \text{getPsgCom (v}_{137} \text{ quoting v}_{138} \text{ says prop v}_{68} :: xs) =} \\
& \quad \text{getPsgCom xs}) \wedge \\
& (\forall xs v_{69} v_{12}. \\
& \quad \text{getPsgCom (v}_{12} \text{ says notf v}_{69} :: xs) = \text{getPsgCom xs}) \wedge \\
& (\forall xs v_{71} v_{70} v_{12}. \\
& \quad \text{getPsgCom (v}_{12} \text{ says (v}_{70} \text{ andf v}_{71}) :: xs) = \text{getPsgCom xs}) \wedge \\
& (\forall xs v_{73} v_{72} v_{12}. \\
& \quad \text{getPsgCom (v}_{12} \text{ says (v}_{72} \text{ orf v}_{73}) :: xs) = \text{getPsgCom xs}) \wedge \\
& (\forall xs v_{75} v_{74} v_{12}. \\
& \quad \text{getPsgCom (v}_{12} \text{ says (v}_{74} \text{ impf v}_{75}) :: xs) = \text{getPsgCom xs}) \wedge
\end{aligned}$$

$$\begin{aligned}
& (\forall xs \ v_{77} \ v_{76} \ v_{12}. \\
& \quad \text{getPsgCom } (v_{12} \text{ says } (v_{76} \text{ eqf } v_{77})::xs) = \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{79} \ v_{78} \ v_{12}. \\
& \quad \text{getPsgCom } (v_{12} \text{ says } v_{78} \text{ says } v_{79}::xs) = \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{81} \ v_{80} \ v_{12}. \\
& \quad \text{getPsgCom } (v_{12} \text{ says } v_{80} \text{ speaks_for } v_{81}::xs) = \\
& \quad \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{83} \ v_{82} \ v_{12}. \\
& \quad \text{getPsgCom } (v_{12} \text{ says } v_{82} \text{ controls } v_{83}::xs) = \\
& \quad \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{86} \ v_{85} \ v_{84} \ v_{12}. \\
& \quad \text{getPsgCom } (v_{12} \text{ says reps } v_{84} \ v_{85} \ v_{86}::xs) = \\
& \quad \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{88} \ v_{87} \ v_{12}. \\
& \quad \text{getPsgCom } (v_{12} \text{ says } v_{87} \text{ domi } v_{88}::xs) = \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{90} \ v_{89} \ v_{12}. \\
& \quad \text{getPsgCom } (v_{12} \text{ says } v_{89} \text{ eqi } v_{90}::xs) = \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{92} \ v_{91} \ v_{12}. \\
& \quad \text{getPsgCom } (v_{12} \text{ says } v_{91} \text{ doms } v_{92}::xs) = \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{94} \ v_{93} \ v_{12}. \\
& \quad \text{getPsgCom } (v_{12} \text{ says } v_{93} \text{ eqs } v_{94}::xs) = \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{96} \ v_{95} \ v_{12}. \\
& \quad \text{getPsgCom } (v_{12} \text{ says } v_{95} \text{ eqn } v_{96}::xs) = \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{98} \ v_{97} \ v_{12}. \\
& \quad \text{getPsgCom } (v_{12} \text{ says } v_{97} \text{ lte } v_{98}::xs) = \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{99} \ v_{12} \ v_{100}. \\
& \quad \text{getPsgCom } (v_{12} \text{ says } v_{99} \text{ lt } v_{100}::xs) = \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{15} \ v_{14}. \\
& \quad \text{getPsgCom } (v_{14} \text{ speaks_for } v_{15}::xs) = \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{17} \ v_{16}. \\
& \quad \text{getPsgCom } (v_{16} \text{ controls } v_{17}::xs) = \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{20} \ v_{19} \ v_{18}. \\
& \quad \text{getPsgCom } (\text{reps } v_{18} \ v_{19} \ v_{20}::xs) = \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{22} \ v_{21}. \text{getPsgCom } (v_{21} \text{ domi } v_{22}::xs) = \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{24} \ v_{23}. \text{getPsgCom } (v_{23} \text{ eqi } v_{24}::xs) = \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{26} \ v_{25}. \text{getPsgCom } (v_{25} \text{ doms } v_{26}::xs) = \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{28} \ v_{27}. \text{getPsgCom } (v_{27} \text{ eqs } v_{28}::xs) = \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{30} \ v_{29}. \text{getPsgCom } (v_{29} \text{ eqn } v_{30}::xs) = \text{getPsgCom } xs) \wedge \\
& (\forall xs \ v_{32} \ v_{31}. \text{getPsgCom } (v_{31} \text{ lte } v_{32}::xs) = \text{getPsgCom } xs) \wedge \\
& \forall xs \ v_{34} \ v_{33}. \text{getPsgCom } (v_{33} \text{ lt } v_{34}::xs) = \text{getPsgCom } xs
\end{aligned}$$

[getPsgCom_ind]

$$\begin{aligned}
& \vdash \forall P. \\
& \quad P \ [] \wedge \\
& \quad (\forall cmd \ xs. \\
& \quad \quad P \\
& \quad \quad (\text{Name PlatoonSergeant says} \\
& \quad \quad \text{prop (SOME (SLc (PSG cmd)))::xs})) \wedge \\
& \quad (\forall xs. P \ xs \Rightarrow P \ (\text{TT}::xs)) \wedge (\forall xs. P \ xs \Rightarrow P \ (\text{FF}::xs)) \wedge
\end{aligned}$$

$$\begin{aligned}
& (\forall v_2 \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (\text{prop } v_2::xs)) \wedge \\
& (\forall v_3 \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (\text{notf } v_3::xs)) \wedge \\
& (\forall v_4 \text{ } v_5 \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_4 \text{ andf } v_5::xs)) \wedge \\
& (\forall v_6 \text{ } v_7 \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_6 \text{ orf } v_7::xs)) \wedge \\
& (\forall v_8 \text{ } v_9 \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_8 \text{ impf } v_9::xs)) \wedge \\
& (\forall v_{10} \text{ } v_{11} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{10} \text{ eqf } v_{11}::xs)) \wedge \\
& (\forall v_{12} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says TT}::xs)) \wedge \\
& (\forall v_{12} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says FF}::xs)) \wedge \\
& (\forall v_{134} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (\text{Name } v_{134} \text{ says prop NONE}::xs)) \wedge \\
& (\forall v_{144} \text{ } xs. \\
& \quad P \text{ } xs \Rightarrow \\
& \quad P \text{ } (\text{Name PlatoonLeader says prop (SOME } v_{144})::xs)) \wedge \\
& (\forall v_{146} \text{ } xs. \\
& \quad P \text{ } xs \Rightarrow \\
& \quad P \\
& \quad (\text{Name PlatoonSergeant says prop (SOME (ESCc } v_{146})):: \\
& \quad \quad xs)) \wedge \\
& (\forall v_{150} \text{ } xs. \\
& \quad P \text{ } xs \Rightarrow \\
& \quad P \\
& \quad (\text{Name PlatoonSergeant says } \\
& \quad \quad \text{prop (SOME (SLc (PL } v_{150}))::xs)) \wedge \\
& (\forall v_{135} \text{ } v_{136} \text{ } v_{68} \text{ } xs. \\
& \quad P \text{ } xs \Rightarrow P \text{ } (v_{135} \text{ meet } v_{136} \text{ says prop } v_{68}::xs)) \wedge \\
& (\forall v_{137} \text{ } v_{138} \text{ } v_{68} \text{ } xs. \\
& \quad P \text{ } xs \Rightarrow P \text{ } (v_{137} \text{ quoting } v_{138} \text{ says prop } v_{68}::xs)) \wedge \\
& (\forall v_{12} \text{ } v_{69} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says notf } v_{69}::xs)) \wedge \\
& (\forall v_{12} \text{ } v_{70} \text{ } 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} \text{ } v_{72} \text{ } 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} \text{ } v_{74} \text{ } 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} \text{ } v_{76} \text{ } 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} \text{ } v_{78} \text{ } 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} \text{ } v_{80} \text{ } 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} \text{ } v_{82} \text{ } 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} \text{ } v_{84} \text{ } v_{85} \text{ } v_{86} \text{ } xs. \\
& \quad P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says reps } v_{84} \text{ } v_{85} \text{ } v_{86}::xs)) \wedge \\
& (\forall v_{12} \text{ } v_{87} \text{ } 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} \text{ } v_{89} \text{ } 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} \text{ } v_{91} \text{ } 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} \text{ } v_{93} \text{ } 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} \text{ } v_{95} \text{ } 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} \text{ } v_{97} \text{ } 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} \text{ } v_{99} \text{ } v_{100} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{99} \text{ lt } v_{100}::xs)) \wedge \\
& (\forall v_{14} \text{ } v_{15} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{14} \text{ speaks_for } v_{15}::xs)) \wedge \\
& (\forall v_{16} \text{ } v_{17} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{16} \text{ controls } v_{17}::xs)) \wedge \\
& (\forall v_{18} \text{ } v_{19} \text{ } v_{20} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (\text{reps } v_{18} \text{ } v_{19} \text{ } v_{20}::xs)) \wedge \\
& (\forall v_{21} \text{ } v_{22} \text{ } xs. P \text{ } xs \Rightarrow P \text{ } (v_{21} \text{ domi } v_{22}::xs)) \wedge
\end{aligned}$$

$$\begin{aligned}
& (\forall v_{23} v_{24} xs. P xs \Rightarrow P (v_{23} \text{ eqi } v_{24} :: xs)) \wedge \\
& (\forall v_{25} v_{26} xs. P xs \Rightarrow P (v_{25} \text{ doms } v_{26} :: xs)) \wedge \\
& (\forall v_{27} v_{28} xs. P xs \Rightarrow P (v_{27} \text{ eqs } v_{28} :: xs)) \wedge \\
& (\forall v_{29} v_{30} xs. P xs \Rightarrow P (v_{29} \text{ eqn } v_{30} :: xs)) \wedge \\
& (\forall v_{31} v_{32} xs. P xs \Rightarrow P (v_{31} \text{ lte } v_{32} :: xs)) \wedge \\
& (\forall v_{33} v_{34} xs. P xs \Rightarrow P (v_{33} \text{ lt } v_{34} :: xs)) \Rightarrow \\
& \forall v. P v
\end{aligned}$$

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

```

    prop (SOME (SLc (PL report1)))::xs) =
  getRecon xs) ∧
(∀ xs.
  getRecon
    (Name PlatoonLeader says
      prop (SOME (SLc (PL completePlan)))::xs) =
  getRecon xs) ∧
(∀ xs.
  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) ∧

```

$(\forall xs \ v_{68} \ v_{138} \ v_{137}.$
 $\quad \text{getRecon } (v_{137} \text{ quoting } v_{138} \text{ says prop } v_{68}::xs) =$
 $\quad \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{69} \ v_{12}.$
 $\quad \text{getRecon } (v_{12} \text{ says notf } v_{69}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{71} \ v_{70} \ v_{12}.$
 $\quad \text{getRecon } (v_{12} \text{ says } (v_{70} \text{ andf } v_{71})::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{73} \ v_{72} \ v_{12}.$
 $\quad \text{getRecon } (v_{12} \text{ says } (v_{72} \text{ orf } v_{73})::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{75} \ v_{74} \ v_{12}.$
 $\quad \text{getRecon } (v_{12} \text{ says } (v_{74} \text{ impf } v_{75})::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{77} \ v_{76} \ v_{12}.$
 $\quad \text{getRecon } (v_{12} \text{ says } (v_{76} \text{ eqf } v_{77})::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{79} \ v_{78} \ v_{12}.$
 $\quad \text{getRecon } (v_{12} \text{ says } v_{78} \text{ says } v_{79}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{81} \ v_{80} \ v_{12}.$
 $\quad \text{getRecon } (v_{12} \text{ says } v_{80} \text{ speaks_for } v_{81}::xs) =$
 $\quad \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{83} \ v_{82} \ v_{12}.$
 $\quad \text{getRecon } (v_{12} \text{ says } v_{82} \text{ controls } v_{83}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{86} \ v_{85} \ v_{84} \ v_{12}.$
 $\quad \text{getRecon } (v_{12} \text{ says reps } v_{84} \ v_{85} \ v_{86}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{88} \ v_{87} \ v_{12}.$
 $\quad \text{getRecon } (v_{12} \text{ says } v_{87} \text{ domi } v_{88}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{90} \ v_{89} \ v_{12}.$
 $\quad \text{getRecon } (v_{12} \text{ says } v_{89} \text{ eqi } v_{90}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{92} \ v_{91} \ v_{12}.$
 $\quad \text{getRecon } (v_{12} \text{ says } v_{91} \text{ doms } v_{92}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{94} \ v_{93} \ v_{12}.$
 $\quad \text{getRecon } (v_{12} \text{ says } v_{93} \text{ eqs } v_{94}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{96} \ v_{95} \ v_{12}.$
 $\quad \text{getRecon } (v_{12} \text{ says } v_{95} \text{ eqn } v_{96}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{98} \ v_{97} \ v_{12}.$
 $\quad \text{getRecon } (v_{12} \text{ says } v_{97} \text{ lte } v_{98}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{99} \ v_{12} \ v_{100}.$
 $\quad \text{getRecon } (v_{12} \text{ says } v_{99} \text{ lt } v_{100}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{15} \ v_{14}.$
 $\quad \text{getRecon } (v_{14} \text{ speaks_for } v_{15}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{17} \ v_{16}.$
 $\quad \text{getRecon } (v_{16} \text{ controls } v_{17}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{20} \ v_{19} \ v_{18}.$
 $\quad \text{getRecon } (\text{reps } v_{18} \ v_{19} \ v_{20}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{22} \ v_{21}.$
 $\quad \text{getRecon } (v_{21} \text{ domi } v_{22}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{24} \ v_{23}.$
 $\quad \text{getRecon } (v_{23} \text{ eqi } v_{24}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{26} \ v_{25}.$
 $\quad \text{getRecon } (v_{25} \text{ doms } v_{26}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{28} \ v_{27}.$
 $\quad \text{getRecon } (v_{27} \text{ eqs } v_{28}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{30} \ v_{29}.$
 $\quad \text{getRecon } (v_{29} \text{ eqn } v_{30}::xs) = \text{getRecon } xs) \wedge$
 $(\forall xs \ v_{32} \ v_{31}.$
 $\quad \text{getRecon } (v_{31} \text{ lte } v_{32}::xs) = \text{getRecon } xs) \wedge$
 $\forall xs \ v_{34} \ v_{33}.$
 $\quad \text{getRecon } (v_{33} \text{ lt } v_{34}::xs) = \text{getRecon } xs$

[getRecon_ind]

$$\begin{aligned}
& \vdash \forall P. \\
& \quad P \square \wedge \\
& \quad (\forall xs. \\
& \quad \quad P \\
& \quad \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \quad \text{prop (SOME (SLc (PL recon)))::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 \quad (\text{Name PlatoonLeader says prop (SOME (ESCc } v_{146})):: \\
& \quad \quad \quad \quad \quad xs)) \wedge \\
& \quad (\forall xs. \\
& \quad \quad P \ xs \Rightarrow \\
& \quad \quad P \\
& \quad \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \quad \text{prop (SOME (SLc (PL receiveMission)))::xs}) \wedge \\
& \quad (\forall xs. \\
& \quad \quad P \ xs \Rightarrow \\
& \quad \quad P \\
& \quad \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \quad \text{prop (SOME (SLc (PL warno)))::xs}) \wedge \\
& \quad (\forall xs. \\
& \quad \quad P \ xs \Rightarrow \\
& \quad \quad P \\
& \quad \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \quad \text{prop (SOME (SLc (PL tentativePlan)))::xs}) \wedge \\
& \quad (\forall xs. \\
& \quad \quad P \ xs \Rightarrow \\
& \quad \quad P \\
& \quad \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \quad \text{prop (SOME (SLc (PL report1)))::xs}) \wedge \\
& \quad (\forall xs. \\
& \quad \quad P \ xs \Rightarrow \\
& \quad \quad P \\
& \quad \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \quad \quad \text{prop (SOME (SLc (PL completePlan)))::xs}) \wedge \\
& \quad (\forall xs. \\
& \quad \quad P \ xs \Rightarrow
\end{aligned}$$

```

P
(Name PlatoonLeader says
  prop (SOME (SLc (PL opoid))))::xs)) ∧
(∀ xs.
  P xs ⇒
  P
    (Name PlatoonLeader says
      prop (SOME (SLc (PL supervise))))::xs)) ∧
(∀ xs.
  P xs ⇒
  P
    (Name PlatoonLeader says
      prop (SOME (SLc (PL report2))))::xs)) ∧
(∀ xs.
  P xs ⇒
  P
    (Name PlatoonLeader says
      prop (SOME (SLc (PL complete))))::xs)) ∧
(∀ xs.
  P xs ⇒
  P
    (Name PlatoonLeader says
      prop (SOME (SLc (PL plIncomplete))))::xs)) ∧
(∀ xs.
  P xs ⇒
  P
    (Name PlatoonLeader says
      prop (SOME (SLc (PL invalidPlCommand))))::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.

```

$$\begin{aligned}
& 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} \ v_{100} \text{ } xs. \ P \text{ } xs \Rightarrow P \text{ } (v_{12} \text{ says } v_{99} \text{ lt } v_{100}::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{ } (reps \ v_{18} \ 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 \ v
\end{aligned}$$

[getReport_def]

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

```

    getReport xs) ∧
  (∀ xs.
    getReport
      (Name PlatoonLeader says prop (SOME (SLc (PL warno))))::
        xs) =
    getReport xs) ∧
  (∀ xs.
    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) ∧
  (∀ xs v20 v19 v18.
    getReport (reps v18 v19 v20::xs) = getReport xs) ∧
  (∀ xs v22 v21. getReport (v21 domi v22::xs) = getReport xs) ∧
  (∀ xs v24 v23. getReport (v23 eqi v24::xs) = getReport xs) ∧
  (∀ xs v26 v25. getReport (v25 doms v26::xs) = getReport xs) ∧
  (∀ xs v28 v27. getReport (v27 eqs v28::xs) = getReport xs) ∧
  (∀ xs v30 v29. getReport (v29 eqn v30::xs) = getReport xs) ∧
  (∀ xs v32 v31. getReport (v31 lte v32::xs) = getReport xs) ∧
  ∀ xs v34 v33. getReport (v33 lt v34::xs) = getReport xs

```

[getReport_ind]

```

⊢ ∀ P.
  P [] ∧
  (∀ xs.
    P
      (Name PlatoonLeader says
        prop (SOME (SLc (PL report1)))::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)) ∧

```

$$\begin{aligned}
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \text{(Name PlatoonLeader says} \\
& \quad \text{prop (SOME (SLc (PL tentativePlan))))::xs))} \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \text{(Name PlatoonLeader says} \\
& \quad \text{prop (SOME (SLc (PL recon))))::xs))} \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \text{(Name PlatoonLeader says} \\
& \quad \text{prop (SOME (SLc (PL completePlan))))::xs))} \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \text{(Name PlatoonLeader says} \\
& \quad \text{prop (SOME (SLc (PL opoid))))::xs))} \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \text{(Name PlatoonLeader says} \\
& \quad \text{prop (SOME (SLc (PL supervise))))::xs))} \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \text{(Name PlatoonLeader says} \\
& \quad \text{prop (SOME (SLc (PL report2))))::xs))} \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \text{(Name PlatoonLeader says} \\
& \quad \text{prop (SOME (SLc (PL complete))))::xs))} \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \text{(Name PlatoonLeader says} \\
& \quad \text{prop (SOME (SLc (PL plIncomplete))))::xs))} \wedge \\
& (\forall v151 \ xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \text{(Name PlatoonLeader says} \\
& \quad \text{prop (SOME (SLc (PL invalidPlCommand))))::xs))} \wedge
\end{aligned}$$

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

[getTentativePlan_def]

$$\begin{aligned}
& \vdash (\text{getTentativePlan []} = [\text{NONE}]) \wedge \\
& (\forall \text{xs.} \\
& \quad \text{getTentativePlan} \\
& \quad (\text{Name PlatoonLeader says} \\
& \quad \text{prop (SOME (SLc (PL tentativePlan)))::xs}) = \\
& \quad [\text{SOME (SLc (PL tentativePlan))}] \wedge \\
& (\forall \text{xs. getTentativePlan (TT::xs)} = \text{getTentativePlan xs}) \wedge \\
& (\forall \text{xs. getTentativePlan (FF::xs)} = \text{getTentativePlan xs}) \wedge \\
& (\forall \text{xs } v_2.
\end{aligned}$$

```

    getTenativePlan (prop v2::xs) = getTenativePlan xs) ∧
(∀ xs v3.
    getTenativePlan (notf v3::xs) = getTenativePlan xs) ∧
(∀ xs v5 v4.
    getTenativePlan (v4 andf v5::xs) = getTenativePlan xs) ∧
(∀ xs v7 v6.
    getTenativePlan (v6 orf v7::xs) = getTenativePlan xs) ∧
(∀ 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.
  getTentativePlan (v12 says (v72 orf v73)::xs) =
  getTentativePlan xs) ∧
(∀ xs v75 v74 v12.
  getTentativePlan (v12 says (v74 impf v75)::xs) =
  getTentativePlan 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.

```

```

    getTenativePlan (reps v18 v19 v20::xs) =
    getTenativePlan xs) ∧
  (∀ xs v22 v21.
    getTenativePlan (v21 domi v22::xs) = getTenativePlan xs) ∧
  (∀ xs v24 v23.
    getTenativePlan (v23 eqi v24::xs) = getTenativePlan 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)) ∧

```

$$\begin{aligned}
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop (SOME (SLc (PL recon)))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop (SOME (SLc (PL report1)))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop (SOME (SLc (PL completePlan)))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop (SOME (SLc (PL opoid)))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop (SOME (SLc (PL supervise)))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop (SOME (SLc (PL report2)))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop (SOME (SLc (PL complete)))::xs}) \wedge \\
& (\forall xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop (SOME (SLc (PL plIncomplete)))::xs}) \wedge \\
& (\forall v151 \ xs. \\
& \quad P \ xs \Rightarrow \\
& \quad P \\
& \quad \quad (\text{Name PlatoonLeader says} \\
& \quad \quad \text{prop (SOME (SLc (PL invalidPlCommand)))::xs}) \wedge
\end{aligned}$$

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

Index

ConductORPType Theory, 39

Datatypes, 40

Theorems, 40

plCommand_distinct_clauses, 40

psgCommand_distinct_clauses, 40

slCommand_distinct_clauses, 40

slCommand_one_one, 40

slOutput_distinct_clauses, 40

slRole_distinct_clauses, 41

slState_distinct_clauses, 41

ConductPBType Theory, 46

Datatypes, 46

Theorems, 47

plCommandPB_distinct_clauses, 47

psgCommandPB_distinct_clauses, 47

slCommand_distinct_clauses, 47

slCommand_one_one, 47

slOutput_distinct_clauses, 47

slRole_distinct_clauses, 47

slState_distinct_clauses, 47

MoveToORPType Theory, 52

Datatypes, 52

Theorems, 52

slCommand_distinct_clauses, 52

slOutput_distinct_clauses, 53

slState_distinct_clauses, 53

MoveToPBType Theory, 57

Datatypes, 58

Theorems, 58

slCommand_distinct_clauses, 58

slOutput_distinct_clauses, 58

slState_distinct_clauses, 58

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

secContext_def, 23

secHelper_def, 24

Theorems, 24

getOmniCommand_def, 24

getOmniCommand_ind, 27

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, 72

Definitions, 72

PL_notWARNO_Auth_def, 72

PL_WARNO_Auth_def, 72

secContext_def, 72

secContextNull_def, 73

Theorems, 73

getInitMove_def, 73

getInitMove_ind, 75

getPlCom_def, 77

getPlCom_ind, 78

getPsgCom_def, 80

getPsgCom_ind, 81

getRecon_def, 83

getRecon_ind, 86

getReport_def, 88

- getReport_ind, 91
- getTenativePlan_def, 93
- getTenativePlan_ind, 97
- PlanPBType Theory**, 68
 - Datatypes, 69
 - Theorems, 69
 - plCommand_distinct_clauses, 69
 - psgCommand_distinct_clauses, 70
 - slCommand_distinct_clauses, 70
 - slCommand_one_one, 70
 - slOutput_distinct_clauses, 70
 - slRole_distinct_clauses, 71
 - slState_distinct_clauses, 71
- 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**, 34
 - Definitions, 34
 - secContextConductORP_def, 34
 - ssmConductORPStateInterp_def, 35
 - Theorems, 35
 - authTestConductORP_cmd_reject_lemma, 35
 - authTestConductORP_def, 35
 - authTestConductORP_ind, 36
 - conductORPNS_def, 36
 - conductORPNS_ind, 37
 - conductORPOut_def, 37
 - conductORPOut_ind, 38
 - PlatoonLeader_exec_plCommand_jus-

tified.thm, 38
 PlatoonLeader_plCommand.lemma, 39
 PlatoonSergeant_exec_psgCommand_-
 justified.thm, 39
 PlatoonSergeant_psgCommand.lemma,
 39
ssmConductPB Theory, 41
 Definitions, 41
 secContextConductPB_def, 41
 ssmConductPBStateInterp_def, 41
 Theorems, 41
 authTestConductPB_cmd_reject.lemma,
 41
 authTestConductPB_def, 41
 authTestConductPB_ind, 42
 conductPBNS_def, 43
 conductPBNS_ind, 43
 conductPBOut_def, 44
 conductPBOut_ind, 44
 PlatoonLeader_exec_plCommandPB_-
 justified.thm, 45
 PlatoonLeader_plCommandPB.lemma,
 45
 PlatoonSergeant_exec_psgCommandPB_-
 justified.thm, 45
 PlatoonSergeant_psgCommandPB.lemma,
 46
ssmMoveToORP Theory, 47
 Definitions, 48
 secContextMoveToORP_def, 48
 ssmMoveToORPStateInterp_def, 48
 Theorems, 48
 authTestMoveToORP_cmd_reject.lemma,
 48
 authTestMoveToORP_def, 48
 authTestMoveToORP_ind, 49
 moveToORPNS_def, 49
 moveToORPNS_ind, 50
 moveToORPOut_def, 50
 moveToORPOut_ind, 51
 PlatoonLeader_exec_slCommand_jus-
 tified.thm, 51
 PlatoonLeader_slCommand.lemma, 52
ssmMoveToPB Theory, 53
 Definitions, 53
 secContextMoveToPB_def, 53
 ssmMoveToPBStateInterp_def, 53
 Theorems, 53
 authTestMoveToPB_cmd_reject.lemma,
 53
 authTestMoveToPB_def, 53
 authTestMoveToPB_ind, 54
 moveToPBNS_def, 55
 moveToPBNS_ind, 55
 moveToPBOut_def, 56
 moveToPBOut_ind, 56
 PlatoonLeader_exec_slCommand_jus-
 tified.thm, 57
 PlatoonLeader_slCommand.lemma, 57
ssmPBIntegrated Theory, 28
 Theorems, 28
 inputOK_def, 28
 inputOK_ind, 29
 PBNS_def, 29
 PBNS_ind, 30
 PBOut_def, 31
 PBOut_ind, 31
 PlatoonLeader_Omni_notDiscard_slCom-
 mand.thm, 32
 PlatoonLeader_PLAN_PB_exec.lemma,
 32
 PlatoonLeader_PLAN_PB_trap_justi-
 fied.lemma, 32
 PlatoonLeader_PLAN_PB_trap_justi-
 fied.thm, 33
 PlatoonLeader_PLAN_PB_trap.lemma,
 34
ssmPlanPB Theory, 58
 Theorems, 59
 inputOK_def, 59
 inputOK_ind, 59
 planPBNS_def, 60
 planPBNS_ind, 61
 planPBOut_def, 61
 planPBOut_ind, 61
 PlatoonLeader_notWARNO_notreport1_-

exec_plCommand_justified_lemma, 62
 PlatoonLeader_notWARNO_notreport1_-
 exec_plCommand_justified_thm, 62
 PlatoonLeader_notWARNO_notreport1_-
 exec_plCommand_lemma, 63
 PlatoonLeader_psgCommand_notDis-
 card_thm, 63
 PlatoonLeader_trap_psgCommand_jus-
 tified_lemma, 63
 PlatoonLeader_trap_psgCommand_lemma,
 64
 PlatoonLeader_WARNO_exec_report1_-
 justified_lemma, 64
 PlatoonLeader_WARNO_exec_report1_-
 justified_thm, 66
 PlatoonLeader_WARNO_exec_report1_-
 lemma, 67
 PlatoonSergeant_trap_plCommand_jus-
 tified_lemma, 67
 PlatoonSergeant_trap_plCommand_jus-
 tified_thm, 68
 PlatoonSergeant_trap_plCommand_lemma,
 68