Week 3 Classwork due on Friday September 30, 23:59 hour

Group 5

Wong Ann Yi (1004000)
Liu Bowen (1004028)
Tan Chin Leong Leonard (1004041)

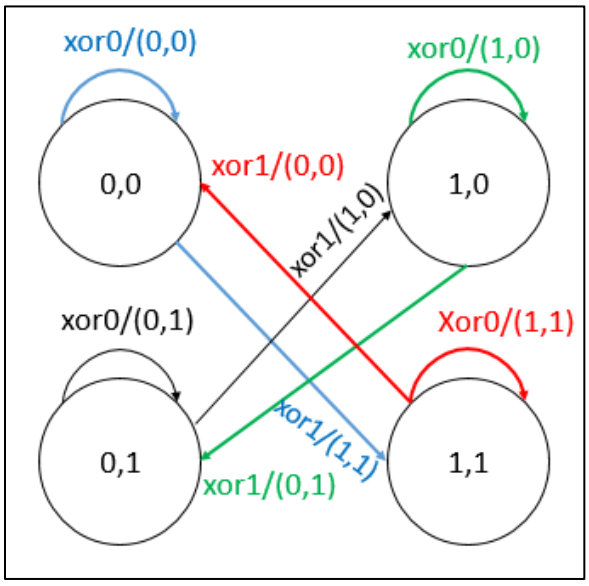
Exercise 1

Draw a finite state machine depicting the semantics of the 2-bit machine in the following Table 8-1 (State Transition Function) of Bishop's. Is this system secure? If yes, explain why. If not, show a counter example.

Commands	Input states (*H*, *L*)			
	(0, 0)	(0, 1)	(1, 0)	(1, 1)
xor0	(0, 0)	(0, 1)	(1, 0)	(1, 1)
xor1	(1, 1)	(1, 0)	(0, 1)	(0, 0)

Answer:

The finite state machine is as follows:



A system is secure if groups of subjects cannot interfere with one another. Therefore, if the set of outputs that any user can see corresponds to the set of inputs that that user can see, then the system is secure.

Using the above 2-bit machine, we provide an example to prove if it is secure. Let the initial state be $\sigma_0 = (0, 1)$ and two users: Holly (who can read high and low information) and Lucy (who can read only low information). Holly applies the command xor0, Lucy the command xor1, and Holly the command xor1 to the state machine. The commands affect both state bits, and both bits are output after each command. We take c_s to be the sequence (Holly, xor0), (Lucy, xor1), (Holly, xor1). The resulting output is 011001 (where bits are written sequentially, the high bit being first in each pair and the low bit is second and in red text).

By definition of 8-2, each command may produce some output, but subjects with insufficient clearance may not be able to see that output, lest they deduce information about the previous state of the system. The function proj(s, cs, σi) yields the list of outputs resulting from removing the outputs that s is not authorized to see. Therefore:

proj(Holly, c_s, σ_0) = 011001 (Holly can see both hers and Lucy's outputs)

proj(Lucy, c_s,σ_0) = 101 (Lucy can only see her outputs)

Base on the above, if the set of outputs that any user can see corresponds to the set of inputs that that user can see, then the system is secure but that is not the case.

Lucy who is cleared for only low, can take a sequence of low inputs and low outputs and from them deduce information from high inputs or outputs, then information has leaked from high to low making the system NOT secure. In the above example, Lucy can see the low bit outputs resulting from (Holly, xor0) and (Holly, xor1) and given sufficient observations, Lucy may deduce Holly's inputs or outputs.

One way to strengthen the security of this system is to modify the commands so that Holly can alter only the high bits and Lucy only the low bits. Using the same 2-bit machine and consider the sequence c_s = (Holly, xor0), (Lucy, xor1), (Holly, xor1). Given an initial state of (0,0), the output is $0_H1_L1_H$ which correspond to Holly's and Lucy's command sequence (where the subscripts indicate the security level of the output).

Applying Definition 8-4 (from Bishop's), we use G = {Holly}, G' = {Lucy}, and A = Ø. Then $\pi_{Holly}(c_s)$ = (Lucy, xor1), so proj(Lucy, $\pi_{Holly}(c_s)$, σ_0) = (1). Now we have proj(Lucy, c_s, σ_0) = proj(Lucy, $\pi_{Holly}(c_s)$, σ_0), and therefore { Holly } :| { Lucy } holds.

This shows no action that Holly takes has an effect on the part of the system that Lucy can observe. In other words, Holly's execution of commands is non-interfering with Lucy. This modification will strengthen the security of the system.

Exercise 2

Consider the instruction x:= y + z. Assume that x does not exist in state s and that the probability distribution of y and z is uniform over their domain y, z ϵ {0, 1}. Confirm that information flows from y and z to x by computing $H(y_s|x_t)$, $H(y_s)$, $H(z_s|x_t)$ and $H(z_s)$ and showing that $H(y_s|x_t) < H(y_s)$ and $H(z_s|x_t) < H(z_s)$.

Answer:

As y is equal distribution therefore p(y=0) or p(y=1) equals 1/2

The entropy $H(y_s) = -2*(1/2)*\lg(1/2) = 1$

Similarly, the entropy $H(z_s) = -2*(1/2)*\lg(1/2) = 1$

According to conditional entropy of X given Y is:

$H(X|Y) = -\sum_j p(Y=y_i) \sum_I p(X=x_i|Y=y_i) \lg p(X=x_i|Y=y_i)$

As for $H(y_s|x_t)$, X has three value: 0, 1, 2 and probability should be 1/4, 1/2 and 1/4 respectively.

Therefore, $H(y_s \mid x_t) = -\sum_t p(X=x_t) \sum_s p(Y=y_s \mid X=x_t) \lg p(Y=y_s \mid X=x_t) = 1/2$

When X has value 0, there is only one circumstances that y and z all are 0 therefore for X=0, the entropy should be 0.

When X has value 2, there is only one circumstances that y and z all are 1 therefore for X=2, the entropy should be 0.

When X has value 1, there is only two circumstances either y or z is 1 and the other is 0, therefore for X=1, the entropy should be 1/2.

Therefore, $H(y_s \mid x_t) = -(1/4+1/2+1/4)*(0+0+1/2)*\lg(1/2) = 1/2$

Similary, $H(z_s \mid x_t) = -\sum_t p(X=x_t) \sum_s p(Z=z_s \mid X=x_t) \lg p(Z=z_s \mid X=x_t) = 1/2$

And $H(z_s \mid x_t) = 1/2$

In conclusion:

$H(y_s \mid x_t) < H(y_s)$ and $H(z_s \mid x_t) < H(z_s)$ therefore information flows can happen from y and z to x.

Exercise 3

Consider the *auth* mechanism from Classwork Exercise 1 (Week 2). Write that mechanism as pseudo-code, where the value of the output variable o equals to 1 if the username and password are correct and 0 otherwise. Is there an information flow from d to o? Would a compiler or enforcement time mechanism have detected this and if so how? What is the conditional entropy of o in terms of the secret d?

Answer:

Write that mechanism as pseudo-code

if pair(username, password) ϵ d

then o := 1

else

o := 0

Is there an information flow from d to o?

Definition 16-1 (from Bishop's) states that the command sequence c causes a flow of information from x to y if $H(x_s \mid y_t) < H(x_s \mid y_s)$.

Note that $H(d_s|o_t) = 0$ as $o_t = 0$ or 1 when we know d, thus $H(d_s|o_t) < H(d_s)$. There is an information flow from d to o

3) Compiler would not be able to detect this as there is no input of u and p when it is complied. But an execution time based mechanism will.

Would a compiler or enforcement time mechanism have detected this and if so how?

The compiler-based mechanisms can detect information flows in program during compilation. In the above-mentioned pseudo-code, o = 1 iff the condition pair(username, password) ϵ d is true and vice versa. Obviously the d (class of d) is less than o. Thus, the compiler-based mechanisms can detect this information flow.

What is the conditional entropy of o in terms of the secret d?

$H(o \mid d) = - \sum_t p(d= d_t) \sum_s p(o= o_s \mid d= d_t) \lg p(o = o_s \mid d= d_t)$