Contents

1 ConductORPType Theory

Built: 16 May 2018
Parent Theories: indexedLists, patternMatches

1.1 Datatypes

plCommand = secure | withdraw | complete | plIncomplete

psgCommand = actionsIn | psgIncomplete

slCommand = PL plCommand | PSG psgCommand

slOutput = ConductORP | Secure | ActionsIn | Withdraw | Complete
         | unAuthenticated | unAuthorized

slState = CONDUCT_ORP | SECURE | ACTIONS_IN | WITHDRAW
         | COMPLETE

stateRole = PlatoonLeader | PlatoonSergeant

1.2 Theorems

[plCommand_distinct_clauses]
⊢ secure ≠ withdraw ∧ secure ≠ complete ∧
  secure ≠ plIncomplete ∧ withdraw ≠ complete ∧
  withdraw ≠ plIncomplete ∧ complete ≠ plIncomplete

[psgCommand_distinct_clauses]
⊢ actionsIn ≠ psgIncomplete

[slCommand_distinct_clauses]
⊢ ∀ a' a. PL a ≠ PSG a'

[slCommand_one_one]
⊢ (∀ a a'. (PL a = PL a') ⟺ (a = a')) ∧
  ∀ a a'. (PSG a = PSG a') ⟺ (a = a')

[slOutput_distinct_clauses]
⊢ ConductORP ≠ Secure ∧ ConductORP ≠ ActionsIn ∧
  ConductORP ≠ Withdraw ∧ ConductORP ≠ Complete ∧
  ConductORP ≠ unAuthenticated ∧ ConductORP ≠ unAuthorized ∧
  Secure ≠ ActionsIn ∧ Secure ≠ Withdraw ∧ Secure ≠ Complete ∧
  Secure ≠ unAuthenticated ∧ Secure ≠ unAuthorized ∧
  ActionsIn ≠ Withdraw ∧ ActionsIn ≠ Complete ∧
  ActionsIn ≠ unAuthenticated ∧ ActionsIn ≠ unAuthorized ∧
  Withdraw ≠ Complete ∧ Withdraw ≠ unAuthenticated ∧
  Withdraw ≠ unAuthorized ∧ Complete ≠ unAuthenticated ∧
  Complete ≠ unAuthorized ∧ unAuthenticated ≠ 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

2 ssmConductORP Theory

Built: 16 May 2018

Parent Theories: ConductORPType, ssm11, OMNIType

2.1 Definitions

[secContextConductORP_def]

⊢ ∀ $plcmd$ $psgcmd$ $incomplete$.
   secContextConductORP $plcmd$ $psgcmd$ $incomplete$ =
   [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]

[ssmConductORPStateInterp_def]

⊢ ∀ $slState$. ssmConductORPStateInterp $slState$ = TT

2.2 Theorems

[authTestConductORP_cmd_reject_lemma]

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

[authTestConductORP_def]

⊢ (authTestConductORP (Name PlatoonLeader says prop cmd) ⟺
   T) ∧
   (authTestConductORP (Name PlatoonSergeant says prop cmd) ⟺
   T) ∧ (authTestConductORP TT ⟺ F) ∧
   (authTestConductORP FF ⟺ F) ∧
   (authTestConductORP (prop v) ⟺ F) ∧
   (authTestConductORP (notf v_1) ⟺ F) ∧
   (authTestConductORP (v_2 andf v_3) ⟺ F) ∧
   (authTestConductORP (v_4 orf v_5) ⟺ F) ∧

```
(authTestConductORP (v₆ impf v₇) ⟺ F) ∧
(authTestConductORP (v₈ eqf v₉) ⟺ F) ∧
(authTestConductORP (v₁₀ says TT) ⟺ F) ∧
(authTestConductORP (v₁₀ says FF) ⟺ F) ∧
(authTestConductORP (v133 meet v134 says prop v₆₆) ⟺ F) ∧
(authTestConductORP (v135 quoting v136 says prop v₆₆) ⟺ F) ∧
(authTestConductORP (v₁₀ says notf v₆₇) ⟺ F) ∧
(authTestConductORP (v₁₀ says (v₆₈ andf v₆₉)) ⟺ F) ∧
(authTestConductORP (v₁₀ says (v₇₀ orf v₇₁)) ⟺ F) ∧
(authTestConductORP (v₁₀ says (v₇₂ impf v₇₃)) ⟺ F) ∧
(authTestConductORP (v₁₀ says (v₇₄ eqf v₇₅)) ⟺ F) ∧
(authTestConductORP (v₁₀ says v₇₆ says v₇₇) ⟺ F) ∧
(authTestConductORP (v₁₀ says v₇₈ speaks_for v₇₉) ⟺ F) ∧
(authTestConductORP (v₁₀ says v₈₀ controls v₈₁) ⟺ F) ∧
(authTestConductORP (v₁₀ says reps v₈₂ v₈₃ v₈₄) ⟺ F) ∧
(authTestConductORP (v₁₀ says v₈₅ domi v₈₆) ⟺ F) ∧
(authTestConductORP (v₁₀ says v₈₇ eqi v₈₈) ⟺ F) ∧
(authTestConductORP (v₁₀ says v₈₉ doms v₉₀) ⟺ F) ∧
(authTestConductORP (v₁₀ says v₉₁ eqs v₉₂) ⟺ F) ∧
(authTestConductORP (v₁₀ says v₉₃ eqn v₉₄) ⟺ F) ∧
(authTestConductORP (v₁₀ says v₉₅ lte v₉₆) ⟺ F) ∧
(authTestConductORP (v₁₀ says v₉₇ lt v₉₈) ⟺ F) ∧
(authTestConductORP (v₁₂ speaks_for v₁₃) ⟺ F) ∧
(authTestConductORP (v₁₄ controls v₁₅) ⟺ F) ∧
(authTestConductORP (reps v₁₆ v₁₇ v₁₈) ⟺ F) ∧
(authTestConductORP (v₁₉ domi v₂₀) ⟺ F) ∧
(authTestConductORP (v₂₁ eqi v₂₂) ⟺ F) ∧
(authTestConductORP (v₂₃ doms v₂₄) ⟺ F) ∧
(authTestConductORP (v₂₅ eqs v₂₆) ⟺ F) ∧
(authTestConductORP (v₂₇ eqn v₂₈) ⟺ F) ∧
(authTestConductORP (v₂₉ lte v₃₀) ⟺ F) ∧
(authTestConductORP (v₃₁ lt v₃₂) ⟺ F)
```

[authTestConductORP_ind]

$\vdash \forall P.$
$\quad (\forall \, cmd. \; P \; (\text{Name PlatoonLeader says prop } cmd)) \; \wedge$
$\quad (\forall \, cmd. \; P \; (\text{Name PlatoonSergeant says prop } cmd)) \; \wedge \; P \; \text{TT} \; \wedge$
$\quad P \; \text{FF} \; \wedge \; (\forall \, v. \; P \; (\text{prop } v)) \; \wedge \; (\forall \, v_1. \; P \; (\text{notf } v_1)) \; \wedge$
$\quad (\forall \, v_2 \; v_3. \; P \; (v_2 \; \text{andf } v_3)) \; \wedge \; (\forall \, v_4 \; v_5. \; P \; (v_4 \; \text{orf } v_5)) \; \wedge$
$\quad (\forall \, v_6 \; v_7. \; P \; (v_6 \; \text{impf } v_7)) \; \wedge \; (\forall \, v_8 \; v_9. \; P \; (v_8 \; \text{eqf } v_9)) \; \wedge$
$\quad (\forall \, v_{10}. \; P \; (v_{10} \; \text{says TT})) \; \wedge \; (\forall \, v_{10}. \; P \; (v_{10} \; \text{says FF})) \; \wedge$
$\quad (\forall \, v133 \; v134 \; v_{66}. \; P \; (v133 \; \text{meet } v134 \; \text{says prop } v_{66})) \; \wedge$
$\quad (\forall \, v135 \; v136 \; v_{66}. \; P \; (v135 \; \text{quoting } v136 \; \text{says prop } v_{66})) \; \wedge$
$\quad (\forall \, v_{10} \; v_{67}. \; P \; (v_{10} \; \text{says notf } v_{67})) \; \wedge$
$\quad (\forall \, v_{10} \; v_{68} \; v_{69}. \; P \; (v_{10} \; \text{says } (v_{68} \; \text{andf } v_{69}))) \; \wedge$
$\quad (\forall \, v_{10} \; v_{70} \; v_{71}. \; P \; (v_{10} \; \text{says } (v_{70} \; \text{orf } v_{71}))) \; \wedge$
$\quad (\forall \, v_{10} \; v_{72} \; v_{73}. \; P \; (v_{10} \; \text{says } (v_{72} \; \text{impf } v_{73}))) \; \wedge$
$\quad (\forall \, v_{10} \; v_{74} \; v_{75}. \; P \; (v_{10} \; \text{says } (v_{74} \; \text{eqf } v_{75}))) \; \wedge$
$\quad (\forall \, v_{10} \; v_{76} \; v_{77}. \; P \; (v_{10} \; \text{says } v_{76} \; \text{says } v_{77})) \; \wedge$

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

[conductORPNS_def]

$\vdash$ (conductORPNS CONDUCT_ORP (exec (PL secure)) = SECURE) $\wedge$
(conductORPNS CONDUCT_ORP (exec (PL plIncomplete)) =
CONDUCT_ORP) $\wedge$
(conductORPNS SECURE (exec (PSG actionsIn)) = ACTIONS_IN) $\wedge$
(conductORPNS SECURE (exec (PSG psgIncomplete)) = SECURE) $\wedge$
(conductORPNS ACTIONS_IN (exec (PL withdraw)) = WITHDRAW) $\wedge$
(conductORPNS ACTIONS_IN (exec (PL plIncomplete)) =
ACTIONS_IN) $\wedge$
(conductORPNS WITHDRAW (exec (PL complete)) = COMPLETE) $\wedge$
(conductORPNS WITHDRAW (exec (PL plIncomplete)) = WITHDRAW) $\wedge$
(conductORPNS s (trap (PL cmd')) = s) $\wedge$
(conductORPNS s (trap (PSG cmd)) = s) $\wedge$
(conductORPNS s (discard (PL cmd')) = s) $\wedge$
(conductORPNS s (discard (PSG cmd)) = s)

[conductORPNS_ind]

$\vdash \forall P.$
P CONDUCT_ORP (exec (PL secure)) $\wedge$
P CONDUCT_ORP (exec (PL plIncomplete)) $\wedge$
P SECURE (exec (PSG actionsIn)) $\wedge$
P SECURE (exec (PSG psgIncomplete)) $\wedge$
P ACTIONS_IN (exec (PL withdraw)) $\wedge$
P ACTIONS_IN (exec (PL plIncomplete)) $\wedge$
P WITHDRAW (exec (PL complete)) $\wedge$
P WITHDRAW (exec (PL plIncomplete)) $\wedge$
$(\forall s \ cmd. \ P \ s \ (\mathtt{trap} \ (\mathtt{PL} \ cmd))) \ \wedge$
$(\forall s \ cmd. \ P \ s \ (\mathtt{trap} \ (\mathtt{PSG} \ cmd))) \ \wedge$
$(\forall s \ cmd. \ P \ s \ (\mathtt{discard} \ (\mathtt{PL} \ cmd))) \ \wedge$

```
(∀ s cmd. P s (discard (PSG cmd))) ∧
P CONDUCT_ORP (exec (PL withdraw)) ∧
P CONDUCT_ORP (exec (PL complete)) ∧
(∀ v₁₁. P CONDUCT_ORP (exec (PSG v₁₁))) ∧
(∀ v₁₃. P SECURE (exec (PL v₁₃))) ∧
P ACTIONS_IN (exec (PL secure)) ∧
P ACTIONS_IN (exec (PL complete)) ∧
(∀ v₁₇. P ACTIONS_IN (exec (PSG v₁₇))) ∧
P WITHDRAW (exec (PL secure)) ∧
P WITHDRAW (exec (PL withdraw)) ∧
(∀ v₂₀. P WITHDRAW (exec (PSG v₂₀))) ∧
(∀ v₂₁. P COMPLETE (exec v₂₁)) ⇒
∀ v  v₁.  P  v  v₁
```

[conductORPOut_def]

```
⊢ (conductORPOut CONDUCT_ORP (exec (PL secure)) = Secure) ∧
  (conductORPOut CONDUCT_ORP (exec (PL plIncomplete)) =
   ConductORP) ∧
  (conductORPOut SECURE (exec (PSG actionsIn)) = ActionsIn) ∧
  (conductORPOut SECURE (exec (PSG psgIncomplete)) = Secure) ∧
  (conductORPOut ACTIONS_IN (exec (PL withdraw)) = Withdraw) ∧
  (conductORPOut ACTIONS_IN (exec (PL plIncomplete)) =
   ActionsIn) ∧
  (conductORPOut WITHDRAW (exec (PL complete)) = Complete) ∧
  (conductORPOut WITHDRAW (exec (PL plIncomplete)) =
   Withdraw) ∧
  (conductORPOut s (trap (PL cmd′)) = unAuthorized) ∧
  (conductORPOut s (trap (PSG cmd)) = unAuthorized) ∧
  (conductORPOut s (discard (PL cmd′)) = unAuthenticated) ∧
  (conductORPOut s (discard (PSG cmd)) = unAuthenticated)
```

[conductORPOut_ind]

```
⊢ ∀ P.
    P CONDUCT_ORP (exec (PL secure)) ∧
    P CONDUCT_ORP (exec (PL plIncomplete)) ∧
    P SECURE (exec (PSG actionsIn)) ∧
    P SECURE (exec (PSG psgIncomplete)) ∧
    P ACTIONS_IN (exec (PL withdraw)) ∧
    P ACTIONS_IN (exec (PL plIncomplete)) ∧
    P WITHDRAW (exec (PL complete)) ∧
    P WITHDRAW (exec (PL plIncomplete)) ∧
    (∀ s cmd. P s (trap (PL cmd))) ∧
    (∀ s cmd. P s (trap (PSG cmd))) ∧
    (∀ s cmd. P s (discard (PL cmd))) ∧
    (∀ s cmd. P s (discard (PSG cmd))) ∧
    P CONDUCT_ORP (exec (PL withdraw)) ∧
    P CONDUCT_ORP (exec (PL complete)) ∧
    (∀ v₁₁. P CONDUCT_ORP (exec (PSG v₁₁))) ∧
    (∀ v₁₃. P SECURE (exec (PL v₁₃))) ∧
```

P `ACTIONS_IN (exec (PL secure))` $\wedge$
P `ACTIONS_IN (exec (PL complete))` $\wedge$
$(\forall v_{17}.\ P$ `ACTIONS_IN (exec (PSG` v_{17}`)))` $\wedge$
P `WITHDRAW (exec (PL secure))` $\wedge$
P `WITHDRAW (exec (PL withdraw))` $\wedge$
$(\forall v_{20}.\ P$ `WITHDRAW (exec (PSG` v_{20}`)))` $\wedge$
$(\forall v_{21}.\ P$ `COMPLETE (exec` v_{21}`))` $\Rightarrow$
$\forall v\ \ v_1.\ P\ \ v\ \ v_1$

[PlatoonLeader_exec_plCommand_justified_thm]

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

[PlatoonLeader_plCommand_lemma]

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

[PlatoonSergeant_exec_psgCommand_justified_thm]

$\vdash \forall NS\ Out\ M\ Oi\ Os.$
$\quad$ TR (M, Oi, Os) (exec (SLc (PSG $psgCommand$)))
$\quad\quad$ (CFG authTestConductORP ssmConductORPStateInterp
$\quad\quad\quad$ (secContextConductORP $plCommand$ $psgCommand$ $incomplete$)
$\quad\quad\quad$ (Name PlatoonSergeant says
$\quad\quad\quad\quad$ prop (SOME (SLc (PSG $psgCommand$)))::ins) s $outs$)
$\quad\quad$ (CFG authTestConductORP ssmConductORPStateInterp
$\quad\quad\quad$ (secContextConductORP $plCommand$ $psgCommand$ $incomplete$)
$\quad\quad\quad$ ins (NS s (exec (SLc (PSG $psgCommand$))))

```
          (Out s (exec (SLc (PSG psgCommand)))::outs))  ⟺
     authTestConductORP
       (Name PlatoonSergeant says
        prop (SOME (SLc (PSG psgCommand)))) ∧
     CFGInterpret (M,Oi,Os)
       (CFG authTestConductORP ssmConductORPStateInterp
          (secContextConductORP plCommand psgCommand incomplete)
          (Name PlatoonSergeant says
           prop (SOME (SLc (PSG psgCommand)))::ins) s outs) ∧
     (M,Oi,Os) sat prop (SOME (SLc (PSG psgCommand)))
```

[PlatoonSergeant_psgCommand_lemma]

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

Index