

Broken High Watermark Specification

George W. Dinolt*

August 19, 2004

1 Introduction

Below we present a (incomplete) specification of the *High Watermark* security model. We also present a proof of one of the lemmas. The model is incomplete in that the specifications of *transform* and *st?* are “too strong” and don’t really capture the intent of the model.

2 The Broken High Watermark Specification

I have included the specification that we are using below. You will notice that it is written in a more “mathematical” style. You should be able to interpret it and map it directly to the PVS.

You may notice that one of the \wedge ’s in the definition of *transform* is red. It is in this location that the definitions are broken. We will see why as we study the proof of *transform-secure*.

The specification appears below.

*Naval Postgraduate School

```

HighWaterMark[Subjects: TYPE+, Objects: TYPE+, Labels:
              TYPE FROM nat]: THEORY
BEGIN

lb(ob: Objects): Labels

Mode: TYPE = {read, write}

Accesses: TYPE =
[# s: Subjects,
   ob: Objects,
   m: Mode,
   seqnumber: nat #]

State: TYPE = setof[Accesses]

st: VAR State

st?(st): bool =

$$\forall (xa, xb: Accesses): st(xa) \wedge st(xb) \Rightarrow$$


$$\text{IF } xa^{\prime}\text{seqnumber} = xb^{\prime}\text{seqnumber}$$


$$\quad \text{THEN } xa = xb$$


$$\quad \text{ELSE } xa^{\prime}s = xb^{\prime}s \wedge$$


$$\quad xa^{\prime}\text{seqnumber} < xb^{\prime}\text{seqnumber} \wedge xb^{\prime}m = \text{write}$$


$$\Rightarrow lb(xa^{\prime}ob) \leq lb(xb^{\prime}ob)$$

ENDIF

transform(a: Accesses, st: State): State =

$$\text{IF } \forall (e: Accesses): st(e) \wedge e^{\prime}\text{seqnumber} < a^{\prime}\text{seqnumber}$$


$$\text{THEN COND } (\neg (\exists (x: Accesses): st(x) \wedge x^{\prime}s = a^{\prime}s)) \rightarrow$$


$$\quad (st \cup \{a\}),$$


$$\quad a^{\prime}m = \text{read} \rightarrow (st \cup \{a\}),$$


$$\quad (\forall (y: Accesses):$$


$$\quad \quad st(y) \wedge$$


$$\quad \quad y^{\prime}s = a^{\prime}s \wedge$$


$$\quad \quad a^{\prime}m = \text{write} \wedge lb(a^{\prime}ob) \geq lb(y^{\prime}ob))$$


$$\quad \rightarrow (st \cup \{a\}),$$


$$\quad \text{ELSE } \rightarrow st$$


$$\quad \text{ENDCOND}$$


$$\text{ELSE } st$$

ENDIF

```

```

transform_secure: LEMMA
   $\forall (a: \text{Accesses}, st: \text{State}):$ 
     $\text{st?}(st) \Rightarrow \text{st?}(\text{transform}(a, st))$ 

empty_is_secure: LEMMA  $\text{st?}(\text{emptyset})$ 

noWriteDownSubject: LEMMA
   $\forall (st: \text{State}, a, b: \text{Accesses}):$ 
     $\text{st?}(st) \wedge$ 
     $(a \in st) \wedge$ 
     $(b \in st) \wedge$ 
     $a^{\prime}m = \text{read} \wedge$ 
     $b^{\prime}m = \text{write} \wedge a^{\prime}s = b^{\prime}s \wedge a^{\prime}\text{seqnumber} < b^{\prime}\text{seqnumber}$ 
     $\Rightarrow \text{lb}(a^{\prime}\text{ob}) \leq \text{lb}(b^{\prime}\text{ob})$ 

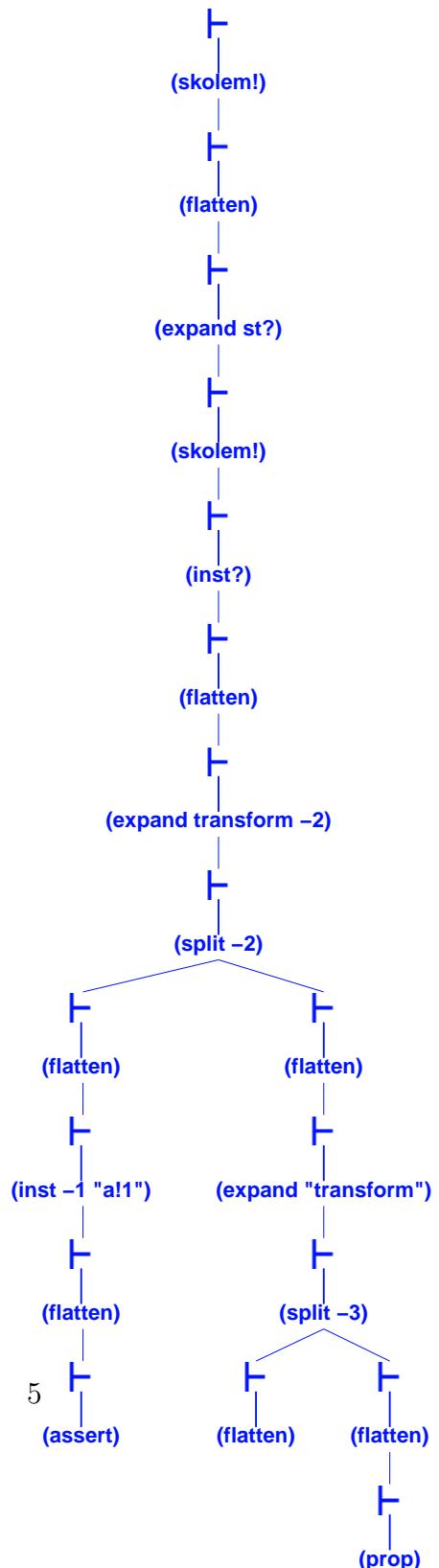
END HighWaterMark

```

3 Proof of the *Transform_Secure* Lemma

3.1 Graph of the Proof

We present the proof of *transform_secure* in two forms. The first is a graph of the sequence of **pvs** steps that were taken. This should give you a quick glimpse of the proof. You will notice that this graph is relatively small. You can use this graph to help you follow the proof



3.2 Proof Details

Below we provide the details of the proof. On page 12 you can find hypothesis {-1} of the form

$$\forall (e: \text{Accesses}) : \text{st}'(e) \wedge e.\text{seqnumber} < a'.\text{seqnumber}$$

Note the form of this hypothesis. By substituting a' in for e we get an obvious contradiction. This clearly is an issue. If you look back to the location on page 2 you can see that the specification of *transform* has this same condition.¹ To fix the specification, we should replace the \wedge with an \Rightarrow .

The condition we are trying to make happen at this point in the spec is that if all the elements that are in the state have a sequence number less than the sequence number of the new element, then we may be able to add the new element, provided that the access is *read* or that the labels have the correct relationship.

You should also note the order of the expands of the *transform* functions (and the order of all the expansions). Since we didn't expand everything at once, we were better able to handle several cases at once.

¹If you look on the previous page of the proof, you will see the completee *if* conditions specified.

Verbose proof for `transform_secure`.

`transform_secure`:

$$\boxed{\{1\} \quad \forall (a: \text{Accesses}, st: \text{State}) : \text{st?}(st) \Rightarrow \text{st?}(\text{transform}(a, st))}$$

`transform_secure`:

$$\boxed{\{1\} \quad \forall (a: \text{Accesses}, st: \text{State}) : \text{st?}(st) \Rightarrow \text{st?}(\text{transform}(a, st))}$$

Skolemizing,

`transform_secure`:

$$\boxed{\{1\} \quad \text{st?}(st') \Rightarrow \text{st?}(\text{transform}(a', st'))}$$

Applying disjunctive simplification to flatten sequent,

`transform_secure`:

$$\boxed{\begin{array}{l} \{-1\} \quad \text{st?}(st') \\ \{1\} \quad \text{st?}(\text{transform}(a', st')) \end{array}}$$

Expanding the definition of `st?`,

`transform_secure`:

$$\boxed{\begin{array}{l} \{-1\} \quad \forall (xa, xb: \text{Accesses}) : \\ \quad \text{st}'(xa) \wedge \text{st}'(xb) \Rightarrow \\ \quad \text{IF } xa' \text{seqnumber} = xb' \text{seqnumber} \\ \quad \quad \text{THEN } xa = xb \\ \quad \quad \text{ELSE } xa's = xb's \wedge \\ \quad \quad \quad xa' \text{seqnumber} < xb' \text{seqnumber} \wedge xb'm = \text{write} \\ \quad \quad \Rightarrow \text{lb}(xa'ob) \leq \text{lb}(xb'ob) \\ \quad \text{ENDIF} \end{array}}$$

$$\boxed{\begin{array}{l} \{1\} \quad \forall (xa, xb: \text{Accesses}) : \\ \quad \text{transform}(a', st')(xa) \wedge \text{transform}(a', st')(xb) \Rightarrow \\ \quad \text{IF } xa' \text{seqnumber} = xb' \text{seqnumber} \\ \quad \quad \text{THEN } xa = xb \\ \quad \quad \text{ELSE } xa's = xb's \wedge \\ \quad \quad \quad xa' \text{seqnumber} < xb' \text{seqnumber} \wedge xb'm = \text{write} \\ \quad \quad \Rightarrow \text{lb}(xa'ob) \leq \text{lb}(xb'ob) \\ \quad \text{ENDIF} \end{array}}$$

Skolemizing,

`transform_secure:`

$\{-1\} \quad \forall (xa, xb : \text{Accesses}) :$ $st'(xa) \wedge st'(xb) \Rightarrow$ $\text{IF } xa' \text{seqnumber} = xb' \text{seqnumber}$ $\quad \text{THEN } xa = xb$ $\quad \text{ELSE } xa's = xb's \wedge$ $\quad \quad xa' \text{seqnumber} < xb' \text{seqnumber} \wedge xb'm = \text{write}$ $\quad \Rightarrow lb(xa'ob) \leq lb(xb'ob)$
ENDIF

$\{1\} \quad \text{transform}(a', st')(xa') \wedge \text{transform}(a', st')(xb') \Rightarrow$ $\text{IF } xa' \text{seqnumber} = xb' \text{seqnumber}$ $\quad \text{THEN } xa' = xb'$ $\quad \text{ELSE } xa's = xb's \wedge$ $\quad \quad xa' \text{seqnumber} < xb' \text{seqnumber} \wedge xb'm = \text{write}$ $\quad \Rightarrow lb(xa'ob) \leq lb(xb'ob)$
ENDIF

Instantiating quantified variables,

`transform_secure:`

$\{-1\} \quad st'(xa') \wedge st'(xb') \Rightarrow$ $\text{IF } xa' \text{seqnumber} = xb' \text{seqnumber}$ $\quad \text{THEN } xa' = xb'$ $\quad \text{ELSE } xa's = xb's \wedge$ $\quad \quad xa' \text{seqnumber} < xb' \text{seqnumber} \wedge xb'm = \text{write}$ $\quad \Rightarrow lb(xa'ob) \leq lb(xb'ob)$
ENDIF

$\{1\} \quad \text{transform}(a', st')(xa') \wedge \text{transform}(a', st')(xb') \Rightarrow$ $\text{IF } xa' \text{seqnumber} = xb' \text{seqnumber}$ $\quad \text{THEN } xa' = xb'$ $\quad \text{ELSE } xa's = xb's \wedge$ $\quad \quad xa' \text{seqnumber} < xb' \text{seqnumber} \wedge xb'm = \text{write}$ $\quad \Rightarrow lb(xa'ob) \leq lb(xb'ob)$
ENDIF

Applying disjunctive simplification to flatten sequent,

`transform_secure:`

```
{-1} st'(xa') ∧ st'(xb') ⇒
    IF xa'‘seqnumber = xb'‘seqnumber
        THEN xa' = xb'
    ELSE xa'‘s = xb'‘s ∧
        xa'‘seqnumber < xb'‘seqnumber ∧ xb'‘m = write
        ⇒ lb(xa'‘ob) ≤ lb(xb'‘ob)
    ENDIF
{-2} transform(a', st')(xa')
{-3} transform(a', st')(xb')
```

```
{1} IF xa'‘seqnumber = xb'‘seqnumber
    THEN xa' = xb'
    ELSE xa'‘s = xb'‘s ∧
        xa'‘seqnumber < xb'‘seqnumber ∧ xb'‘m = write
        ⇒ lb(xa'‘ob) ≤ lb(xb'‘ob)
    ENDIF
```

Expanding the definition of transform,

`transform_secure:`

```

{-1}  st'(xa') ∧ st'(xb') ⇒
      IF xa'‘seqnumber = xb'‘seqnumber
          THEN xa' = xb'
      ELSE xa'‘s = xb'‘s ∧
          xa'‘seqnumber < xb'‘seqnumber ∧ xb'‘m = write
          ⇒ lb(xa'‘ob) ≤ lb(xb'‘ob)

      ENDIF
{-2}  IF ∀ (e: Accesses): st'(e) ∧ e‘seqnumber < a‘seqnumber
      THEN IF (¬ (Ǝ (x: Accesses): st'(x) ∧ x‘s = a‘s))
          THEN add(a', st')(xa')
      ELSE IF a‘m = read
          THEN add(a', st')(xa')
      ELSE IF ∀ (y: Accesses):
          st'(y) ∧
          y‘s = a‘s ∧
          a‘m = write ∧ lb(a‘ob) ≥ lb(y‘ob)
          THEN add(a', st')(xa')
      ELSE st'(xa')
      ENDIF
      ENDIF
      ELSE st'(xa')
      ENDIF
{-3}  transform(a', st')(xb')

```

```

{1}  IF xa'‘seqnumber = xb'‘seqnumber
      THEN xa' = xb'
      ELSE xa'‘s = xb'‘s ∧
          xa'‘seqnumber < xb'‘seqnumber ∧ xb'‘m = write
          ⇒ lb(xa'‘ob) ≤ lb(xb'‘ob)

      ENDIF

```

Splitting conjunctions,
we get 2 subgoals:

`transform_secure.1:`

```

{-1}  ( $\forall (e: \text{Accesses}) : st'(e) \wedge e' \text{seqnumber} < a' \text{seqnumber}$ )  $\wedge$ 
      IF ( $\neg (\exists (x: \text{Accesses}) : st'(x) \wedge x's = a's)$ )
          THEN add( $a'$ ,  $st'$ )( $xa'$ )
      ELSE IF  $a'm = \text{read}$ 
          THEN add( $a'$ ,  $st'$ )( $xa'$ )
      ELSE IF  $\forall (y: \text{Accesses}) :$ 
           $st'(y) \wedge$ 
           $y's = a's \wedge$ 
           $a'm = \text{write} \wedge \text{lb}(a'ob) \geq \text{lb}(yob)$ 
          THEN add( $a'$ ,  $st'$ )( $xa'$ )
      ELSE  $st'(xa')$ 
      ENDIF
  ENDIF
ENDIF

{-2}  $st'(xa') \wedge st'(xb') \Rightarrow$ 
      IF  $xa' \text{seqnumber} = xb' \text{seqnumber}$ 
          THEN  $xa' = xb'$ 
      ELSE  $xa's = xb's \wedge$ 
           $xa' \text{seqnumber} < xb' \text{seqnumber} \wedge xb'm = \text{write}$ 
           $\Rightarrow \text{lb}(xa'ob) \leq \text{lb}(xb'ob)$ 
      ENDIF
ENDIF

{-3} transform( $a'$ ,  $st'$ )( $xb'$ )

```

```

{1} IF  $xa' \text{seqnumber} = xb' \text{seqnumber}$ 
    THEN  $xa' = xb'$ 
ELSE  $xa's = xb's \wedge$ 
     $xa' \text{seqnumber} < xb' \text{seqnumber} \wedge xb'm = \text{write}$ 
     $\Rightarrow \text{lb}(xa'ob) \leq \text{lb}(xb'ob)$ 
ENDIF

```

Applying disjunctive simplification to flatten sequent,

`transform_secure.1:`

```

{-1}  ∀ (e: Accesses): st'(e) ∧ e'seqnumber < a'seqnumber
{-2}  IF (¬ (exists (x: Accesses): st'(x) ∧ x's = a's))
      THEN add(a', st')(xa')
      ELSE IF a'm = read
          THEN add(a', st')(xa')
          ELSE IF ∀ (y: Accesses):
              st'(y) ∧
              y's = a's ∧
              a'm = write ∧ lb(a'ob) ≥ lb(y'ob)
              THEN add(a', st')(xa')
              ELSE st'(xa')
              ENDIF
          ENDIF
      ENDIF
{-3}  st'(xa') ∧ st'(xb') ⇒
      IF xa'seqnumber = xb'seqnumber
          THEN xa' = xb'
      ELSE xa's = xb's ∧
          xa'seqnumber < xb'seqnumber ∧ xb'm = write
          ⇒ lb(xa'ob) ≤ lb(xb'ob)
      ENDIF
{-4}  transform(a', st')(xb')

```

```

{1}  IF xa'seqnumber = xb'seqnumber
      THEN xa' = xb'
      ELSE xa's = xb's ∧
          xa'seqnumber < xb'seqnumber ∧ xb'm = write
          ⇒ lb(xa'ob) ≤ lb(xb'ob)
      ENDIF

```

Instantiating the top quantifier in -1 with the terms: a' ,

`transform_secure.1:`

```

{-1}  st'(a') ∧ a'`seqnumber < a'`seqnumber
{-2}  IF (¬ (exists (x: Accesses): st'(x) ∧ x`s = a`s))
      THEN add(a', st')(xa')
      ELSE IF a`m = read
          THEN add(a', st')(xa')
          ELSE IF ∀ (y: Accesses):
              st'(y) ∧
              y`s = a`s ∧
              a`m = write ∧ lb(a`ob) ≥ lb(y`ob)
              THEN add(a', st')(xa')
              ELSE st'(xa')
              ENDIF
          ENDIF
      ENDIF
{-3}  st'(xa') ∧ st'(xb') ⇒
      IF xa`seqnumber = xb`seqnumber
          THEN xa' = xb'
      ELSE xa`s = xb`s ∧
          xa`seqnumber < xb`seqnumber ∧ xb`m = write
          ⇒ lb(xa`ob) ≤ lb(xb`ob)
      ENDIF
{-4}  transform(a', st')(xb')


---


{1}  IF xa`seqnumber = xb`seqnumber
      THEN xa' = xb'
      ELSE xa`s = xb`s ∧
          xa`seqnumber < xb`seqnumber ∧ xb`m = write
          ⇒ lb(xa`ob) ≤ lb(xb`ob)
      ENDIF

```

Applying disjunctive simplification to flatten sequent,

`transform_secure.1:`

```

{-1}  st'(a')
{-2}  a'`seqnumber < a'`seqnumber
{-3}  IF ( $\neg (\exists (x: \text{Accesses}) : st'(x) \wedge x`s = a`s)$ )
      THEN add(a', st')(xa')
      ELSE IF a`m = read
          THEN add(a', st')(xa')
          ELSE IF  $\forall (y: \text{Accesses}) :$ 
              st'(y)  $\wedge$ 
              y`s = a`s  $\wedge$ 
              a`m = write  $\wedge$  lb(a`ob)  $\geq$  lb(y`ob)
          THEN add(a', st')(xa')
          ELSE st'(xa')
          ENDIF
      ENDIF
      ENDIF
{-4}  st'(xa')  $\wedge$  st'(xb')  $\Rightarrow$ 
      IF xa'`seqnumber = xb'`seqnumber
      THEN xa' = xb'
      ELSE xa`s = xb`s  $\wedge$ 
          xa`seqnumber < xb`seqnumber  $\wedge$  xb`m = write
           $\Rightarrow$  lb(xa`ob)  $\leq$  lb(xb`ob)
      ENDIF
{-5}  transform(a', st')(xb')


---


{1}  IF xa'`seqnumber = xb'`seqnumber
      THEN xa' = xb'
      ELSE xa`s = xb`s  $\wedge$ 
          xa`seqnumber < xb`seqnumber  $\wedge$  xb`m = write
           $\Rightarrow$  lb(xa`ob)  $\leq$  lb(xb`ob)
      ENDIF

```

Simplifying, rewriting, and recording with decision procedures,
This completes the proof of `transform_secure.1`.

`transform_secure.2:`

{-1}	$\neg (\forall (e: \text{Accesses}): st'(e) \wedge e.\text{seqnumber} < a'.\text{seqnumber}) \wedge$ $st'(xa')$
{-2}	$st'(xa') \wedge st'(xb') \Rightarrow$ IF $xa'.\text{seqnumber} = xb'.\text{seqnumber}$ THEN $xa' = xb'$ ELSE $xa'.s = xb'.s \wedge$ $xa'.\text{seqnumber} < xb'.\text{seqnumber} \wedge xb'.m = \text{write}$ $\Rightarrow lb(xa'.ob) \leq lb(xb'.ob)$
{-3}	ENDIF transform($a', st')(xb')$
{1}	IF $xa'.\text{seqnumber} = xb'.\text{seqnumber}$ THEN $xa' = xb'$ ELSE $xa'.s = xb'.s \wedge$ $xa'.\text{seqnumber} < xb'.\text{seqnumber} \wedge xb'.m = \text{write}$ $\Rightarrow lb(xa'.ob) \leq lb(xb'.ob)$

Applying disjunctive simplification to flatten sequent,
`transform_secure.2:`

{-1}	$st'(xa')$
{-2}	$st'(xa') \wedge st'(xb') \Rightarrow$ IF $xa'.\text{seqnumber} = xb'.\text{seqnumber}$ THEN $xa' = xb'$ ELSE $xa'.s = xb'.s \wedge$ $xa'.\text{seqnumber} < xb'.\text{seqnumber} \wedge xb'.m = \text{write}$ $\Rightarrow lb(xa'.ob) \leq lb(xb'.ob)$
{-3}	ENDIF transform($a', st')(xb')$
{1}	$\forall (e: \text{Accesses}): st'(e) \wedge e.\text{seqnumber} < a'.\text{seqnumber}$
{2}	IF $xa'.\text{seqnumber} = xb'.\text{seqnumber}$ THEN $xa' = xb'$ ELSE $xa'.s = xb'.s \wedge$ $xa'.\text{seqnumber} < xb'.\text{seqnumber} \wedge xb'.m = \text{write}$ $\Rightarrow lb(xa'.ob) \leq lb(xb'.ob)$

Expanding the definition of transform,

`transform_secure.2:`

```

{-1}  st'(xa')
{-2}  st'(xa') ∧ st'(xb') ⇒
      IF xa'‘seqnumber = xb'‘seqnumber
          THEN xa' = xb'
      ELSE xa'‘s = xb'‘s ∧
          xa'‘seqnumber < xb'‘seqnumber ∧ xb'‘m = write
          ⇒ lb(xa'‘ob) ≤ lb(xb'‘ob)
      ENDIF
{-3}  IF ∀ (e: Accesses): st'(e) ∧ e‘seqnumber < a‘seqnumber
      THEN IF (¬ (Ǝ (x: Accesses): st'(x) ∧ x‘s = a‘s))
          THEN add(a', st')(xb')
          ELSE IF a‘m = read
              THEN add(a', st')(xb')
          ELSE IF ∀ (y: Accesses):
              st'(y) ∧
              y‘s = a‘s ∧
              a‘m = write ∧ lb(a‘ob) ≥ lb(y‘ob)
              THEN add(a', st')(xb')
              ELSE st'(xb')
              ENDIF
          ENDIF
      ENDIF
      ELSE st'(xb')
      ENDIF
{1}  ∀ (e: Accesses): st'(e) ∧ e‘seqnumber < a‘seqnumber
{2}  IF xa'‘seqnumber = xb'‘seqnumber
      THEN xa' = xb'
      ELSE xa'‘s = xb'‘s ∧
          xa'‘seqnumber < xb'‘seqnumber ∧ xb'‘m = write
          ⇒ lb(xa'‘ob) ≤ lb(xb'‘ob)
      ENDIF

```

Splitting conjunctions,
we get 2 subgoals:

`transform_secure.2.1:`

```

{-1}  ( $\forall (e: \text{Accesses}) : st'(e) \wedge e^{\prime\prime}\text{seqnumber} < a'^{\prime\prime}\text{seqnumber}$ )  $\wedge$ 
      IF ( $\neg (\exists (x: \text{Accesses}) : st'(x) \wedge x^{\prime}s = a'^{\prime}s)$ )
          THEN add( $a'$ ,  $st'$ )( $xb'$ )
      ELSE IF  $a'^{\prime\prime}m = \text{read}$ 
          THEN add( $a'$ ,  $st'$ )( $xb'$ )
      ELSE IF  $\forall (y: \text{Accesses}) :$ 
           $st'(y) \wedge$ 
           $y^{\prime}s = a'^{\prime}s \wedge$ 
           $a'^{\prime\prime}m = \text{write} \wedge \text{lb}(a'^{\prime\prime}\text{ob}) \geq \text{lb}(y^{\prime}\text{ob})$ 
          THEN add( $a'$ ,  $st'$ )( $xb'$ )
      ELSE  $st'(xb')$ 
      ENDIF
  ENDIF
ENDIF

{-2}  $st'(xa')$ 
{-3}  $st'(xa') \wedge st'(xb') \Rightarrow$ 
    IF  $xa'^{\prime\prime}\text{seqnumber} = xb'^{\prime\prime}\text{seqnumber}$ 
        THEN  $xa' = xb'$ 
    ELSE  $xa'^{\prime}s = xb'^{\prime}s \wedge$ 
         $xa'^{\prime\prime}\text{seqnumber} < xb'^{\prime\prime}\text{seqnumber} \wedge xb'^{\prime\prime}m = \text{write}$ 
         $\Rightarrow \text{lb}(xa'^{\prime\prime}\text{ob}) \leq \text{lb}(xb'^{\prime\prime}\text{ob})$ 
    ENDIF


---


{1}  $\forall (e: \text{Accesses}) : st'(e) \wedge e^{\prime\prime}\text{seqnumber} < a'^{\prime\prime}\text{seqnumber}$ 
{2} IF  $xa'^{\prime\prime}\text{seqnumber} = xb'^{\prime\prime}\text{seqnumber}$ 
    THEN  $xa' = xb'$ 
    ELSE  $xa'^{\prime}s = xb'^{\prime}s \wedge$ 
         $xa'^{\prime\prime}\text{seqnumber} < xb'^{\prime\prime}\text{seqnumber} \wedge xb'^{\prime\prime}m = \text{write}$ 
         $\Rightarrow \text{lb}(xa'^{\prime\prime}\text{ob}) \leq \text{lb}(xb'^{\prime\prime}\text{ob})$ 
    ENDIF

```

Applying disjunctive simplification to flatten sequent,
This completes the proof of `transform_secure.2.1`.

transform_secure.2.2:

```

{-1}  ⊢ (forall (e: Accesses): st'(e) ∧ e' seqnumber < a' seqnumber) ∧
      st'(xb')
{-2}  st'(xa')
{-3}  st'(xa') ∧ st'(xb') ⇒
      IF xa' seqnumber = xb' seqnumber
          THEN xa' = xb'
          ELSE xa' s = xb' s ∧
              xa' seqnumber < xb' seqnumber ∧ xb' m = write
              ⇒ lb(xa' ob) ≤ lb(xb' ob)
      ENDIF
{1}   ∀ (e: Accesses): st'(e) ∧ e' seqnumber < a' seqnumber
{2}   IF xa' seqnumber = xb' seqnumber
       THEN xa' = xb'
       ELSE xa' s = xb' s ∧
           xa' seqnumber < xb' seqnumber ∧ xb' m = write
           ⇒ lb(xa' ob) ≤ lb(xb' ob)
   ENDIF

```

Applying disjunctive simplification to flatten sequent,

transform_secure.2.2:

```

{-1} st'(xb')
{-2} st'(xa')
{-3} st'(xa') ∧ st'(xb') ⇒
      IF xa'`seqnumber = xb'`seqnumber
          THEN xa' = xb'
      ELSE xa'`s = xb'`s ∧
          xa'`seqnumber < xb'`seqnumber ∧ xb'`m = write
          ⇒ lb(xa'`ob) ≤ lb(xb'`ob)
      ENDIF
{1} ∀ (e: Accesses): st'(e) ∧ e`seqnumber < a`seqnumber
{2} ∀ (e: Accesses): st'(e) ∧ e`seqnumber < a`seqnumber
{3} IF xa'`seqnumber = xb'`seqnumber
      THEN xa' = xb'
      ELSE xa'`s = xb'`s ∧
          xa'`seqnumber < xb'`seqnumber ∧ xb'`m = write
          ⇒ lb(xa'`ob) ≤ lb(xb'`ob)
      ENDIF

```

Applying propositional simplification,

This completes the proof of `transform_secure`.
2.2.

Q.E.D.