Project: DRA FRONT END FILTER PROJECT

Title: Informal Justifications for Proof of Security

Ref: DS/FMU/FEF/016 *Issue: Revision* : *2.1* *Date:* 5 June 2016

Status: Approved *Type:* Specification

Keywords:

Author:

Name	*Location*	*Signature*	*Date*
G. M. Prout	WIN01		

Authorisation for Issue:

Name	*Function*	*Signature*	*Date*
R.B. Jones	HAT Manager		

Abstract: This document gives informal justifications of those unproven axioms which have been included in the formal proof of security for Phase 1 of the DRA front end filter project RSRE 1C/6130.

Distribution: HAT FEF File
Simon Wiseman

0 DOCUMENT CONTROL

0.1 Contents List

0.2 Document Cross References

[1] *Secure Database Technical Proposal - Amendent 1.* High Assurance Team, ICL Secure Systems, WIN01, 11th February 1992.

[2] DS/FMU/FEF/015. *Proof of Security (IIe).* G.M. Prout, ICL Secure Systems, WIN01.

0.3 Changes History

Issue *Revision* : *2.1* **(5 June 2016)** First approved version.

Issue 2.2 Removed dependency on ICL logo font

0.4 Changes Forecast

None.

1 GENERAL

1.1 Scope

This document gives informal justifications for the unproven axioms used in the formal Phase 1 proof that the behaviour of the SSQL abstract machine is secure. It constitutes deliverable D17 of work package 1a, as given in the amendment to the Secure Database Technical Proposal, [1].

1.2 Introduction

For the purpose of proving the security of the SSQL semantics, DRA-ED has agreed that they do not require the proof of all lemmas whose substance is a mathematical result whose proof would be routine and would not depend on any of the security features in SWORD. Five unproven axioms have been used in the proof of the main result *secureSSQL*, given in [2], that the behaviour of the SSQL abstract machine is secure:

HOL output

$$secureSSQL = \vdash behaviours\ SSQLam \in secure$$

Informal justifications of these axioms are given below.

2 AXIOMS USED

2.1 Enumerate Axiom

$$\mathbf{fin\_enumerate\_ax} =$$
$$\forall\ s \bullet s \in Finite \Rightarrow Enumerate\ s \in Finite \wedge \#\ s = \#\ (Enumerate\ s)$$

The constant *Enumerate* orders a set of natural numbers. For example:

$$Enumerate\ \{4,9,2,7\} = \{(1,2),(2,4),(3,7),(4,9)\}$$

For every element in the set s there is exactly one element in *Enumerate* s. Hence *Enumerate* s is finite if and only if s is finite, and if s is finite then it must be the same size as *Enumerate* s.

2.2 Squash Axiom

$$\mathbf{fin\_squash\_eq\_ax} =$$
$$\forall\ r_1\ r_2 \bullet$$
$$r_1 \in Finite \wedge r_2 \in Finite \wedge Squash\ r_1 = Squash\ r_2$$
$$\Rightarrow \#(Dom\ r_1) = \#(Dom\ r_2)$$

The constant *Squash* takes a relation whose domain is a set of natural numbers and returns a relation where the domain is 'squashed' into consecutive numbers. For example:

$$Squash\ \{(3,p),(5,q),(7,r),(7,s),(8,t),(9,s)\} = \{(1,p),(2,q),(3,r),(3,s),(4,t),(5,s)\}$$

For every element in the relation r there is exactly one element in *Squash* r. Hence *Squash* r is finite if and only if r is finite, and if the squash of two finite relations is equal, then the domains of the two relations must be the same size.

2.3 Extract Axiom

$$\begin{array}{l}\textbf{extract\_}\frown\textbf{\_single\_ax} = \\ \forall\ l\ a\ x \bullet Extract\ a\ (l \frown [x]) = \\ \qquad\qquad if\ Length\ l + 1 \in a \\ \qquad\qquad then\ (Extract\ a\ l) \frown [x] \\ \qquad\qquad else\ Extract\ a\ l\end{array}$$

The constant *Extract* takes a set of indices and a list and returns the list obtained by extracting those elements from the list which are indexed by the elements of the set. For example:

$$Extract\ \{2,3,5\}\ [a,b,c,d] = [b,c]$$

Given a list l with a single element x appended to it, then the list obtained by extracting elements identified by a set of indices s will be the same as appending x to that list obtained by extracting elements from l provided that s contains the index for the next element in the list. If not, the result will be the same list as that obtained by just extracting elements from the list l.

2.4 RelList Axioms

The constant *ListRel* takes a list and returns a sequence. For example:

$$ListRel\ [a,f,d,a] = \{(1,a),(2,f),(3,d),(4,a)\}$$

The constant *RelList* is the left inverse of *ListRel*.

The relational override constant $\oplus$ takes a relation f and a relation g and returns the relation which agrees with f in the case where the domain of f differs from the domain of g, and agrees with g elsewhere. For example:

$$\{(a_1,b_1),(a_2,b_2)\} \oplus \{(a_2,c_1),(a_3,c_2)\} = \{(a_1,b_1),(a_2,c_1),(a_3,c_2)\}$$

$$(where\ a_1 \neq a_2,\ a_2 \neq a_3,\ a_1 \neq a_3)$$

2.4.1 RelList Axiom I

$$
\begin{array}{|l}
\textbf{rel\_list\_}\oplus\textbf{\_ax1} = \\
\forall\ r\ l\ x \bullet \\
\quad r \in \mathit{Functional} \land \mathit{Dom}\ r \subseteq \mathit{Dom}(\mathit{ListRel}\ l) \\
\quad \Rightarrow \\
\quad \mathit{RelList}((\mathit{ListRel}(l \frown [x])) \oplus r) \\
\quad = \\
\quad \mathit{RelList}((\mathit{ListRel}\ l) \oplus r) \frown [x]
\end{array}
$$

This axiom is for use in an inductive proof where the elements of a list are to be overridden by values given from a relation for which there is no value for the last element of the list, hence it remains unchanged.

rel_list_⊕_ax1 is given a relation r whose domain is contained in the sequence obtained from the list l. Hence:

$$|\ (\mathit{ListRel}(l \frown [x])) \oplus r = ((\mathit{ListRel}\ l) \oplus r) \cup \{(\#l + 1, x)\}$$

and since r is also functional

$$|\ \exists\ l_1 \bullet \mathit{ListRel}\ l_1 = (\mathit{ListRel}\ l) \oplus r \land \#\ l_1 = \#l$$

Hence:

$$|\ \mathit{RelList}((\mathit{ListRel}(l \frown [x])) \oplus r) = \mathit{RelList}(((\mathit{ListRel}\ l) \oplus r) \cup \{(\#l + 1, x)\})$$

and

$$|\ \mathit{RelList}(((\mathit{ListRel}\ l) \oplus r) \cup \{(\#l + 1, x)\}) = \mathit{RelList}((\mathit{ListRel}\ l_1) \cup \{(\#l_1 + 1, x)\})$$

and

$$|\ \mathit{RelList}((\mathit{ListRel}\ l_1) \cup \{(\#l_1 + 1, x)\}) = \mathit{RelList}((\mathit{ListRel}\ l_1) \frown [x]$$

and finally

$$|\ \mathit{RelList}((\mathit{ListRel}\ l_1) \frown [x] = \mathit{RelList}((\mathit{ListRel}\ l) \oplus r) \frown [x]$$

as required.

2.4.2 RelList Axiom II

This axiom is for use in an inductive proof where the elements of a list are to be overridden by values given from a relation for which there is a value for the last element of the list, hence it takes that value.

rel_list_⊕_ax2 is given a relation r whose domain is contained in the sequence obtained from the list l. Hence:

$$\vdash (ListRel(l \frown [x])) \oplus (r \cup \{(\#l + 1, y)\}) = ((ListRel\ l) \oplus r) \cup \{(\#l + 1, y)\}$$

and again, since r is also functional

$$\vdash \exists\ l_1 \bullet ListRel\ l_1 = (ListRel\ l) \oplus r \land \#\ l_1 = \#l$$

Hence:

$$\vdash RelList((ListRel(l \frown [x])) \oplus (r \cup \{(\#l + 1, y)\})) = RelList(((ListRel\ l) \oplus r) \cup \{(\#l + 1, y)\})$$

and

$$\vdash RelList(((ListRel\ l) \oplus r) \cup \{(\#l + 1, y)\}) = RelList((ListRel\ l_1) \cup \{(\#l_1 + 1, y)\})$$

and

$$\vdash RelList((ListRel\ l_1) \cup \{(\#l_1 + 1, y)\}) = RelList((ListRel\ l_1) \frown [y]$$

and finally

$$\vdash RelList((ListRel\ l_1) \frown [y] = RelList((ListRel\ l) \oplus r) \frown [y]$$

as required.

3 INDEX