Programming Assignment 2: Internet of Things - Smart Home Edition

TESTCASES DOCUMENT

Submitted by: - Rahul Raj and Olenka Dey

We inspected our program for different conditions few of the conditions are:

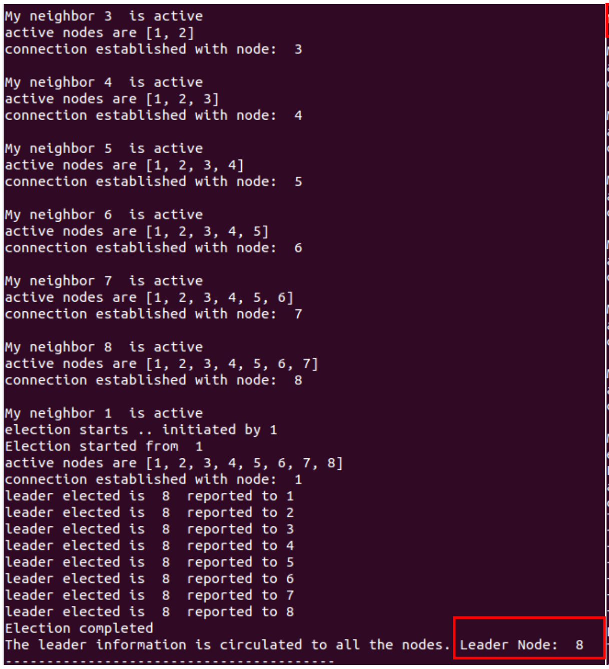
1. Testing if the leader election restarts when the leader dies.
2. Check if clock synchronization happens every time a leader is elected.
3. Testing the security system whether it can detect intrusion.
4. Detecting user home or away based on the motion sensor and door sensor ordering.
5. Checking multiple occupancy and detecting if that give some false positive detection.
6. Testing normal activities of the events and ordering of them by inspection using the history file and the plot generated based on the logical clocks

We basically tested our code by inspection these scenarios. These scenarios are created manually in the eventLog file.

Few of the above cases are shown below:

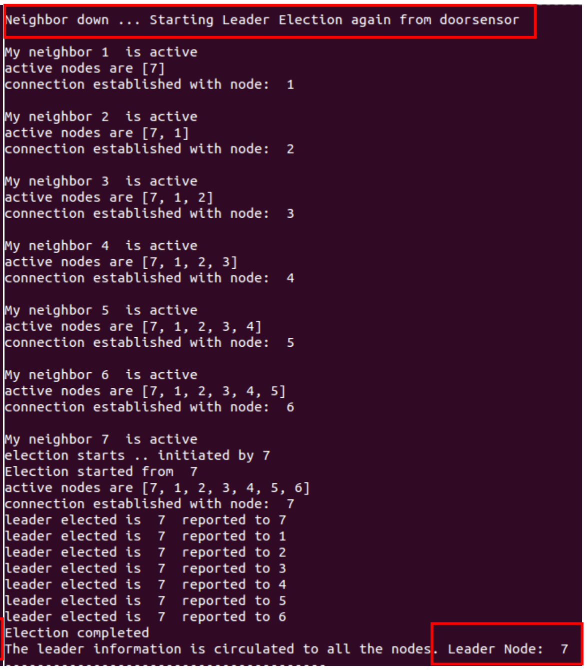
1. Testing restart of leader election process when current leader dies:

First run of leader Election:



```
My neighbor 3  is active
active nodes are [1, 2]
connection established with node:  3

My neighbor 4  is active
active nodes are [1, 2, 3]
connection established with node:  4

My neighbor 5  is active
active nodes are [1, 2, 3, 4]
connection established with node:  5

My neighbor 6  is active
active nodes are [1, 2, 3, 4, 5]
connection established with node:  6

My neighbor 7  is active
active nodes are [1, 2, 3, 4, 5, 6]
connection established with node:  7

My neighbor 8  is active
active nodes are [1, 2, 3, 4, 5, 6, 7]
connection established with node:  8

My neighbor 1  is active
election starts .. initiated by 1
Election started from  1
active nodes are [1, 2, 3, 4, 5, 6, 7, 8]
connection established with node:  1
leader elected is  8  reported to 1
leader elected is  8  reported to 2
leader elected is  8  reported to 3
leader elected is  8  reported to 4
leader elected is  8  reported to 5
leader elected is  8  reported to 6
leader elected is  8  reported to 7
leader elected is  8  reported to 8
Election completed
The leader information is circulated to all the nodes. Leader Node:  8
----------------------------------------
```

Second Run when node 8 died:



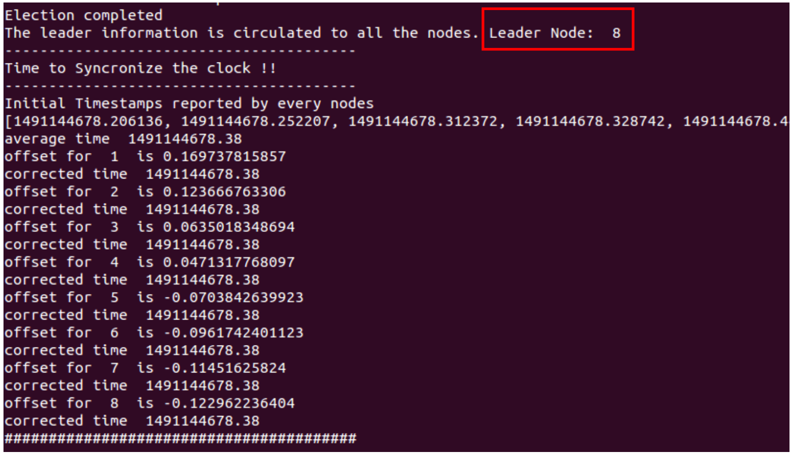
```
Neighbor down ... Starting Leader Election again from doorsensor

My neighbor 1  is active
active nodes are [7]
connection established with node:  1

My neighbor 2  is active
active nodes are [7, 1]
connection established with node:  2

My neighbor 3  is active
active nodes are [7, 1, 2]
connection established with node:  3

My neighbor 4  is active
active nodes are [7, 1, 2, 3]
connection established with node:  4

My neighbor 5  is active
active nodes are [7, 1, 2, 3, 4]
connection established with node:  5

My neighbor 6  is active
active nodes are [7, 1, 2, 3, 4, 5]
connection established with node:  6

My neighbor 7  is active
election starts .. initiated by 7
Election started from  7
active nodes are [7, 1, 2, 3, 4, 5, 6]
connection established with node:  7
leader elected is  7  reported to 7
leader elected is  7  reported to 1
leader elected is  7  reported to 2
leader elected is  7  reported to 3
leader elected is  7  reported to 4
leader elected is  7  reported to 5
leader elected is  7  reported to 6
Election completed
The leader information is circulated to all the nodes. Leader Node:  7
-----------------------------------------------
```

For killing the node 8, use the following two command:

1. ‘ps -eaf| grep python’
2. Locate the process id proc_id, for node 8.
3. Run ‘kill -9 proc_id’

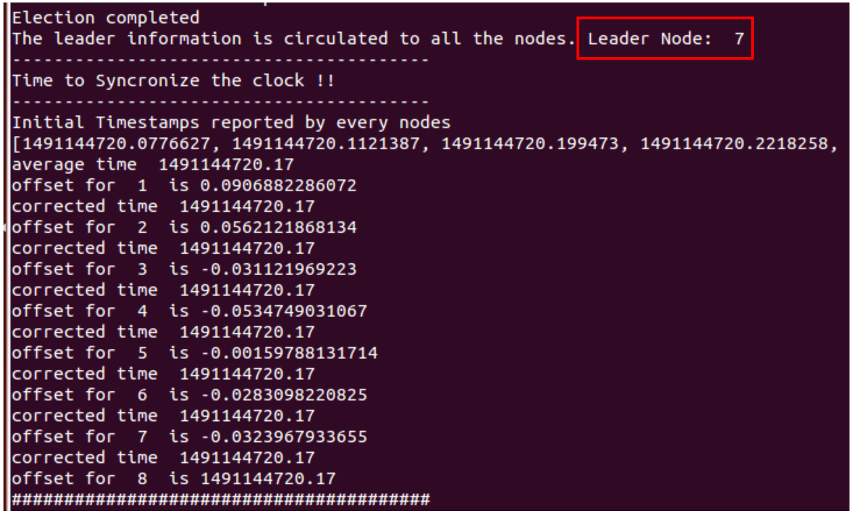
2. Testing if clock Synchronization happens every time a new leader gets elected.

First time clock synchronization:



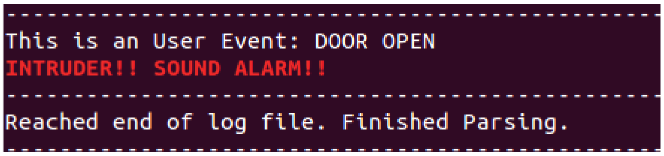
```
Election completed
The leader information is circulated to all the nodes. Leader Node:  8
----------------------------------------
Time to Syncronize the clock !!
----------------------------------------
Initial Timestamps reported by every nodes
[1491144678.206136, 1491144678.252207, 1491144678.312372, 1491144678.328742, 1491144678.4
average time  1491144678.38
offset for  1  is 0.169737815857
corrected time  1491144678.38
offset for  2  is 0.123666763306
corrected time  1491144678.38
offset for  3  is 0.0635018348694
corrected time  1491144678.38
offset for  4  is 0.0471317768097
corrected time  1491144678.38
offset for  5  is -0.0703842639923
corrected time  1491144678.38
offset for  6  is -0.0961742401123
corrected time  1491144678.38
offset for  7  is -0.11451625824
corrected time  1491144678.38
offset for  8  is -0.122962236404
corrected time  1491144678.38
########################################
```

Re-run of clock synchronization automatically when the new leader gets elected.



```
Election completed
The leader information is circulated to all the nodes. Leader Node:  7
----------------------------------------
Time to Syncronize the clock !!
----------------------------------------
Initial Timestamps reported by every nodes
[1491144720.0776627, 1491144720.1121387, 1491144720.199473, 1491144720.2218258,
average time  1491144720.17
offset for  1  is 0.0906882286072
corrected time  1491144720.17
offset for  2  is 0.0562121868134
corrected time  1491144720.17
offset for  3  is -0.031121969223
corrected time  1491144720.17
offset for  4  is -0.0534749031067
corrected time  1491144720.17
offset for  5  is -0.00159788131714
corrected time  1491144720.17
offset for  6  is -0.0283098220825
corrected time  1491144720.17
offset for  7  is -0.0323967933655
corrected time  1491144720.17
offset for  8  is 1491144720.17
########################################
```

3. Testing security system to detect intrusion

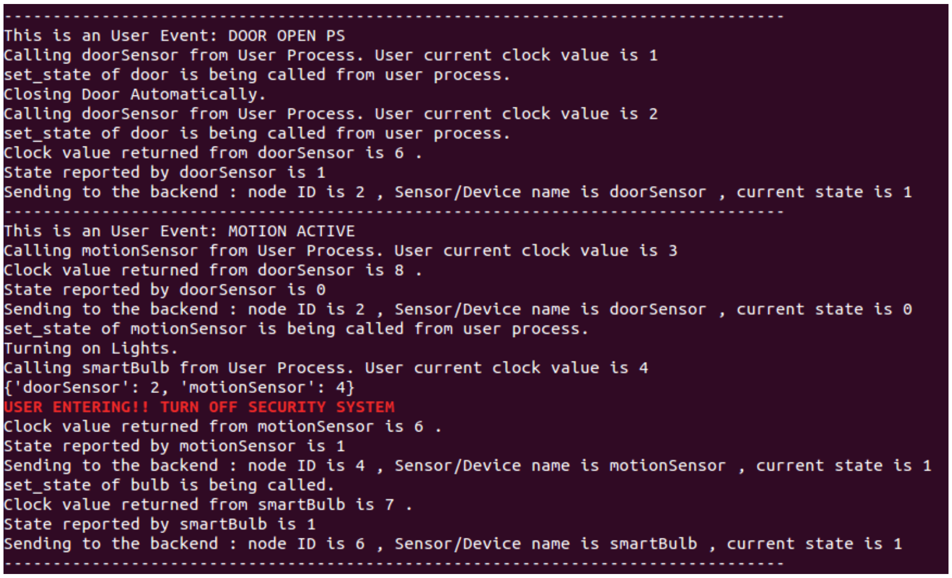


```
------------------------------------------------
This is an User Event: DOOR OPEN
INTRUDER!! SOUND ALARM!!
------------------------------------------------
Reached end of log file. Finished Parsing.
------------------------------------------------
```

When the user opens the door without the presence sensor “PS” attached to it, that means he/she is not the owner and thus the gateway reports security breach instantly and sounds alarm.

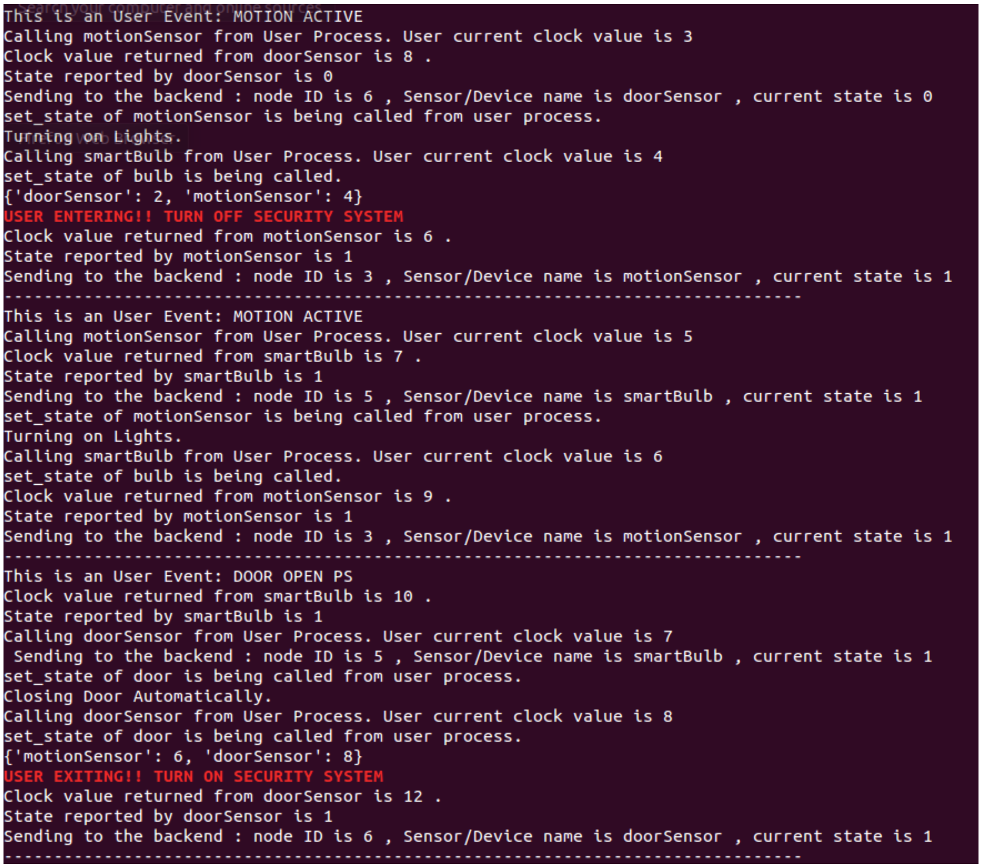
4. Detection of events based on the door and motion sensor ordering.

<u>Detecting “User Entered”:</u>



```
--------------------------------------------------------------------------------
This is an User Event: DOOR OPEN PS
Calling doorSensor from User Process. User current clock value is 1
set_state of door is being called from user process.
Closing Door Automatically.
Calling doorSensor from User Process. User current clock value is 2
set_state of door is being called from user process.
Clock value returned from doorSensor is 6 .
State reported by doorSensor is 1
Sending to the backend : node ID is 2 , Sensor/Device name is doorSensor , current state is 1
--------------------------------------------------------------------------------
This is an User Event: MOTION ACTIVE
Calling motionSensor from User Process. User current clock value is 3
Clock value returned from doorSensor is 8 .
State reported by doorSensor is 0
Sending to the backend : node ID is 2 , Sensor/Device name is doorSensor , current state is 0
set_state of motionSensor is being called from user process.
Turning on Lights.
Calling smartBulb from User Process. User current clock value is 4
{'doorSensor': 2, 'motionSensor': 4}
USER ENTERING!! TURN OFF SECURITY SYSTEM
Clock value returned from motionSensor is 6 .
State reported by motionSensor is 1
Sending to the backend : node ID is 4 , Sensor/Device name is motionSensor , current state is 1
set_state of bulb is being called.
Clock value returned from smartBulb is 7 .
State reported by smartBulb is 1
Sending to the backend : node ID is 6 , Sensor/Device name is smartBulb , current state is 1
--------------------------------------------------------------------------------
```

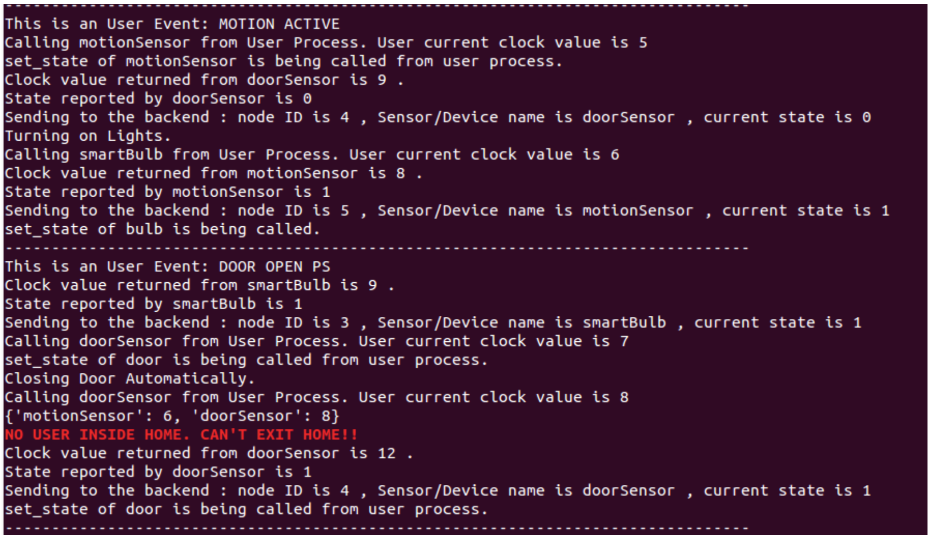
<u>Detecting User “Exit”:</u>



```
This is an User Event: MOTION ACTIVE
Calling motionSensor from User Process. User current clock value is 3
Clock value returned from doorSensor is 8 .
State reported by doorSensor is 0
Sending to the backend : node ID is 6 , Sensor/Device name is doorSensor , current state is 0
set_state of motionSensor is being called from user process.
Turning on Lights.
Calling smartBulb from User Process. User current clock value is 4
set_state of bulb is being called.
{'doorSensor': 2, 'motionSensor': 4}
USER ENTERING!! TURN OFF SECURITY SYSTEM
Clock value returned from motionSensor is 6 .
State reported by motionSensor is 1
Sending to the backend : node ID is 3 , Sensor/Device name is motionSensor , current state is 1
--------------------------------------------------------------------------------
This is an User Event: MOTION ACTIVE
Calling motionSensor from User Process. User current clock value is 5
Clock value returned from smartBulb is 7 .
State reported by smartBulb is 1
Sending to the backend : node ID is 5 , Sensor/Device name is smartBulb , current state is 1
set_state of motionSensor is being called from user process.
Turning on Lights.
Calling smartBulb from User Process. User current clock value is 6
set_state of bulb is being called.
Clock value returned from motionSensor is 9 .
State reported by motionSensor is 1
Sending to the backend : node ID is 3 , Sensor/Device name is motionSensor , current state is 1
--------------------------------------------------------------------------------
This is an User Event: DOOR OPEN PS
Clock value returned from smartBulb is 10 .
State reported by smartBulb is 1
Calling doorSensor from User Process. User current clock value is 7
 Sending to the backend : node ID is 5 , Sensor/Device name is smartBulb , current state is 1
set_state of door is being called from user process.
Closing Door Automatically.
Calling doorSensor from User Process. User current clock value is 8
set_state of door is being called from user process.
{'motionSensor': 6, 'doorSensor': 8}
USER EXITING!! TURN ON SECURITY SYSTEM
Clock value returned from doorSensor is 12 .
State reported by doorSensor is 1
Sending to the backend : node ID is 6 , Sensor/Device name is doorSensor , current state is 1
--------------------------------------------------------------------------------
```

Here it clearly detects multiple sequence of motion and door sensor and determines if user is home or away.

5. Checking false positive event detection:



```
-----------------------------------------------------------------------------
This is an User Event: MOTION ACTIVE
Calling motionSensor from User Process. User current clock value is 5
set_state of motionSensor is being called from user process.
Clock value returned from doorSensor is 9 .
State reported by doorSensor is 0
Sending to the backend : node ID is 4 , Sensor/Device name is doorSensor , current state is 0
Turning on Lights.
Calling smartBulb from User Process. User current clock value is 6
Clock value returned from motionSensor is 8 .
State reported by motionSensor is 1
Sending to the backend : node ID is 5 , Sensor/Device name is motionSensor , current state is 1
set_state of bulb is being called.
-----------------------------------------------------------------------------
This is an User Event: DOOR OPEN PS
Clock value returned from smartBulb is 9 .
State reported by smartBulb is 1
Sending to the backend : node ID is 3 , Sensor/Device name is smartBulb , current state is 1
Calling doorSensor from User Process. User current clock value is 7
set_state of door is being called from user process.
Closing Door Automatically.
Calling doorSensor from User Process. User current clock value is 8
{'motionSensor': 6, 'doorSensor': 8}
NO USER INSIDE HOME. CAN'T EXIT HOME!!
Clock value returned from doorSensor is 12 .
State reported by doorSensor is 1
Sending to the backend : node ID is 4 , Sensor/Device name is doorSensor , current state is 1
set_state of door is being called from user process.
-----------------------------------------------------------------------------
```

We are keeping track of how many users are inside the home and if not and activity of exit detected then we notify that it can be false detection as there is no one in the house.

6. Testing by inspection:

We specify different activity sequence and check the outputs generated in the console, history logged in the history.csv file about the activities and the plot generated during the run. We manually compare each event in all the three to check if the ordering matches or not.

As per out experiments, we found that the event ordering is achieved by our implementation as required.