

High WaterMark PVS Problem

George W. Dinolt

August 9, 2004

1 Description of the Model

The figure below is an example of the High Water Mark Security Model. The model consist of the sets *Subjects*, *Objects* and *Labels*. Associated with each *Object* is a *Label*.

Each action of the system consists of adding an *Access* to the system. The basic idea is that a subject can read anything, but once it has read something, it can only write at or above the label of all the *Objects* it has ever read.

We model this by associating a “sequence number” with each access. We assume that if two *accesses* have the same sequence number they are the identical (the same). If they have different sequence numbers then the one with the lower number “happened” first. So, if a subject wants to write to an object, then the label of the object must dominate (be greater than or equal to) the label of all the objects it has read, i.e. the objects associated with all the accesses it has made with lower sequence numbers.

We define the transform function (the state transition function) in such a way that it guarantees that the sequence numbers are monotonically increasing.¹

2 What you need to do

As usual, you will be able to find the specification on the Web Pages. You should be able to download-it from there. You should try to convince yourself that the specification actually does capture the intent of the policy above. I have added one lemma to try and make it clear that we are “doing the right thing.”

There are 3 LEMMA's, `transform_secure`, `empty_is_secure`, and `noWriteDownSubject`. You should hand in proofs only for `empty_is_secure` and `noWriteDownSubject`.

A proof of `transform_secure` will give you extra credit. It is fairly long and complicated.

¹If you don't know what that means, look it up.

3 The High Water Mark Specification

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 =

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

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

IF $\forall (e: \text{Accesses}):$
 $st(e) \Rightarrow e'seqnumber < a'seqnumber$
 THEN COND ($\neg (\exists (x: \text{Accesses}): st(x) \wedge x's = a's)$)
 $\rightarrow (st \cup \{a\}),$
 $a'm = \text{read} \rightarrow (st \cup \{a\}),$
 $(\forall (y: \text{Accesses}):$
 $st(y)$
 \wedge
 $y's = a's$
 \wedge
 $a'm = \text{write} \wedge lb(a'ob) \geq lb(y'ob))$
 $\rightarrow (st \cup \{a\}),$

```

        ELSE → st
    ENDCOND
ELSE st
ENDIF

```

```

transform_secure: LEMMA
  ∀ (a: Accesses, st: State):
    st?(st) ⇒
      st?(transform(a, st))

```

```

empty_is_secure: LEMMA
  st?(emptyset)

```

```

noWriteDownSubject: LEMMA
  ∀ (st: State, a, b: Accesses):
    st?(st) ∧
      (a ∈ st) ∧
      (b ∈ st) ∧
      a'm = read ∧
      b'm = write ∧
      a's = b's ∧ a'seqnumber < b'seqnumber
    ⇒ lb(a'ob) ≤ lb(b'ob)

```

```

END HighWaterMark

```