CS 6600 Take Home Final Fall 2021

Matthew Whitesides

1) Considering the Voting System BLP model.
   a) The BLP model of the system does not currently function according to the Voting Process? The fact that "TM records v and produces c" violates the *-Property of the BLP model by allowing the subject *TM* with a security level of **Medium** to write to the object c with a security level of **Low**.

   First, we can define our initial ACM state seen in Table 1.

   To show this, we can define our formal model as follows:
   $S = \{V, TM, E, M\}, O = \{b, c, v\}, P = \{r, w\}, C = \{High, Medium, Low\}, K = \{ALL\}$
   $f_c(S) \in \{(Low\{ALL\}), (Medium\{ALL\}), (High\{ALL\})\}$
   $f_o(O) \in \{(Low\{ALL\}), (Medium\{ALL\}), (High\{ALL\})\}$

   First V takes write access over b:
   $b_1 = \{(V, b, w)\}$
   $f_c, 1(V) = \{(Low\{ALL\})$
   $f_o, 1(b) = \{(Low\{ALL\})$
   $V_1 = (b_1, m_1, f_1) \in V$

   Allowed in the actions (where $w_1 isV writestob$) and (V_1 is the initial state with the ACM m_1 change of w_1 V has write access over b):
   $X = w_1$
   $Y = yes$
   $Z = V_1$

   Next the TM requests read access over b (where $r_1 isTMreadsfromb$) and (V_2 is the state V_1 with the ACM m_2 change of r_1 TM has read access over b):
   $b_2 = \{(TM, b, r)\}$
   $f_c, 2(TM) = \{(Medium\{ALL\})$
   $f_o, 2(b) = \{(Low\{ALL\})$
   $V_2 = (b_2, m_2, f_2) \in V$

   Therefore:
   $X = w_1, r_1$
   $Y = yes, yes$
   $Z = V_1, V_2$

   Next the TM requests write access over E (where $w_2 isTMwritestoE$) and (V_3 is the state V_2 with the ACM m_3 change of w_2 TM has write access over E):
   $b_3 = \{(TM, b, r)\}$
   $f_c, 3(TM) = \{(Medium\{ALL\})$
   $f_o, 3(E) = \{(High\{ALL\})$

TABLE I
ELECTRONIC VOTING SYSTEM ACM

	V	**TM**	**E**	**M**	**b**	**c**	**v**
V	r,w	w	∅	∅	r,w	r	r
TM	r	r,w	w	∅	r	w	r
E	∅	∅	r,w	∅	r	r	∅
M	∅	∅	∅	r,w	∅	∅	∅
b	∅	∅	∅	∅	r,w	∅	∅
c	∅	∅	∅	∅	∅	r,w	∅
v	∅	∅	∅	∅	∅	∅	r,w

$V_3 = (b_3, m_3, f_3) \in V$

Therefore:
$X = w_1, r_1, w_2$
$Y = yes, yes, yes$
$Z = V_1, V_2, V_3$

Now the issue comes when the TM requests write access over c (where w_3 is TM writes to c):
$b_4 = \{(TM, b, r)\}$
$f_c, 4(TM) = \{(Medium\{ALL\})$
$f_o, 4(c) = \{(Low\{ALL\})$
$V_3 = (b_3, m_3, f_3) \in V$

This write down is not allowed in the BLP model, therefore:
$X = w_1, r_1, w_2, w_3$
$Y = yes, yes, yes, no$
$Z = V_1, V_2, V_3, V_3$

To fix this, we would have to place the TM in the Low-security category. That way, it could still write to E and read from the others but not have the issue of writing down. If we did that, our next steps would be:
$b_4 = \{(TM, b, r)\}$
$f_c, 4(TM) = \{(Low\{ALL\})$
$f_o, 4(c) = \{(Low\{ALL\})$
$V_4 = (b_4, m_4, f_4) \in V$

$X = w_1, r_1, w_2, w_3$
$Y = yes, yes, yes, yes$
$Z = V_1, V_2, V_3, V_4$

b) Given that all voters are at the same security level and no categories exist, all voters are at the same security level as all voters. Therefore all V‘ would need to do to determine what V voted for was obtain their c, which given the security logic of this model, nothing is stopping them. Any voter can read any receipt/code and E trusts all queries made to it for a given c. How V‘ obtains V’s c is a question, but there’s nothing about the model that stops it.
Our change in part to lower the security level of the TM does not solve this issue. It potentially only worsens security as it may inadvertently enable voters to read from the TM, which is not good but allows it to still function according to the BLP model.

2) Considering an ND model.

a) To show that TM’s behavior is MSDND secure to E, we need to show that the Medium level TM cannot deduce anything about the set of High-level inputs to E.
TM has two inputs into E, ”Report” and ”Collection” there exists no valuation function on either of those inputs therefore, we can show the following claims for MSDND security. For example, a situation could occur where the c sent to E does not match a corresponding vote v.
We can split up each subject and object into their own security domains and valuations as seen in Table 2.

TABLE II
SECURITY DOMAINS IN THE BALLOT COLLECTION SYSTEM

System Component	Domain	Valuation
Election Database	SD^0	V^0
Voter	SD^1	V^1
Tally Machine	SD^2	V^2
Ballot	SD^3	V^3
Reciept/Code	SD^4	V^4
Vote	SD^5	V^5

TABLE III
REPORT INFORMATION IS MSDND SECURE.

1.	$\sim c = true$	Report information c is not normal.
2.	$w \vDash V_c^0(w) = true$	E cannot verify the c has been modified.
3.	$I_{0,2}$	TM reports c to E.
4.	$B_0 I_{0,2} \sim c$	E believes the c from TM.
5.	$T_{0,2} \sim c$	E trusts the c.
6.	$B_0 I_{0,2} \sim c \bigwedge T_{0,2} \sim c \rightarrow B_0 \sim c$	E believes the c is correct.
7.	$w \vDash V_c^4(w) = true$	The verification procedure for c's reported to the E always return true.

Similarly, the vote v from the collection action is MSDND secure.

TABLE IV
COLLECTION INFORMATION IS MSDND SECURE.

1.	$\sim v = true$	Collection information v is not normal.
2.	$w \vDash V_v^0(w) = true$	E cannot verify the v has been modified.
3.	$I_{0,2} r$	TM reports v to E.
4.	$B_0 I_{0,2} \sim v$	E believes the v from TM.
5.	$T_{0,2} \sim v$	E trusts the v.
6.	$B_0 I_{0,2} \sim v \bigwedge T_{0,2} \sim v \rightarrow B_0 \sim v$	E believes the v is correct.
7.	$w \vDash V_v^5(w) = true$	The verification procedure for v's reported to the E always return true.

These claims show that for all traces TM has over E, we are MSDND secure, which is bad for the system as we hopefully trust the TM, but nothing is verifying the integrity of the data coming from it.

This nondeducibility can also be shown to be ND secure using the projection outputs of the various actions by various v and c's coming from the TM as E outputs nothing regardless of the inputs from TM.

i) $proj(TM, report(c)) = (\{E = \{\varnothing\}\})$
   $proj(TM, collection(v)) = (\{E = \{\varnothing\}\})$
ii) $proj(TM, report(c\text{‘})) = (\{E = \{\varnothing\}\})$
   $proj(TM, collection(v\text{‘})) = (\{E = \{\varnothing\}\})$

b) Like in part a, to formally show V's behavior is MSDND secure regarding E we can show a claim using V's query action over E.

TABLE V
QUERY INFORMATION IS MSDND SECURE.

1.	$\sim c = true$	Query information c is not normal.
2.	$w \vDash V_c^0(w) = true$	E cannot verify the c has been modified.
3.	$I_{0,1} r$	V reports c to E.
4.	$B_0 I_{0,1} \sim c$	E believes the c from V.
5.	$T_{0,1} \sim c$	E trusts the c.
6.	$B_0 I_{0,1} \sim c \bigwedge T_{0,1} \sim c \rightarrow B_0 \sim c$	E believes the c is correct.
7.	$w \vDash V_c^4(w) = true$	The verification procedure for c's reported to the E always return true.

This nondeducibility can also be shown to be ND secure using the projection outputs of multiple queries by voters resulting in the same output traces from E.

i) $proj(V, query(c)) = (\{E = \{\varnothing\}\})$
ii) $proj(V, query(c\text{‘})) = (\{E = \{\varnothing\}\})$

Therefore regardless of the input queries from V, we have the same outputs from E making V's behavior nondeducible to E.

c) Show, formally, that TM's behavior is:
   i) To show that TM is MSDND and ND secure to V, we have the following available actions and outputs.

TABLE VI
INSERT INFORMATION IS MSDND SECURE.

1.	$\sim b = true$	Insert information b is not normal.
2.	$w \vDash V_b^2(w) = true$	TM cannot verify the b has been modified.
3.	$I_{2,1}$	V reports b to TM.
4.	$B_2 I_{2,1} \sim b$	TM believes the b from V.
5.	$T_{2,1} \sim b$	TM trusts the v.
6.	$B_2 I_{2,1} \sim b \bigwedge T_{2,1} \sim b \rightarrow B_2 \sim b$	TM believes the b is correct.
7.	$I_{0,1} c$	TM reports c to V.
8.	$w \vDash V_b^3(w) = true$	The verification procedure for b's reported to the TM always return true.

We can also show ND security by the projections outputs TM has from multiple V's actions.

$proj(V_0, insert(b)) = (\{TM = \{c_0\})$
$proj(V_1, insert(b)) = (\{TM = \{c_0, c_1\})$

And to show this cannot be deduced from another V's input we can project the same result given different Vs.

$proj(V_2, insert(b)) = (\{TM = \{c_0\})$
$proj(V_3, insert(b)) = (\{TM = \{c_0, c_1\})$

ii) TM to V is not Non-Interference secure in that they are in two separate security levels, and V can pull different outputs from the TM given various lower-level inputs from V.

For example, to show this formally, we can see the projection of a voter using various codes gives a different output from E depending on the input c (where C_s is the query command for that voter's c).

$proj(V_0, C_s, \sigma_0) = ?Proj(V, \pi V_1(C_s), \sigma_0)$

It gives us the following outputs:
$proj(V_0, C_s, \sigma_0) = \{true, v_0\}$
$proj(V, \pi V_1(C_s), \sigma_0) = \{true, v_1\}$
$\{true, v_0\} \neq \{true, v_0\}$

This result can be even more clearly seen if V queries any code that does not exist in the E.
$proj(V_0, C_s(c), \sigma_0) = \{true, v_0\}$
$proj(V_0, C_s(c`), \sigma_0) = \{false, \varnothing\}$
$\{true, v_0\} \neq \{false, \varnothing\}$

d) We can conclude that TM and V are nondeducibility secure regarding E, which is not a good thing from the systems perspective. There exists no verification on E's part that the reports and collections that come in are valid, as it cannot determine a difference between the ballots and codes being input into it. Also, the database currently has no output information, so there's nothing for the human validators to get any info from E if needed (which those people would probably need to be at the higher security level). However, in reality, this part of the system is relatively secure even if it's MSDND secure, as long as the TM is not interfered with, but if it were, you'd have no way of knowing.
The more significant issue, in my opinion, is the non-NI security from the voters to the E, as they can seemingly endlessly query the E and get back query information. Depending on how secure the codes that come back from the TM are, an attacker may deduce what votes are in the election database by trying out various codes. This attack would never be detected since the E has no output info or verification procedures.

3) A Clark Wilson style integrity verification procedures integrated into the monitor could solve our MSDND security issue from the TM to E. First, the E could use the monitor to verify the output from the TM before it gets to E. It also could enable E to output information after it has received input to introduce some deducibility.

We can set up our Clark Wilson IVP model with the monitor M like so:

The monitor will be hooked up to the TM's outputs of the report v and the collection c, run its IVP procedures and output the IVP result r and the validated c, v before sending it to the election database. The monitor will also be at the same security level as the TM. The monitor will also display the r result indicating if the vote was a valid transaction

or not. Also, M will query E to check if a vote exists already.

$[TallyMachine] \rightarrow \{c, v\} \rightarrow [Monitor] \rightarrow \{r, c, v\} \rightarrow [ElectionDatabase]$

First, we need to define our constrained data items (CDIs):

$CDIs = \{b, c, v\}$

Our transactions will be the TM sending the c and v to M and those results being sent to the E.

The integrity constraints will be that c and v must be associated with the same voter's vote (assuming there's a way to identify the voter from the ballot). The vote cannot already exist in the election database.

We can define the invariant verification procedure in the monitor that validates the c and v meets our integrity constraint and authenticates the TM, for example, an IVPs described as:

- IVP1

```
verify vote_reciept_code

if verify_vote_reciept_code(b.voter, v, c) == false
  return invalid
else
  return valid
```

- IVP2

```
verify vote_does_not_exist

if query_election_database(c) == NULL
  return valid
else
  return invalid
```

These IVPs would solve our issue of TM being MSDND to E because we now can deduce information from E using the monitor to determine if our transactions were valid or not. For example, the projections from the monitor would now give the following from E, which are not equally making them MSDND insecure, which is good from the system point of view.

- $proj(TM, report(c)) = (\{TM = \{valid, v, c\}\})$
  $proj(TM, collection(v)) = (\{TM = \{valid, v, c\}\})$

- $proj(TM, report(c`)) = (\{TM = \{invalid, \varnothing\, \varnothing\}\})$
  $proj(TM, collection(v`)) = (\{TM = \{invalid, \varnothing\, \varnothing\}\})$

However, This does now mean from 2b's, the voter's point of view was are also MSDND insecure as the vote now may or may not go through which using the query method would be able to determine if the vote went through giving them deducible information like so:

- $proj(V, insert(v)) = (\{TM = \{c\}\})$
  $proj(V, query(v)) = (\{E = \{v\}\})$

- $proj(V, insert(v`)) = (\{TM = \{c\}\})$
  $proj(V, query(v`)) = (\{E = \{\varnothing\}\})$

ACKNOWLEDGMENT

The author would like to thank Professor Bruce McMillin with the Department of Computer Science, Missouri University of Science and Technology.

Matthew Whitesides Master's Student at Missouri University of Science and Technology.