Contents

1 ConductORPType Theory

Built: 11 June 2018
Parent Theories: indexedLists, patternMatches

1.1 Datatypes

```
omniCommand = ssmSecureComplete | ssmActionsInComplete
            | ssmWithdrawComplete | invalidOmniCommand

plCommand = secure | withdraw | complete | plIncomplete

psgCommand = actionsIn | psgIncomplete

slCommand = PL plCommand | PSG psgCommand | OMNI omniCommand

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

slState = CONDUCT_ORP | SECURE | ACTIONS_IN | WITHDRAW
        | COMPLETE

stateRole = PlatoonLeader | PlatoonSergeant | Omni
```

1.2 Theorems

[omniCommand_distinct_clauses]

$\vdash$ ssmSecureComplete $\neq$ ssmActionsInComplete $\wedge$
ssmSecureComplete $\neq$ ssmWithdrawComplete $\wedge$
ssmSecureComplete $\neq$ invalidOmniCommand $\wedge$
ssmActionsInComplete $\neq$ ssmWithdrawComplete $\wedge$
ssmActionsInComplete $\neq$ invalidOmniCommand $\wedge$
ssmWithdrawComplete $\neq$ invalidOmniCommand

[plCommand_distinct_clauses]

$\vdash$ secure $\neq$ withdraw $\wedge$ secure $\neq$ complete $\wedge$
secure $\neq$ plIncomplete $\wedge$ withdraw $\neq$ complete $\wedge$
withdraw $\neq$ plIncomplete $\wedge$ complete $\neq$ plIncomplete

[psgCommand_distinct_clauses]

$\vdash$ actionsIn $\neq$ psgIncomplete

[slCommand_distinct_clauses]

$\vdash$ ($\forall a'\ a.$ PL $a \neq$ PSG a') $\wedge$ ($\forall a'\ a.$ PL $a \neq$ OMNI a') $\wedge$
$\forall a'\ a.$ PSG $a \neq$ OMNI a'

[slCommand_one_one]

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

[slOutput_distinct_clauses]

$\vdash$ ConductORP $\neq$ Secure $\land$ ConductORP $\neq$ ActionsIn $\land$
ConductORP $\neq$ Withdraw $\land$ ConductORP $\neq$ Complete $\land$
ConductORP $\neq$ unAuthenticated $\land$ ConductORP $\neq$ unAuthorized $\land$
Secure $\neq$ ActionsIn $\land$ Secure $\neq$ Withdraw $\land$ Secure $\neq$ Complete $\land$
Secure $\neq$ unAuthenticated $\land$ Secure $\neq$ unAuthorized $\land$
ActionsIn $\neq$ Withdraw $\land$ ActionsIn $\neq$ Complete $\land$
ActionsIn $\neq$ unAuthenticated $\land$ ActionsIn $\neq$ unAuthorized $\land$
Withdraw $\neq$ Complete $\land$ Withdraw $\neq$ unAuthenticated $\land$
Withdraw $\neq$ unAuthorized $\land$ Complete $\neq$ unAuthenticated $\land$
Complete $\neq$ unAuthorized $\land$ unAuthenticated $\neq$ unAuthorized

[slRole_distinct_clauses]

$\vdash$ PlatoonLeader $\neq$ PlatoonSergeant $\land$ PlatoonLeader $\neq$ Omni $\land$
PlatoonSergeant $\neq$ Omni

[slState_distinct_clauses]

$\vdash$ CONDUCT_ORP $\neq$ SECURE $\land$ CONDUCT_ORP $\neq$ ACTIONS_IN $\land$
CONDUCT_ORP $\neq$ WITHDRAW $\land$ CONDUCT_ORP $\neq$ COMPLETE $\land$
SECURE $\neq$ ACTIONS_IN $\land$ SECURE $\neq$ WITHDRAW $\land$ SECURE $\neq$ COMPLETE $\land$
ACTIONS_IN $\neq$ WITHDRAW $\land$ ACTIONS_IN $\neq$ COMPLETE $\land$
WITHDRAW $\neq$ COMPLETE

2 ssmConductORP Theory

Built: 11 June 2018
Parent Theories: ConductORPDef

2.1 Theorems

[conductORPNS_def]

$\vdash$ (conductORPNS CONDUCT_ORP (exec x) =
 if getPlCom x = secure **then** SECURE **else** CONDUCT_ORP) $\land$
(conductORPNS SECURE (exec x) =
 if getPsgCom x = actionsIn **then** ACTIONS_IN **else** SECURE) $\land$
(conductORPNS ACTIONS_IN (exec x) =
 if getPlCom x = withdraw **then** WITHDRAW **else** ACTIONS_IN) $\land$
(conductORPNS WITHDRAW (exec x) =
 if getPlCom x = complete **then** COMPLETE **else** WITHDRAW) $\land$
(conductORPNS s (trap x) = s) $\land$
(conductORPNS s (discard x) = s)

[conductORPNS_ind]

```
⊢ ∀P.
    (∀x. P CONDUCT_ORP (exec x)) ∧ (∀x. P SECURE (exec x)) ∧
    (∀x. P ACTIONS_IN (exec x)) ∧ (∀x. P WITHDRAW (exec x)) ∧
    (∀s x. P s (trap x)) ∧ (∀s x. P s (discard x)) ∧
    (∀v5. P COMPLETE (exec v5)) ⇒
    ∀v v1. P v v1
```

[conductORPOut_def]

```
⊢ (conductORPOut CONDUCT_ORP (exec x) =
   if getPlCom x = secure then Secure else ConductORP) ∧
  (conductORPOut SECURE (exec x) =
   if getPsgCom x = actionsIn then ActionsIn else Secure) ∧
  (conductORPOut ACTIONS_IN (exec x) =
   if getPlCom x = withdraw then Withdraw else ActionsIn) ∧
  (conductORPOut WITHDRAW (exec x) =
   if getPlCom x = complete then Complete else Withdraw) ∧
  (conductORPOut s (trap x) = unAuthorized) ∧
  (conductORPOut s (discard x) = unAuthenticated)
```

[conductORPOut_ind]

```
⊢ ∀P.
    (∀x. P CONDUCT_ORP (exec x)) ∧ (∀x. P SECURE (exec x)) ∧
    (∀x. P ACTIONS_IN (exec x)) ∧ (∀x. P WITHDRAW (exec x)) ∧
    (∀s x. P s (trap x)) ∧ (∀s x. P s (discard x)) ∧
    (∀v5. P COMPLETE (exec v5)) ⇒
    ∀v v1. P v v1
```

[inputOK_cmd_reject_lemma]

```
⊢ ∀cmd. ¬inputOK (prop (SOME cmd))
```

[inputOK_def]

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

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

[inputOK_ind]

$\vdash \forall P.$
$(\forall cmd.\ P\ (\texttt{Name PlatoonLeader says prop}\ cmd)) \land$
$(\forall cmd.\ P\ (\texttt{Name PlatoonSergeant says prop}\ cmd)) \land$
$(\forall cmd.\ P\ (\texttt{Name Omni says prop}\ cmd)) \land P\ \texttt{TT} \land P\ \texttt{FF} \land$
$(\forall v.\ P\ (\texttt{prop}\ v)) \land (\forall v_1.\ P\ (\texttt{notf}\ v_1)) \land$
$(\forall v_2\ v_3.\ P\ (v_2\ \texttt{andf}\ v_3)) \land (\forall v_4\ v_5.\ P\ (v_4\ \texttt{orf}\ v_5)) \land$
$(\forall v_6\ v_7.\ P\ (v_6\ \texttt{impf}\ v_7)) \land (\forall v_8\ v_9.\ P\ (v_8\ \texttt{eqf}\ v_9)) \land$
$(\forall v_{10}.\ P\ (v_{10}\ \texttt{says TT})) \land (\forall v_{10}.\ P\ (v_{10}\ \texttt{says FF})) \land$
$(\forall v133\ v134\ v_{66}.\ P\ (v133\ \texttt{meet}\ v134\ \texttt{says prop}\ v_{66})) \land$
$(\forall v135\ v136\ v_{66}.\ P\ (v135\ \texttt{quoting}\ v136\ \texttt{says prop}\ v_{66})) \land$
$(\forall v_{10}\ v_{67}.\ P\ (v_{10}\ \texttt{says notf}\ v_{67})) \land$
$(\forall v_{10}\ v_{68}\ v_{69}.\ P\ (v_{10}\ \texttt{says}\ (v_{68}\ \texttt{andf}\ v_{69}))) \land$
$(\forall v_{10}\ v_{70}\ v_{71}.\ P\ (v_{10}\ \texttt{says}\ (v_{70}\ \texttt{orf}\ v_{71}))) \land$
$(\forall v_{10}\ v_{72}\ v_{73}.\ P\ (v_{10}\ \texttt{says}\ (v_{72}\ \texttt{impf}\ v_{73}))) \land$
$(\forall v_{10}\ v_{74}\ v_{75}.\ P\ (v_{10}\ \texttt{says}\ (v_{74}\ \texttt{eqf}\ v_{75}))) \land$
$(\forall v_{10}\ v_{76}\ v_{77}.\ P\ (v_{10}\ \texttt{says}\ v_{76}\ \texttt{says}\ v_{77})) \land$
$(\forall v_{10}\ v_{78}\ v_{79}.\ P\ (v_{10}\ \texttt{says}\ v_{78}\ \texttt{speaks\_for}\ v_{79})) \land$
$(\forall v_{10}\ v_{80}\ v_{81}.\ P\ (v_{10}\ \texttt{says}\ v_{80}\ \texttt{controls}\ v_{81})) \land$
$(\forall v_{10}\ v_{82}\ v_{83}\ v_{84}.\ P\ (v_{10}\ \texttt{says reps}\ v_{82}\ v_{83}\ v_{84})) \land$
$(\forall v_{10}\ v_{85}\ v_{86}.\ P\ (v_{10}\ \texttt{says}\ v_{85}\ \texttt{domi}\ v_{86})) \land$
$(\forall v_{10}\ v_{87}\ v_{88}.\ P\ (v_{10}\ \texttt{says}\ v_{87}\ \texttt{eqi}\ v_{88})) \land$
$(\forall v_{10}\ v_{89}\ v_{90}.\ P\ (v_{10}\ \texttt{says}\ v_{89}\ \texttt{doms}\ v_{90})) \land$
$(\forall v_{10}\ v_{91}\ v_{92}.\ P\ (v_{10}\ \texttt{says}\ v_{91}\ \texttt{eqs}\ v_{92})) \land$
$(\forall v_{10}\ v_{93}\ v_{94}.\ P\ (v_{10}\ \texttt{says}\ v_{93}\ \texttt{eqn}\ v_{94})) \land$
$(\forall v_{10}\ v_{95}\ v_{96}.\ P\ (v_{10}\ \texttt{says}\ v_{95}\ \texttt{lte}\ v_{96})) \land$
$(\forall v_{10}\ v_{97}\ v_{98}.\ P\ (v_{10}\ \texttt{says}\ v_{97}\ \texttt{lt}\ v_{98})) \land$
$(\forall v_{12}\ v_{13}.\ P\ (v_{12}\ \texttt{speaks\_for}\ v_{13})) \land$
$(\forall v_{14}\ v_{15}.\ P\ (v_{14}\ \texttt{controls}\ v_{15})) \land$
$(\forall v_{16}\ v_{17}\ v_{18}.\ P\ (\texttt{reps}\ v_{16}\ v_{17}\ v_{18})) \land$

```
(∀ v19 v20. P (v19 domi v20)) ∧
(∀ v21 v22. P (v21 eqi v22)) ∧
(∀ v23 v24. P (v23 doms v24)) ∧
(∀ v25 v26. P (v25 eqs v26)) ∧ (∀ v27 v28. P (v27 eqn v28)) ∧
(∀ v29 v30. P (v29 lte v30)) ∧ (∀ v31 v32. P (v31 lt v32)) ⇒
∀ v. P v
```

[PlatoonLeader_ACTIONS_IN_exec_justified_lemma]

```
⊢ ∀ NS Out M Oi Os.
     TR (M,Oi,Os)
       (exec
          (inputList
             [Name Omni says
              prop (SOME (SLc (OMNI ssmActionsInComplete)));
              Name PlatoonLeader says
              prop (SOME (SLc (PL withdraw)))]))
       (CFG inputOK secContext secAuthorization
          ([Name Omni says
            prop (SOME (SLc (OMNI ssmActionsInComplete)));
            Name PlatoonLeader says
            prop (SOME (SLc (PL withdraw)))]::ins) ACTIONS_IN
          outs)
       (CFG inputOK secContext secAuthorization ins
          (NS ACTIONS_IN
             (exec
                (inputList
                   [Name Omni says
                    prop
                      (SOME (SLc (OMNI ssmActionsInComplete)));
                    Name PlatoonLeader says
                    prop (SOME (SLc (PL withdraw)))])))
          (Out ACTIONS_IN
             (exec
                (inputList
                   [Name Omni says
                    prop
                      (SOME (SLc (OMNI ssmActionsInComplete)));
                    Name PlatoonLeader says
                    prop (SOME (SLc (PL withdraw)))]))::
                outs)) ⟺
     authenticationTest inputOK
       [Name Omni says
        prop (SOME (SLc (OMNI ssmActionsInComplete)));
        Name PlatoonLeader says
        prop (SOME (SLc (PL withdraw)))] ∧
     CFGInterpret (M,Oi,Os)
       (CFG inputOK secContext secAuthorization
          ([Name Omni says
            prop (SOME (SLc (OMNI ssmActionsInComplete)));
          Name PlatoonLeader says
          prop (SOME (SLc (PL withdraw)))]::ins) ACTIONS_IN
        outs) ∧
   (M,Oi,Os) satList
   propCommandList
     [Name Omni says
      prop (SOME (SLc (OMNI ssmActionsInComplete)));
      Name PlatoonLeader says prop (SOME (SLc (PL withdraw)))]
```

[PlatoonLeader_ACTIONS_IN_exec_justified_thm]

```
 ⊢ ∀NS Out M Oi Os.
     TR (M,Oi,Os)
       (exec
          [SOME (SLc (OMNI ssmActionsInComplete));
           SOME (SLc (PL withdraw))])
       (CFG inputOK secContext secAuthorization
          ([Name Omni says
            prop (SOME (SLc (OMNI ssmActionsInComplete)));
            Name PlatoonLeader says
            prop (SOME (SLc (PL withdraw)))]::ins) ACTIONS_IN
          outs)
       (CFG inputOK secContext secAuthorization ins
          (NS ACTIONS_IN
             (exec
                [SOME (SLc (OMNI ssmActionsInComplete));
                 SOME (SLc (PL withdraw))]))
          (Out ACTIONS_IN
             (exec
                [SOME (SLc (OMNI ssmActionsInComplete));
                 SOME (SLc (PL withdraw))])::outs)) ⟺
     authenticationTest inputOK
       [Name Omni says
        prop (SOME (SLc (OMNI ssmActionsInComplete)));
        Name PlatoonLeader says
        prop (SOME (SLc (PL withdraw)))] ∧
     CFGInterpret (M,Oi,Os)
       (CFG inputOK secContext secAuthorization
          ([Name Omni says
            prop (SOME (SLc (OMNI ssmActionsInComplete)));
            Name PlatoonLeader says
            prop (SOME (SLc (PL withdraw)))]::ins) ACTIONS_IN
          outs) ∧
     (M,Oi,Os) satList
     [prop (SOME (SLc (OMNI ssmActionsInComplete)));
      prop (SOME (SLc (PL withdraw)))]
```

[PlatoonLeader_ACTIONS_IN_exec_lemma]

```
 ⊢ ∀M Oi Os.
     CFGInterpret (M,Oi,Os)
```

```
        (CFG inputOK secContext secAuthorization
           ([Name Omni says
             prop (SOME (SLc (OMNI ssmActionsInComplete)));
             Name PlatoonLeader says
             prop (SOME (SLc (PL withdraw)))]::ins) ACTIONS_IN
           outs) ⇒
      (M,Oi,Os) satList
      propCommandList
        [Name Omni says
         prop (SOME (SLc (OMNI ssmActionsInComplete)));
         Name PlatoonLeader says prop (SOME (SLc (PL withdraw)))]
```

[PlatoonLeader_ACTIONS_IN_trap_justified_lemma]

```
 ⊢ omniCommand ≠ ssmActionsInComplete ⇒
   (s = ACTIONS_IN) ⇒
   ∀ NS Out M Oi Os.
      TR (M,Oi,Os)
        (trap
           (inputList
              [Name Omni says
               prop (SOME (SLc (OMNI omniCommand)));
               Name PlatoonLeader says
               prop (SOME (SLc (PL withdraw)))]))
        (CFG inputOK secContext secAuthorization
           ([Name Omni says prop (SOME (SLc (OMNI omniCommand)));
             Name PlatoonLeader says
             prop (SOME (SLc (PL withdraw)))]::ins) ACTIONS_IN
           outs)
        (CFG inputOK secContext secAuthorization ins
           (NS ACTIONS_IN
              (trap
                 (inputList
                    [Name Omni says
                     prop (SOME (SLc (OMNI omniCommand)));
                     Name PlatoonLeader says
                     prop (SOME (SLc (PL withdraw)))])))
           (Out ACTIONS_IN
              (trap
                 (inputList
                    [Name Omni says
                     prop (SOME (SLc (OMNI omniCommand)));
                     Name PlatoonLeader says
                     prop (SOME (SLc (PL withdraw)))]))::
                outs)) ⇔
      authenticationTest inputOK
        [Name Omni says prop (SOME (SLc (OMNI omniCommand)));
         Name PlatoonLeader says
         prop (SOME (SLc (PL withdraw)))] ∧
      CFGInterpret (M,Oi,Os)
```

```
        (CFG inputOK secContext secAuthorization
           ([Name Omni says prop (SOME (SLc (OMNI omniCommand)));
             Name PlatoonLeader says
             prop (SOME (SLc (PL withdraw)))]::ins) ACTIONS_IN
           outs) ∧ (M,Oi,Os) sat prop NONE
```

[PlatoonLeader_ACTIONS_IN_trap_justified_thm]

```
 ⊢ omniCommand ≠ ssmActionsInComplete ⇒
   (s = ACTIONS_IN) ⇒
   ∀NS Out M Oi Os.
     TR (M,Oi,Os)
       (trap
          [SOME (SLc (OMNI omniCommand));
           SOME (SLc (PL withdraw))])
       (CFG inputOK secContext secAuthorization
          ([Name Omni says prop (SOME (SLc (OMNI omniCommand)));
            Name PlatoonLeader says
            prop (SOME (SLc (PL withdraw)))]::ins) ACTIONS_IN
          outs)
       (CFG inputOK secContext secAuthorization ins
          (NS ACTIONS_IN
             (trap
                [SOME (SLc (OMNI omniCommand));
                 SOME (SLc (PL withdraw))]))
          (Out ACTIONS_IN
             (trap
                [SOME (SLc (OMNI omniCommand));
                 SOME (SLc (PL withdraw))])::outs)) ⇔
     authenticationTest inputOK
       [Name Omni says prop (SOME (SLc (OMNI omniCommand)));
        Name PlatoonLeader says
        prop (SOME (SLc (PL withdraw)))] ∧
     CFGInterpret (M,Oi,Os)
       (CFG inputOK secContext secAuthorization
          ([Name Omni says prop (SOME (SLc (OMNI omniCommand)));
            Name PlatoonLeader says
            prop (SOME (SLc (PL withdraw)))]::ins) ACTIONS_IN
          outs) ∧ (M,Oi,Os) sat prop NONE
```

[PlatoonLeader_ACTIONS_IN_trap_lemma]

```
 ⊢ omniCommand ≠ ssmActionsInComplete ⇒
   (s = ACTIONS_IN) ⇒
   ∀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 withdraw)))]::ins) ACTIONS_IN
          outs) ⇒
     (M,Oi,Os) sat prop NONE
```

[PlatoonLeader_CONDUCT_ORP_exec_secure_justified_thm]

```
 ⊢ ∀NS Out M Oi Os.
     TR (M,Oi,Os) (exec [SOME (SLc (PL secure))])
       (CFG inputOK secContext secAuthorization
          ([Name PlatoonLeader says
            prop (SOME (SLc (PL secure)))]::ins) CONDUCT_ORP
          outs)
       (CFG inputOK secContext secAuthorization ins
          (NS CONDUCT_ORP (exec [SOME (SLc (PL secure))]))
          (Out CONDUCT_ORP (exec [SOME (SLc (PL secure))])::
               outs)) ⟺
     authenticationTest inputOK
       [Name PlatoonLeader says prop (SOME (SLc (PL secure)))] ∧
     CFGInterpret (M,Oi,Os)
       (CFG inputOK secContext secAuthorization
          ([Name PlatoonLeader says
            prop (SOME (SLc (PL secure)))]::ins) CONDUCT_ORP
          outs) ∧
     (M,Oi,Os) satList [prop (SOME (SLc (PL secure)))]
```

[PlatoonLeader_CONDUCT_ORP_exec_secure_lemma]

```
 ⊢ ∀M Oi Os.
     CFGInterpret (M,Oi,Os)
       (CFG inputOK secContext secAuthorization
          ([Name PlatoonLeader says
            prop (SOME (SLc (PL secure)))]::ins) CONDUCT_ORP
          outs) ⇒
     (M,Oi,Os) satList
     propCommandList
       [Name PlatoonLeader says prop (SOME (SLc (PL secure)))]
```

[PlatoonSergeant_SECURE_exec_justified_lemma]

```
 ⊢ ∀NS Out M Oi Os.
     TR (M,Oi,Os)
       (exec
          (inputList
             [Name Omni says
              prop (SOME (SLc (OMNI ssmSecureComplete)));
              Name PlatoonSergeant says
              prop (SOME (SLc (PSG actionsIn)))]))
       (CFG inputOK secContext secAuthorization
          ([Name Omni says
            prop (SOME (SLc (OMNI ssmSecureComplete)));
            Name PlatoonSergeant says
            prop (SOME (SLc (PSG actionsIn)))]::ins) SECURE
          outs)
       (CFG inputOK secContext secAuthorization ins
          (NS SECURE
              (exec
                 (inputList
                    [Name Omni says
                     prop (SOME (SLc (OMNI ssmSecureComplete)));
                     Name PlatoonSergeant says
                     prop (SOME (SLc (PSG actionsIn)))])))
           (Out SECURE
              (exec
                 (inputList
                    [Name Omni says
                     prop (SOME (SLc (OMNI ssmSecureComplete)));
                     Name PlatoonSergeant says
                     prop (SOME (SLc (PSG actionsIn)))])):: 
               outs)) ⇔
     authenticationTest inputOK
       [Name Omni says
        prop (SOME (SLc (OMNI ssmSecureComplete)));
        Name PlatoonSergeant says
        prop (SOME (SLc (PSG actionsIn)))] ∧
     CFGInterpret (M,Oi,Os)
       (CFG inputOK secContext secAuthorization
          ([Name Omni says
            prop (SOME (SLc (OMNI ssmSecureComplete)));
            Name PlatoonSergeant says
            prop (SOME (SLc (PSG actionsIn)))]::ins) SECURE
          outs) ∧
     (M,Oi,Os) satList
     propCommandList
       [Name Omni says
        prop (SOME (SLc (OMNI ssmSecureComplete)));
        Name PlatoonSergeant says
        prop (SOME (SLc (PSG actionsIn)))]
```

[PlatoonSergeant_SECURE_exec_justified_thm]

```
 ⊢ ∀NS Out M Oi Os.
     TR (M,Oi,Os)
       (exec
          [SOME (SLc (OMNI ssmSecureComplete));
           SOME (SLc (PSG actionsIn))])
       (CFG inputOK secContext secAuthorization
          ([Name Omni says
            prop (SOME (SLc (OMNI ssmSecureComplete)));
            Name PlatoonSergeant says
            prop (SOME (SLc (PSG actionsIn)))]::ins) SECURE
          outs)
       (CFG inputOK secContext secAuthorization ins
          (NS SECURE
             (exec
                [SOME (SLc (OMNI ssmSecureComplete));
                SOME (SLc (PSG actionsIn))]))
         (Out SECURE
            (exec
               [SOME (SLc (OMNI ssmSecureComplete));
                SOME (SLc (PSG actionsIn))])::outs)) ⟺
     authenticationTest inputOK
       [Name Omni says
        prop (SOME (SLc (OMNI ssmSecureComplete)));
        Name PlatoonSergeant says
        prop (SOME (SLc (PSG actionsIn)))] ∧
     CFGInterpret (M,Oi,Os)
       (CFG inputOK secContext secAuthorization
          ([Name Omni says
            prop (SOME (SLc (OMNI ssmSecureComplete)));
            Name PlatoonSergeant says
            prop (SOME (SLc (PSG actionsIn)))]::ins) SECURE
          outs) ∧
     (M,Oi,Os) satList
     [prop (SOME (SLc (OMNI ssmSecureComplete)));
      prop (SOME (SLc (PSG actionsIn)))]
```

[PlatoonSergeant_SECURE_exec_lemma]

```
 ⊢ ∀M Oi Os.
     CFGInterpret (M,Oi,Os)
       (CFG inputOK secContext secAuthorization
          ([Name Omni says
            prop (SOME (SLc (OMNI ssmSecureComplete)));
            Name PlatoonSergeant says
            prop (SOME (SLc (PSG actionsIn)))]::ins) SECURE
          outs) ⇒
     (M,Oi,Os) satList
     propCommandList
       [Name Omni says
        prop (SOME (SLc (OMNI ssmSecureComplete)));
        Name PlatoonSergeant says
        prop (SOME (SLc (PSG actionsIn)))]
```

Index