Contents

1 SM0Solutions Theory **3**
1.1 Definitions 3
1.2 Theorems 3

2 SM0 Theory **8**
2.1 Datatypes 8
2.2 Definitions 8
2.3 Theorems 9

1 SM0Solutions Theory

Built: 09 April 2019
Parent Theories: SM0

1.1 Definitions

[certs2_def]

```
⊢ ∀cmd npriv privcmd.
    certs2 cmd npriv privcmd =
    [Name Carol controls prop (SOME (NP npriv));
     Name Carol says prop (SOME (PR privcmd)) impf prop NONE]
```

1.2 Theorems

[Alice_exec_npriv_justified_thm]

```
⊢ ∀NS Out M Oi Os.
    TR (M,Oi,Os) (exec (NP npriv))
      (CFG inputOK SM0StateInterp (certs cmd npriv privcmd)
         (Name Alice says prop (SOME (NP npriv))::ins) s outs)
      (CFG inputOK SM0StateInterp (certs cmd npriv privcmd) ins
         (NS s (exec (NP npriv)))
         (Out s (exec (NP npriv))::outs)) ⟺
    inputOK (Name Alice says prop (SOME (NP npriv))) ∧
    CFGInterpret (M,Oi,Os)
      (CFG inputOK SM0StateInterp (certs cmd npriv privcmd)
         (Name Alice says prop (SOME (NP npriv))::ins) s
         outs) ∧ (M,Oi,Os) sat prop (SOME (NP npriv))
```

[Alice_justified_npriv_exec_thm]

```
⊢ ∀NS Out M Oi Os cmd npriv privcmd ins s outs.
    inputOK (Name Alice says prop (SOME (NP npriv))) ∧
    CFGInterpret (M,Oi,Os)
      (CFG inputOK SM0StateInterp (certs cmd npriv privcmd)
         (Name Alice says prop (SOME (NP npriv))::ins) s
         outs) ⇒
    TR (M,Oi,Os) (exec (NP npriv))
      (CFG inputOK SM0StateInterp (certs cmd npriv privcmd)
         (Name Alice says prop (SOME (NP npriv))::ins) s outs)
      (CFG inputOK SM0StateInterp (certs cmd npriv privcmd) ins
         (NS s (exec (NP npriv)))
         (Out s (exec (NP npriv))::outs))
```

[Alice_npriv_lemma]

```
⊢ CFGInterpret (M,Oi,Os)
    (CFG inputOK SM0StateInterp (certs cmd npriv privcmd)
       (Name Alice says prop (SOME (NP npriv))::ins) s outs) ⇒
  (M,Oi,Os) sat prop (SOME (NP npriv))
```

[Alice_npriv_verified_thm]

```
⊢ ∀NS Out M Oi Os.
    TR (M,Oi,Os) (exec (NP npriv))
      (CFG inputOK SM0StateInterp (certs cmd npriv privcmd)
         (Name Alice says prop (SOME (NP npriv))::ins) s outs)
      (CFG inputOK SM0StateInterp (certs cmd npriv privcmd) ins
         (NS s (exec (NP npriv)))
         (Out s (exec (NP npriv))::outs)) ⇒
    (M,Oi,Os) sat prop (SOME (NP npriv))
```

[Carol_exec_npriv_justified_thm]

```
⊢ ∀NS Out M Oi Os.
    TR (M,Oi,Os) (exec (NP npriv))
      (CFG inputOK2 SM0StateInterp (certs2 cmd npriv privcmd)
         (Name Carol says prop (SOME (NP npriv))::ins) s outs)
      (CFG inputOK2 SM0StateInterp (certs2 cmd npriv privcmd)
         ins (NS s (exec (NP npriv)))
         (Out s (exec (NP npriv))::outs)) ⟺
    inputOK2 (Name Carol says prop (SOME (NP npriv))) ∧
    CFGInterpret (M,Oi,Os)
      (CFG inputOK2 SM0StateInterp (certs2 cmd npriv privcmd)
         (Name Carol says prop (SOME (NP npriv))::ins) s
         outs) ∧ (M,Oi,Os) sat prop (SOME (NP npriv))
```

[Carol_justified_npriv_exec_thm]

```
⊢ ∀NS Out M Oi Os cmd npriv privcmd ins s outs.
    inputOK2 (Name Carol says prop (SOME (NP npriv))) ∧
    CFGInterpret (M,Oi,Os)
      (CFG inputOK2 SM0StateInterp (certs2 cmd npriv privcmd)
         (Name Carol says prop (SOME (NP npriv))::ins) s
         outs) ⇒
    TR (M,Oi,Os) (exec (NP npriv))
      (CFG inputOK2 SM0StateInterp (certs2 cmd npriv privcmd)
         (Name Carol says prop (SOME (NP npriv))::ins) s outs)
      (CFG inputOK2 SM0StateInterp (certs2 cmd npriv privcmd)
         ins (NS s (exec (NP npriv)))
         (Out s (exec (NP npriv))::outs))
```

[Carol_justified_privcmd_trap_thm]

```
⊢ ∀ NS Out M Oi Os cmd npriv privcmd ins s outs.
    inputOK2 (Name Carol says prop (SOME (PR privcmd))) ∧
    CFGInterpret (M,Oi,Os)
      (CFG inputOK2 SM0StateInterp (certs2 cmd npriv privcmd)
         (Name Carol says prop (SOME (PR privcmd))::ins) s
         outs) ⇒
    TR (M,Oi,Os) (trap (PR privcmd))
      (CFG inputOK2 SM0StateInterp (certs2 cmd npriv privcmd)
         (Name Carol says prop (SOME (PR privcmd))::ins) s
         outs)
      (CFG inputOK2 SM0StateInterp (certs2 cmd npriv privcmd)
         ins (NS s (trap (PR privcmd)))
         (Out s (trap (PR privcmd))::outs))
```

[Carol_npriv_lemma]

```
⊢ CFGInterpret (M,Oi,Os)
    (CFG inputOK2 SM0StateInterp (certs2 cmd npriv privcmd)
       (Name Carol says prop (SOME (NP npriv))::ins) s outs) ⇒
  (M,Oi,Os) sat prop (SOME (NP npriv))
```

[Carol_npriv_verified_thm]

```
⊢ ∀ NS Out M Oi Os.
    TR (M,Oi,Os) (exec (NP npriv))
      (CFG inputOK2 SM0StateInterp (certs2 cmd npriv privcmd)
         (Name Carol says prop (SOME (NP npriv))::ins) s outs)
      (CFG inputOK2 SM0StateInterp (certs2 cmd npriv privcmd)
         ins (NS s (exec (NP npriv)))
         (Out s (exec (NP npriv))::outs)) ⇒
    (M,Oi,Os) sat prop (SOME (NP npriv))
```

[Carol_privcmd_trap_lemma]

```
⊢ CFGInterpret (M,Oi,Os)
    (CFG inputOK2 SM0StateInterp (certs2 cmd npriv privcmd)
       (Name Carol says prop (SOME (PR privcmd))::ins) s
       outs) ⇒
  (M,Oi,Os) sat prop NONE
```

[Carol_privcmd_trapped_thm]

```
⊢ ∀ NS Out M Oi Os.
    TR (M,Oi,Os) (trap (PR privcmd))
      (CFG inputOK2 SM0StateInterp (certs2 cmd npriv privcmd)
         (Name Carol says prop (SOME (PR privcmd))::ins) s
```

$outs$)
(CFG inputOK2 SM0StateInterp (certs2 cmd $npriv$ $privcmd$)
ins (NS s (trap (PR $privcmd$)))
(Out s (trap (PR $privcmd$))::$outs$)) ⇒
(M,Oi,Os) sat prop NONE

[Carol_trap_privcmd_justified_thm]

⊢ ∀ NS Out M Oi Os.
TR (M,Oi,Os) (trap (PR $privcmd$))
(CFG inputOK2 SM0StateInterp (certs2 cmd $npriv$ $privcmd$)
(Name Carol says prop (SOME (PR $privcmd$))::ins) s
$outs$)
(CFG inputOK2 SM0StateInterp (certs2 cmd $npriv$ $privcmd$)
ins (NS s (trap (PR $privcmd$)))
(Out s (trap (PR $privcmd$))::$outs$)) ⟺
inputOK2 (Name Carol says prop (SOME (PR $privcmd$))) ∧
CFGInterpret (M,Oi,Os)
(CFG inputOK2 SM0StateInterp (certs2 cmd $npriv$ $privcmd$)
(Name Carol says prop (SOME (PR $privcmd$))::ins) s
$outs$) ∧ (M,Oi,Os) sat prop NONE

[inputOK2_def]

⊢ (inputOK2 (Name Carol says prop (SOME cmd)) ⟺ T) ∧
(inputOK2 TT ⟺ F) ∧ (inputOK2 FF ⟺ F) ∧
(inputOK2 (prop v) ⟺ F) ∧ (inputOK2 (notf v_1) ⟺ F) ∧
(inputOK2 (v_2 andf v_3) ⟺ F) ∧ (inputOK2 (v_4 orf v_5) ⟺ F) ∧
(inputOK2 (v_6 impf v_7) ⟺ F) ∧ (inputOK2 (v_8 eqf v_9) ⟺ F) ∧
(inputOK2 (v_{10} says TT) ⟺ F) ∧
(inputOK2 (v_{10} says FF) ⟺ F) ∧
(inputOK2 (Name Alice says prop (SOME $v142$)) ⟺ F) ∧
(inputOK2 (Name Bob says prop (SOME $v142$)) ⟺ F) ∧
(inputOK2 (Name $v132$ says prop NONE) ⟺ F) ∧
(inputOK2 ($v133$ meet $v134$ says prop v_{66}) ⟺ F) ∧
(inputOK2 ($v135$ quoting $v136$ says prop v_{66}) ⟺ F) ∧
(inputOK2 (v_{10} says notf v_{67}) ⟺ F) ∧
(inputOK2 (v_{10} says (v_{68} andf v_{69})) ⟺ F) ∧
(inputOK2 (v_{10} says (v_{70} orf v_{71})) ⟺ F) ∧
(inputOK2 (v_{10} says (v_{72} impf v_{73})) ⟺ F) ∧
(inputOK2 (v_{10} says (v_{74} eqf v_{75})) ⟺ F) ∧
(inputOK2 (v_{10} says v_{76} says v_{77}) ⟺ F) ∧
(inputOK2 (v_{10} says v_{78} speaks_for v_{79}) ⟺ F) ∧
(inputOK2 (v_{10} says v_{80} controls v_{81}) ⟺ F) ∧
(inputOK2 (v_{10} says reps v_{82} v_{83} v_{84}) ⟺ F) ∧
(inputOK2 (v_{10} says v_{85} domi v_{86}) ⟺ F) ∧
(inputOK2 (v_{10} says v_{87} eqi v_{88}) ⟺ F) ∧

(inputOK2 (v_{10} says v_{89} doms v_{90}) $\iff$ F) $\wedge$
(inputOK2 (v_{10} says v_{91} eqs v_{92}) $\iff$ F) $\wedge$
(inputOK2 (v_{10} says v_{93} eqn v_{94}) $\iff$ F) $\wedge$
(inputOK2 (v_{10} says v_{95} lte v_{96}) $\iff$ F) $\wedge$
(inputOK2 (v_{10} says v_{97} lt v_{98}) $\iff$ F) $\wedge$
(inputOK2 (v_{12} speaks_for v_{13}) $\iff$ F) $\wedge$
(inputOK2 (v_{14} controls v_{15}) $\iff$ F) $\wedge$
(inputOK2 (reps v_{16} v_{17} v_{18}) $\iff$ F) $\wedge$
(inputOK2 (v_{19} domi v_{20}) $\iff$ F) $\wedge$
(inputOK2 (v_{21} eqi v_{22}) $\iff$ F) $\wedge$
(inputOK2 (v_{23} doms v_{24}) $\iff$ F) $\wedge$
(inputOK2 (v_{25} eqs v_{26}) $\iff$ F) $\wedge$
(inputOK2 (v_{27} eqn v_{28}) $\iff$ F) $\wedge$
(inputOK2 (v_{29} lte v_{30}) $\iff$ F) $\wedge$ (inputOK2 (v_{31} lt v_{32}) $\iff$ F)

[inputOK2_ind]

$\vdash \forall P.$
(∀ *cmd*. P (Name Carol says prop (SOME *cmd*))) $\wedge$ P TT $\wedge$ P FF $\wedge$
(∀ v. P (prop v)) $\wedge$ (∀ v_1. P (notf v_1)) $\wedge$
(∀ v_2 v_3. P (v_2 andf v_3)) $\wedge$ (∀ v_4 v_5. P (v_4 orf v_5)) $\wedge$
(∀ v_6 v_7. P (v_6 impf v_7)) $\wedge$ (∀ v_8 v_9. P (v_8 eqf v_9)) $\wedge$
(∀ v_{10}. P (v_{10} says TT)) $\wedge$ (∀ v_{10}. P (v_{10} says FF)) $\wedge$
(∀ *v142*. P (Name Alice says prop (SOME *v142*))) $\wedge$
(∀ *v142*. P (Name Bob says prop (SOME *v142*))) $\wedge$
(∀ *v132*. P (Name *v132* says prop NONE)) $\wedge$
(∀ *v133* *v134* v_{66}. P (*v133* meet *v134* says prop v_{66})) $\wedge$
(∀ *v135* *v136* v_{66}. P (*v135* quoting *v136* says prop v_{66})) $\wedge$
(∀ v_{10} v_{67}. P (v_{10} says notf v_{67})) $\wedge$
(∀ v_{10} v_{68} v_{69}. P (v_{10} says (v_{68} andf v_{69}))) $\wedge$
(∀ v_{10} v_{70} v_{71}. P (v_{10} says (v_{70} orf v_{71}))) $\wedge$
(∀ v_{10} v_{72} v_{73}. P (v_{10} says (v_{72} impf v_{73}))) $\wedge$
(∀ v_{10} v_{74} v_{75}. P (v_{10} says (v_{74} eqf v_{75}))) $\wedge$
(∀ v_{10} v_{76} v_{77}. P (v_{10} says v_{76} says v_{77})) $\wedge$
(∀ v_{10} v_{78} v_{79}. P (v_{10} says v_{78} speaks_for v_{79})) $\wedge$
(∀ v_{10} v_{80} v_{81}. P (v_{10} says v_{80} controls v_{81})) $\wedge$
(∀ v_{10} v_{82} v_{83} v_{84}. P (v_{10} says reps v_{82} v_{83} v_{84})) $\wedge$
(∀ v_{10} v_{85} v_{86}. P (v_{10} says v_{85} domi v_{86})) $\wedge$
(∀ v_{10} v_{87} v_{88}. P (v_{10} says v_{87} eqi v_{88})) $\wedge$
(∀ v_{10} v_{89} v_{90}. P (v_{10} says v_{89} doms v_{90})) $\wedge$
(∀ v_{10} v_{91} v_{92}. P (v_{10} says v_{91} eqs v_{92})) $\wedge$
(∀ v_{10} v_{93} v_{94}. P (v_{10} says v_{93} eqn v_{94})) $\wedge$
(∀ v_{10} v_{95} v_{96}. P (v_{10} says v_{95} lte v_{96})) $\wedge$
(∀ v_{10} v_{97} v_{98}. P (v_{10} says v_{97} lt v_{98})) $\wedge$
(∀ v_{12} v_{13}. P (v_{12} speaks_for v_{13})) $\wedge$
(∀ v_{14} v_{15}. P (v_{14} controls v_{15})) $\wedge$

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

2 SM0 Theory

Built: 09 April 2019
Parent Theories: ssm1

2.1 Datatypes

command = NP npriv | PR privcmd

npriv = status

output = on | off

privcmd = launch | reset

staff = Alice | Bob | Carol

state = STBY | ACTIVE

2.2 Definitions

[certs_def]

$\vdash \forall\, cmd\ npriv\ privcmd.$
certs *cmd npriv privcmd* =
[Name Alice controls prop (SOME (NP *npriv*));
Name Alice controls prop (SOME (PR *privcmd*));
Name Bob controls prop (SOME (NP *npriv*));
Name Bob says prop (SOME (PR *privcmd*)) impf prop NONE]

[SM0StateInterp_def]

$\vdash \forall\, state.$ SM0StateInterp *state* = TT

2.3 Theorems

[Alice_exec_privcmd_justified_thm]

```
⊢ ∀ NS Out M Oi Os.
    TR (M,Oi,Os) (exec (PR privcmd))
      (CFG inputOK SMOStateInterp (certs cmd npriv privcmd)
         (Name Alice says prop (SOME (PR privcmd))::ins) s
         outs)
      (CFG inputOK SMOStateInterp (certs cmd npriv privcmd) ins
         (NS s (exec (PR privcmd)))
         (Out s (exec (PR privcmd))::outs)) ⟺
    inputOK (Name Alice says prop (SOME (PR privcmd))) ∧
    CFGInterpret (M,Oi,Os)
      (CFG inputOK SMOStateInterp (certs cmd npriv privcmd)
         (Name Alice says prop (SOME (PR privcmd))::ins) s
         outs) ∧ (M,Oi,Os) sat prop (SOME (PR privcmd))
```

[Alice_justified_privcmd_exec_thm]

```
⊢ ∀ NS Out M Oi Os cmd npriv privcmd ins s outs.
    inputOK (Name Alice says prop (SOME (PR privcmd))) ∧
    CFGInterpret (M,Oi,Os)
      (CFG inputOK SMOStateInterp (certs cmd npriv privcmd)
         (Name Alice says prop (SOME (PR privcmd))::ins) s
         outs) ⇒
    TR (M,Oi,Os) (exec (PR privcmd))
      (CFG inputOK SMOStateInterp (certs cmd npriv privcmd)
         (Name Alice says prop (SOME (PR privcmd))::ins) s
         outs)
      (CFG inputOK SMOStateInterp (certs cmd npriv privcmd) ins
         (NS s (exec (PR privcmd)))
         (Out s (exec (PR privcmd))::outs))
```

[Alice_privcmd_lemma]

```
⊢ CFGInterpret (M,Oi,Os)
    (CFG inputOK SMOStateInterp (certs cmd npriv privcmd)
       (Name Alice says prop (SOME (PR privcmd))::ins) s
       outs) ⇒
  (M,Oi,Os) sat prop (SOME (PR privcmd))
```

[Alice_privcmd_verified_thm]

```
⊢ ∀ NS Out M Oi Os.
    TR (M,Oi,Os) (exec (PR privcmd))
      (CFG inputOK SMOStateInterp (certs cmd npriv privcmd)
         (Name Alice says prop (SOME (PR privcmd))::ins) s
```

outs)
(CFG inputOK SMOStateInterp (certs *cmd npriv privcmd*) *ins*
(*NS s* (exec (PR *privcmd*)))
(*Out s* (exec (PR *privcmd*))::*outs*)) ⇒
(*M*,*Oi*,*Os*) sat prop (SOME (PR *privcmd*))

[Carol_discard_lemma]

⊢ TR (*M*,*Oi*,*Os*) discard
(CFG inputOK SMOStateInterp (certs *cmd npriv privcmd*)
(Name Carol says prop (SOME *cmd*)::*ins*) *s outs*)
(CFG inputOK SMOStateInterp (certs *cmd npriv privcmd*) *ins*
(SMOns *s* discard) (SMOout *s* discard::*outs*))

[Carol_rejected_lemma]

⊢ ¬inputOK (Name Carol says prop (SOME *cmd*))

[command_distinct_clauses]

⊢ ∀ *a′ a*. NP *a* ≠ PR *a′*

[command_one_one]

⊢ (∀ *a a′*. (NP *a* = NP *a′*) ⟺ (*a* = *a′*)) ∧
∀ *a a′*. (PR *a* = PR *a′*) ⟺ (*a* = *a′*)

[inputOK_def]

⊢ (inputOK (Name Alice says prop (SOME *cmd*)) ⟺ T) ∧
(inputOK (Name Bob says prop (SOME *cmd*)) ⟺ T) ∧
(inputOK TT ⟺ F) ∧ (inputOK FF ⟺ F) ∧
(inputOK (prop *v*) ⟺ F) ∧ (inputOK (notf v_1) ⟺ F) ∧
(inputOK (v_2 andf v_3) ⟺ F) ∧ (inputOK (v_4 orf v_5) ⟺ F) ∧
(inputOK (v_6 impf v_7) ⟺ F) ∧ (inputOK (v_8 eqf v_9) ⟺ F) ∧
(inputOK (v_{10} says TT) ⟺ F) ∧ (inputOK (v_{10} says FF) ⟺ F) ∧
(inputOK (Name Carol says prop (SOME *v142*)) ⟺ F) ∧
(inputOK (Name *v132* says prop NONE) ⟺ F) ∧
(inputOK (*v133* meet *v134* says prop v_{66}) ⟺ F) ∧
(inputOK (*v135* quoting *v136* says prop v_{66}) ⟺ F) ∧
(inputOK (v_{10} says notf v_{67}) ⟺ F) ∧
(inputOK (v_{10} says (v_{68} andf v_{69})) ⟺ F) ∧
(inputOK (v_{10} says (v_{70} orf v_{71})) ⟺ F) ∧
(inputOK (v_{10} says (v_{72} impf v_{73})) ⟺ F) ∧
(inputOK (v_{10} says (v_{74} eqf v_{75})) ⟺ F) ∧
(inputOK (v_{10} says v_{76} says v_{77}) ⟺ F) ∧
(inputOK (v_{10} says v_{78} speaks_for v_{79}) ⟺ F) ∧
(inputOK (v_{10} says v_{80} controls v_{81}) ⟺ F) ∧
(inputOK (v_{10} says reps v_{82} v_{83} v_{84}) ⟺ F) ∧

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

[inputOK_ind]

$\vdash \forall P.$
(∀ *cmd*. P (Name Alice says prop (SOME *cmd*))) $\wedge$
(∀ *cmd*. P (Name Bob says prop (SOME *cmd*))) $\wedge$ P TT $\wedge$ P FF $\wedge$
(∀ v. P (prop v)) $\wedge$ (∀ v_1. P (notf v_1)) $\wedge$
(∀ v_2 v_3. P (v_2 andf v_3)) $\wedge$ (∀ v_4 v_5. P (v_4 orf v_5)) $\wedge$
(∀ v_6 v_7. P (v_6 impf v_7)) $\wedge$ (∀ v_8 v_9. P (v_8 eqf v_9)) $\wedge$
(∀ v_{10}. P (v_{10} says TT)) $\wedge$ (∀ v_{10}. P (v_{10} says FF)) $\wedge$
(∀ *v142*. P (Name Carol says prop (SOME *v142*))) $\wedge$
(∀ *v132*. P (Name *v132* says prop NONE)) $\wedge$
(∀ *v133* *v134* v_{66}. P (*v133* meet *v134* says prop v_{66})) $\wedge$
(∀ *v135* *v136* v_{66}. P (*v135* quoting *v136* says prop v_{66})) $\wedge$
(∀ v_{10} v_{67}. P (v_{10} says notf v_{67})) $\wedge$
(∀ v_{10} v_{68} v_{69}. P (v_{10} says (v_{68} andf v_{69}))) $\wedge$
(∀ v_{10} v_{70} v_{71}. P (v_{10} says (v_{70} orf v_{71}))) $\wedge$
(∀ v_{10} v_{72} v_{73}. P (v_{10} says (v_{72} impf v_{73}))) $\wedge$
(∀ v_{10} v_{74} v_{75}. P (v_{10} says (v_{74} eqf v_{75}))) $\wedge$
(∀ v_{10} v_{76} v_{77}. P (v_{10} says v_{76} says v_{77})) $\wedge$
(∀ v_{10} v_{78} v_{79}. P (v_{10} says v_{78} speaks_for v_{79})) $\wedge$
(∀ v_{10} v_{80} v_{81}. P (v_{10} says v_{80} controls v_{81})) $\wedge$
(∀ v_{10} v_{82} v_{83} v_{84}. P (v_{10} says reps v_{82} v_{83} v_{84})) $\wedge$
(∀ v_{10} v_{85} v_{86}. P (v_{10} says v_{85} domi v_{86})) $\wedge$
(∀ v_{10} v_{87} v_{88}. P (v_{10} says v_{87} eqi v_{88})) $\wedge$
(∀ v_{10} v_{89} v_{90}. P (v_{10} says v_{89} doms v_{90})) $\wedge$
(∀ v_{10} v_{91} v_{92}. P (v_{10} says v_{91} eqs v_{92})) $\wedge$
(∀ v_{10} v_{93} v_{94}. P (v_{10} says v_{93} eqn v_{94})) $\wedge$
(∀ v_{10} v_{95} v_{96}. P (v_{10} says v_{95} lte v_{96})) $\wedge$
(∀ v_{10} v_{97} v_{98}. P (v_{10} says v_{97} lt v_{98})) $\wedge$
(∀ v_{12} v_{13}. P (v_{12} speaks_for v_{13})) $\wedge$

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

[output_distinct_clauses]

$\vdash$ on $\neq$ off

[privcmd_distinct_clauses]

$\vdash$ launch $\neq$ reset

[SM0ns_def]

```
⊢ (SM0ns STBY (exec (PR reset)) = STBY) ∧
  (SM0ns STBY (exec (PR launch)) = ACTIVE) ∧
  (SM0ns STBY (exec (NP status)) = STBY) ∧
  (SM0ns ACTIVE (exec (PR reset)) = STBY) ∧
  (SM0ns ACTIVE (exec (PR launch)) = ACTIVE) ∧
  (SM0ns ACTIVE (exec (NP status)) = ACTIVE) ∧
  (SM0ns STBY (trap (PR reset)) = STBY) ∧
  (SM0ns STBY (trap (PR launch)) = STBY) ∧
  (SM0ns STBY (trap (NP status)) = STBY) ∧
  (SM0ns ACTIVE (trap (PR reset)) = ACTIVE) ∧
  (SM0ns ACTIVE (trap (PR launch)) = ACTIVE) ∧
  (SM0ns ACTIVE (trap (NP status)) = ACTIVE) ∧
  (SM0ns STBY discard = STBY) ∧ (SM0ns ACTIVE discard = ACTIVE)
```

[SM0ns_ind]

$\vdash \forall\, P.$
P STBY (exec (PR reset)) $\wedge$ P STBY (exec (PR launch)) $\wedge$
P STBY (exec (NP status)) $\wedge$ P ACTIVE (exec (PR reset)) $\wedge$
P ACTIVE (exec (PR launch)) $\wedge$ P ACTIVE (exec (NP status)) $\wedge$
P STBY (trap (PR reset)) $\wedge$ P STBY (trap (PR launch)) $\wedge$
P STBY (trap (NP status)) $\wedge$ P ACTIVE (trap (PR reset)) $\wedge$
P ACTIVE (trap (PR launch)) $\wedge$ P ACTIVE (trap (NP status)) $\wedge$
P STBY discard $\wedge$ P ACTIVE discard $\Rightarrow$
$\forall\, v\ v_1.\ P\ v\ v_1$

[SM0out_def]

```
⊢ (SM0out STBY (exec (PR reset)) = off) ∧
  (SM0out STBY (exec (PR launch)) = on) ∧
  (SM0out STBY (exec (NP status)) = off) ∧
  (SM0out ACTIVE (exec (PR reset)) = off) ∧
  (SM0out ACTIVE (exec (PR launch)) = on) ∧
  (SM0out ACTIVE (exec (NP status)) = on) ∧
  (SM0out STBY (trap (PR reset)) = off) ∧
  (SM0out STBY (trap (PR launch)) = off) ∧
  (SM0out STBY (trap (NP status)) = off) ∧
  (SM0out ACTIVE (trap (PR reset)) = on) ∧
  (SM0out ACTIVE (trap (PR launch)) = on) ∧
  (SM0out ACTIVE (trap (NP status)) = on) ∧
  (SM0out STBY discard = off) ∧ (SM0out ACTIVE discard = on)
```

[SM0out_ind]

```
⊢ ∀P.
    P STBY (exec (PR reset)) ∧ P STBY (exec (PR launch)) ∧
    P STBY (exec (NP status)) ∧ P ACTIVE (exec (PR reset)) ∧
    P ACTIVE (exec (PR launch)) ∧ P ACTIVE (exec (NP status)) ∧
    P STBY (trap (PR reset)) ∧ P STBY (trap (PR launch)) ∧
    P STBY (trap (NP status)) ∧ P ACTIVE (trap (PR reset)) ∧
    P ACTIVE (trap (PR launch)) ∧ P ACTIVE (trap (NP status)) ∧
    P STBY discard ∧ P ACTIVE discard ⇒
    ∀v v1. P v v1
```

[staff_distinct_clauses]

```
⊢ Alice ≠ Bob ∧ Alice ≠ Carol ∧ Bob ≠ Carol
```

[state_distinct_clauses]

```
⊢ STBY ≠ ACTIVE
```

Index

SM0 Theory, 8
  Datatypes, 8
  Definitions, 8
    certs_def, 8
    SM0StateInterp_def, 8
  Theorems, 9
    Alice_exec_privcmd_justified_thm, 9
    Alice_justified_privcmd_exec_thm, 9
    Alice_privcmd_lemma, 9
    Alice_privcmd_verified_thm, 9
    Carol_discard_lemma, 10
    Carol_rejected_lemma, 10
    command_distinct_clauses, 10
    command_one_one, 10
    inputOK_def, 10
    inputOK_ind, 11
    output_distinct_clauses, 12
    privcmd_distinct_clauses, 12
    SM0ns_def, 12
    SM0ns_ind, 12
    SM0out_def, 13
    SM0out_ind, 13
    staff_distinct_clauses, 13
    state_distinct_clauses, 13
SM0Solutions Theory, 3
  Definitions, 3
    certs2_def, 3
  Theorems, 3
    Alice_exec_npriv_justified_thm, 3
    Alice_justified_npriv_exec_thm, 3
    Alice_npriv_lemma, 4
    Alice_npriv_verified_thm, 4
    Carol_exec_npriv_justified_thm, 4
    Carol_justified_npriv_exec_thm, 4
    Carol_justified_privcmd_trap_thm, 5
    Carol_npriv_lemma, 5
    Carol_npriv_verified_thm, 5
    Carol_privcmd_trap_lemma, 5
    Carol_privcmd_trapped_thm, 5
    Carol_trap_privcmd_justified_thm, 6
    inputOK2_def, 6
    inputOK2_ind, 7