# Contents

# 1 exam4 Theory

**Built:** 27 March 2020
**Parent Theories:** aclDrules

## 1.1 Datatypes

*access* = print | read | write

*keys* = KTGS | KFS | KU | KUTGS | KUFS

*people* = Ursala

*princs* = PR servers | User people | Key keys

*props* = USE serviceKey | PK people keys | AC access

*servers* = AS | TGS | FS

*serviceKey* = SERV servers | K keys

## 1.2 Theorems

[init_auth_thm]

```
⊢ (M, Oi, Os) sat Name (User Ursala) says prop (USE (SERV TGS)) ⇒
  (M, Oi, Os) sat
  prop (USE (SERV TGS)) impf
  Name (Key KU) says prop (USE (K KUTGS)) andf
  Name (Key KTGS) says prop (PK Ursala KUTGS) ⇒
  (M, Oi, Os) sat
  Name (User Ursala) controls prop (USE (SERV TGS)) ⇒
  (M, Oi, Os) sat
  Name (Key KU) says prop (USE (K KUTGS)) andf
  Name (Key KTGS) says prop (PK Ursala KUTGS)
```

[request_for_services_thm]

```
⊢ (M, Oi, Os) sat Name (Key KTGS) speaks_for Name (PR AS) ⇒
  (M, Oi, Os) sat
  prop (PK Ursala KUTGS) impf
  Name (Key KUFS) says prop (AC read) ⇒
  (M, Oi, Os) sat
  prop (PK Ursala KUTGS) impf
  Name (Key KFS) says
  Name (Key KUFS) speaks_for Name (User Ursala) ⇒
  (M, Oi, Os) sat Name (PR AS) controls prop (PK Ursala KUTGS) ⇒
  (M, Oi, Os) sat
  Name (Key KUTGS) says prop (USE (SERV FS)) andf
  Name (Key KTGS) says prop (PK Ursala KUTGS) ⇒
```

```
   (M, Oi, Os) sat
   Name (Key KFS) says
   Name (Key KUFS) speaks_for Name (User Ursala) andf
   Name (Key KUFS) says prop (AC read)
```

[service_request_thm]

```
 ⊢ (M, Oi, Os) sat Name (Key KFS) speaks_for Name (PR TGS) ⇒
   (M, Oi, Os) sat Name (User Ursala) controls prop (AC print) ⇒
   (M, Oi, Os) sat
   Name (PR TGS) controls
   Name (Key KUFS) speaks_for Name (User Ursala) ⇒
   (M, Oi, Os) sat
   Name (Key KFS) says
   Name (Key KUFS) speaks_for Name (User Ursala) andf
   Name (Key KUFS) says prop (AC print) ⇒
   (M, Oi, Os) sat prop (AC print)
```

[session_key_receipt_thm]

```
 ⊢ (M, Oi, Os) sat Name (Key KU) speaks_for Name (PR AS) ⇒
   (M, Oi, Os) sat
   prop (USE (K KUTGS)) impf
   Name (Key KUTGS) says prop (USE (SERV FS)) ⇒
   (M, Oi, Os) sat Name (PR AS) controls prop (USE (K KUTGS)) ⇒
   (M, Oi, Os) sat
   Name (Key KU) says prop (USE (K KUTGS)) andf
   Name (Key KTGS) says prop (PK Ursala KUTGS) ⇒
   (M, Oi, Os) sat
   Name (Key KUTGS) says prop (USE (SERV FS)) andf
   Name (Key KTGS) says prop (PK Ursala KUTGS)
```

# 2 simpleOpener Theory

**Built:** 27 March 2020
**Parent Theories:** sm

## 2.1 Datatypes

*command* = i0 | i1

*output* = o0 | o1

*state* = S0 | S1

## 2.2 Theorems

[command_distinct_clauses]

```
 ⊢ i0 ≠ i1
```

[output_distinct_clauses]

$\vdash$ o0 $\neq$ o1

[simpleCounter_rules]

$\vdash$ ($\forall$ *ins outs*.
TR i1 (CFG (i1::*ins*) S0 *outs*) (CFG *ins* S1 (o1::*outs*))) $\wedge$
$\forall$ *ins outs*.
TR i0 (CFG (i0::*ins*) S1 *outs*) (CFG *ins* S0 (o0::*outs*))

[simpleOpenerns_def]

$\vdash$ (simpleOpenerns S0 i1 = S1) $\wedge$ (simpleOpenerns S1 i0 = S0)

[simpleOpenerns_ind]

$\vdash$ $\forall P$. $P$ S0 i1 $\wedge$ $P$ S1 i0 $\wedge$ $P$ S0 i0 $\wedge$ $P$ S1 i1 $\Rightarrow$ $\forall v\ v_1$. $P\ v\ v_1$

[simpleOpenerout_def]

$\vdash$ (simpleOpenerout S0 i1 = o1) $\wedge$ (simpleOpenerout S1 i0 = o0)

[simpleOpenerout_ind]

$\vdash$ $\forall P$. $P$ S0 i1 $\wedge$ $P$ S1 i0 $\wedge$ $P$ S0 i0 $\wedge$ $P$ S1 i1 $\Rightarrow$ $\forall v\ v_1$. $P\ v\ v_1$

[simpleOpenerTR_clauses]

$\vdash$ ($\forall x\ x1s\ s_1\ out1s\ x2s\ out2s\ s_2$.
TR $x$ (CFG $x1s\ s_1\ out1s$) (CFG $x2s\ s_2\ out2s$) $\iff$
$\exists NS\ Out\ ins$.
($x1s$ = $x$::$ins$) $\wedge$ ($x2s$ = $ins$) $\wedge$ ($s_2$ = $NS\ s_1\ x$) $\wedge$
($out2s$ = $Out\ s_1\ x$::$out1s$)) $\wedge$
$\forall x\ x1s\ s_1\ out1s\ x2s\ out2s$.
TR $x$ (CFG $x1s\ s_1\ out1s$)
(CFG $x2s$ (simpleOpenerns $s_1\ x$)
(simpleOpenerout $s_1\ x$::$out2s$)) $\iff$
$\exists ins$. ($x1s$ = $x$::$ins$) $\wedge$ ($x2s$ = $ins$) $\wedge$ ($out2s$ = $out1s$)

[simpleOpenerTR_rules]

$\vdash$ $\forall s\ x\ ins\ outs$.
TR $x$ (CFG ($x$::$ins$) $s\ outs$)
(CFG $ins$ (simpleOpenerns $s\ x$)
(simpleOpenerout $s\ x$::$outs$))

[simpleOpenerTrans_Equiv_TR]

$\vdash$ TR $x$ (CFG ($x$::$ins$) $s\ outs$)
(CFG $ins$ (simpleOpenerns $s\ x$)
(simpleOpenerout $s\ x$::$outs$)) $\iff$
Trans $x\ s$ (simpleOpenerns $s\ x$)

[state_distinct_clauses]

$\vdash$ S0 $\neq$ S1

# Index