Appendix A

Automation of experiments

This chapter covers the detail about the automation of experiments. Each experiments consists of four different processes: COOJA, router to connect the COOJA with other processes, DA and UA. Each process is started in an order depending on the parameters of a simulation run, as shown in Figure 6.5. Therefore, a bash script is written to automate the process for different runs and their multiple iterations. This chapter explains the automation process and presents examples from the actual codes used in simulations.

A.1 Scenarios automation

This research project has defined several bash scripts to execute simulations in a bulk. There are multiple scripts written to automate the process. A Code Listing A.1 gives an example script that is actually used in experiments. Furthermore, Figure A.1 shows the command given to this script to execute a set of simulation scenarios. The script uses the passed attributes to generate and update file appropriately, before actually starting all the process. The script makes changes in code files of various entities (SA, GM or GL) and generates new COOJA simulation files with the specified topology and settings. At the end of each simulation run, it saves the log files generated by all processes in a directory for debugging, validation and further processing.

Listing A.1: Simulations Automation script

```
1  #!/bin/bash

3  #Swicthing to correct GIT branches
   experimentDIR="/home/talal"
5  cd $experimentDIR/uatrendy/
   git checkout trendy-1
7  cd $experimentDIR/datrendy/
   git checkout trendy-1
9  cd $experimentDIR/sensinode-contiki/
   git add .
11 git reset --hard HEAD
   git checkout trendy-1-fullp
```

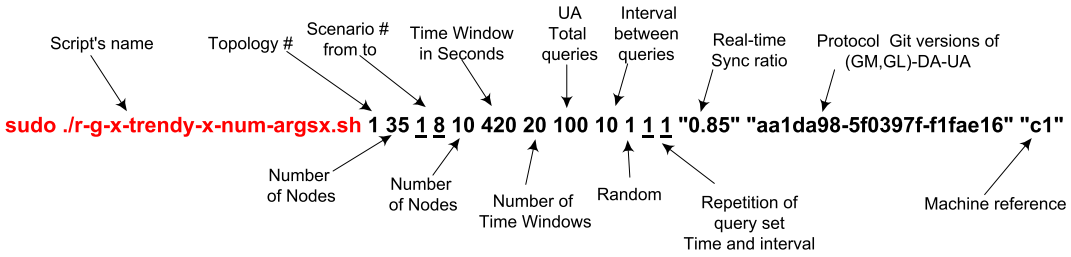

Figure A.1: Simulation automation: an example command to run bash script

```
13

15  #Checking for the right number of arguments
    noOfArgumentsReqd=12;
17  if [ "$#" -ne "$noOfArgumentsReqd" ]; then
      echo "Usage: Need $noOfArgumentsReqd arguments"
19    echo "Format: NUMBEROFNODES simModeFrom simModeTo ... DATotalTimeWindows simIterations
          simRef VMNo"
      exit 1
21  fi
     echo "topology= $1, simModeFrom=$2, simModeTo=$3, simIterations=$4, DATimeWindow=$5,
         DATotalTimeWindows=$6, UaStart = $7, queryInOneSet=$8, ratio=$9, nullRDC=${10}, simRef=$
         {11}, VMNo=${12}"
23
    topology=$1
25  simModeFrom=$2
    simModeTo=$3
27  simIterations=$4
    DATimeWindow=$5
29  DATimeWindow=$(( ${DATimeWindow#0} ))
    totalTimeWindows=$6
31  totalTimeWindows=$(( ${totalTimeWindows#0} ))
    DATotalTimeWindows=$6
33  UAStartTime=$7
    queryInOneSet=$8
35  queryInterval=10
    queryRandomInterval=1
37  querySetRepeatTimes=1
    querySetInterval=1
39  ratio=${9}
    nullRDC=${10}
41  simRef=${11}
    VMNo=${12}
43  NUMBEROFNODES=35

45  #Stopping Dropbox to free system's resources
    sudo service dropbox stop
47
    DA="$experimentDIR/datrendy/dist/datrendy.jar"
49  UA="$experimentDIR/uatrendy/dist/uatrendy.jar"
    totalSimulationDuration=$DATimeWindow*$DATotalTimeWindows
51
    #Logging Directories for all
53  BASE="/home/talal/Dropbox/experiments/results"
    gBASE="/home/talal/Dropbox/experiments/gresults"
55
    #Creating specific Directories
57  DATE=$(date +%F)
    TIME=$(date +%H%M)
59  FDIR="$BASE/$DATE"
    mkdir -p $FDIR
61  FDIR="$FDIR/$TIME-ALL-$simRef"
    mkdir -p $FDIR
63  echo $FDIR" is created"

65  gFDIR="$gBASE/$DATE"
    mkdir -p $gFDIR
67  gFDIR="$gFDIR/$TIME-$simRef"
    mkdir -p $gFDIR
69  echo $gFDIR" is created"
```

```
71
GDIR="$gFDIR/ALL-$NUMBEROFNODES-top$topology-tw$totalTimeWindows-q$queryInOneSet-
    r$queryRandomInterval-n$querySetRepeatTimes-for-$simIterations-[$VMNo]"
73 mkdir -p $GDIR
echo $GDIR" is created for Simulation record"
75
#Selecting a simulation variation
77 for ((i = $simModeFrom; i <= $simModeTo; i++)); do

79 SIMNAMEFORMATTED="case-$i-top$topology-tw$totalTimeWindows-q$queryInOneSet-
    r$queryRandomInterval-n$querySetRepeatTimes"
CONTIKI="$experimentDIR/sensinode-contiki"
81 timerMin=1;
timerMax=1;
83 timerThreshold=1;
timerStep=0;
85 #Setting configurations for each simulation
case "$i" in
87     "0")
           isgrouping=0;
89         numOfGLs=0;
           DAMode=0;
91         appubThreshold=0
           ;;
93     "1")
           isgrouping=0;
95         numOfGLs=0;
           DAMode=1;
97         appubThreshold=0
           ;;
99     "2")
           isgrouping=1;
101        numOfGLs=5;
           DAMode=4;
103        appubThreshold=0
           ;;
105    "3")
           isgrouping=0;
107        numOfGLs=0;
           DAMode=7;
109        appubThreshold=2
           ;;
111    "4")
           isgrouping=1;
113        numOfGLs=5;
           DAMode=8;
115        appubThreshold=2
           ;;
117    *)
           echo "Ending"
119        ;;
esac
121
#Random seeds for different number of iterations
123 for ((j = 1; j <= $simIterations; j++)); do

125 case "$j" in

127    "1")
           newrandomseed=1668841902061472829;
129        ;;
       "2")
131        newrandomseed=-8020676306221162569;
           ;;
133    "3")
           newrandomseed=-5174799744808039206;
135        ;;
       "4")
137        newrandomseed=5471226677158381259;
           ;;
139    "5")
           newrandomseed=2442086531776532400;
141        ;;
       "6")
143        newrandomseed=-52470407069163422;
           ;;
145    "7")
           newrandomseed=-290209011825205304;
147        ;;
```

```
    "8")
149         newrandomseed=-8489044479846245982;
            ;;
151     "9")
            newrandomseed=-2437497649994250447;
153         ;;
        "10")
155         newrandomseed=4869737434645930484;
            ;;
157     *)
            echo "Ending"
159         ;;
    esac
161
    #----------------COOJA SIMULATION GENERATION----------------------
163 #Total Nodes and/or grouping [Specific to Simulation]
    LOWPAN="$CONTIKI/work/trendy-gm"
165 COOJA="$CONTIKI/tools/cooja/dist/cooja.jar"

167 cd $LOWPAN
    headerFile="sim-header.cooja"
169 topologyFile="$NUMBEROFNODES-"g"-$numOfGLs"g"-$topology.topology"
    footerFile="sim-footer.cooja"
171
    SIMFILE="$NUMBEROFNODES-"g"-$numOfGLs"g"-$topology-grouping.csc"
173 cat $headerFile $topologyFile $footerFile > "$SIMFILE"
    SIMULATION="$SIMFILE"
175
    #---------Selecting COOJA SCRIPT-----------
177 scriptFile="$NUMBEROFNODES-$DATimeWindow-t$totalTimeWindows.js"
    cat "$LOWPAN/general-sim.js" > "$scriptFile"
179 echo "$scriptFile file is created"

181 #---------Changes in COOJA SCRIPT for configurations--------
    timeout=$(($DATimeWindow*$totalTimeWindows*1000))
183 timeoutPlus=$(($DATimeWindow*$totalTimeWindows*1000+100000))

185 linetochange="MAIN-TIMEOUT"
    newlinewithcoojascript="$timeoutPlus"
187 awk -v var1="$linetochange" -v var2="$newlinewithcoojascript" '{
      gsub(var1,var2)
189   print
    }' $LOWPAN/$scriptFile > temp
191 mv temp $LOWPAN/$scriptFile

193 linetochange="TIME-OUT-FINALIZE"
    newlinewithcoojascript="$timeout"
195 awk -v var1="$linetochange" -v var2="$newlinewithcoojascript" '{
      gsub(var1,var2)
197   print
    }' $LOWPAN/$scriptFile > temp
199 mv temp $LOWPAN/$scriptFile

201 linetochange="nrNodes = 0;"
    numOfNodesForJS=$(($NUMBEROFNODES+1))
203 newlinewithcoojascript="nrNodes = $numOfNodesForJS;"
    awk -v var1="$linetochange" -v var2="$newlinewithcoojascript" '{
205   gsub(var1,var2)
      print
207 }' $LOWPAN/$scriptFile > temp
    mv temp $LOWPAN/$scriptFile
209
    linetochange="total_reports_needed = 0;"
211 numOfNodesForJS=$(($NUMBEROFNODES+1))
    newlinewithcoojascript="total_reports_needed = $NUMBEROFNODES;"
213 awk -v var1="$linetochange" -v var2="$newlinewithcoojascript" '{
      gsub(var1,var2)
215   print
    }' $LOWPAN/$scriptFile > temp
217 mv temp $LOWPAN/$scriptFile

219 linetochange="RATIO-TO-CHANGE1"
    newlinewithcoojascript="$ratio"
221 awk -v var1="$linetochange" -v var2="$newlinewithcoojascript" '{
      gsub(var1,var2)
223   print
    }' $LOWPAN/$scriptFile > temp
225 mv temp $LOWPAN/$scriptFile

227 linetochange="RATIO-TO-CHANGE2"
```

```
    newlinewithcoojascript="$ratio"
229 awk -v var1="$linetochange" -v var2="$newlinewithcoojascript" '{
      gsub(var1,var2)
231   print
    }' $LOWPAN/$scriptFile > temp
233 mv temp $LOWPAN/$scriptFile

235 echo "$scriptFile file is CHANGED"

237 #----------------Java script for COOJA SIMULATION-----------------
    linetochange="<scriptfile></scriptfile>"
239 newlinewithcoojascript="<scriptfile>[CONTIKI_DIR]/work/trendy-gm/$scriptFile</scriptfile>"

241 awk -v var1="$linetochange" -v var2="$newlinewithcoojascript" '{
      gsub(var1,var2)
243   print
    }' $SIMULATION > temp
245 mv temp $SIMULATION

247 #-------------Changing Simulation speed-----------------
    linetochange="<speedlimit>0.85</speedlimit>"
249 newlinewithcoojascript="<speedlimit>$ratio</speedlimit>"
    awk -v var1="$linetochange" -v var2="$newlinewithcoojascript" '{
251   gsub(var1,var2)
      print
253 }' $SIMULATION > temp
    mv temp $SIMULATION
255
    #---------------------Changes in other files-------------------------
257 workDIR="$CONTIKI/work"
    SAVEIFS=$IFS
259 IFS=$(echo -en "\n\b")
    for ((gm = 1; gm <= 5; gm++)); do
261
    case "$gm" in
263
        "1")
265         gmFile="$workDIR/trendy-gm/group-member.c"
            ;;
267     "2")
            gmFile="$workDIR/trendy-gm group-2/group-member.c"
269         ;;
        "3")
271         gmFile="$workDIR/trendy-gm group-3/group-member.c"
            ;;
273     "4")
            gmFile="$workDIR/trendy-gm group-4/group-member.c"
275         ;;
        "5")
277         gmFile="$workDIR/trendy-gm group-5/group-member.c"
            ;;
279     *)
            echo "Ending"
281         ;;
    esac
283 echo "selected file: $gmFile and APPUB = $appubThreshold"

285 linetochange="#define hit_count_threshold"
    newlinewithcoojascript="#define hit_count_threshold $appubThreshold"
287 sed -i "s/$linetochange.*/$newlinewithcoojascript/" $gmFile
    done
289
    IFS=$SAVEIFS
291
    #---------------Selection of RDC-------------------
293 workDIR="$CONTIKI/work"
    SAVEIFS=$IFS
295 IFS=$(echo -en "\n\b")
    for ((sa = 1; sa <= 11; sa++)); do
297
    case "$sa" in
299
        "1")
301         saFile="$workDIR/trendy-gm/project-conf.h"
            ;;
303     "2")
            saFile="$workDIR/trendy-gm group-2/project-conf.h"
305         ;;
        "3")
307         saFile="$workDIR/trendy-gm group-3/project-conf.h"
```

```
            ;;
309     "4")
            saFile="$workDIR/trendy-gm group-4/project-conf.h"
311         ;;
        "5")
313         saFile="$workDIR/trendy-gm group-5/project-conf.h"
            ;;
315     "6")
            saFile="$workDIR/trendy-gl/project-conf.h"
317         ;;
        "7")
319         saFile="$workDIR/trendy-gl group-2/project-conf.h"
            ;;
321     "8")
            saFile="$workDIR/trendy-gl group-3/project-conf.h"
323         ;;
        "9")
325         saFile="$workDIR/trendy-gl group-4/project-conf.h"
            ;;
327     "10")
            saFile="$workDIR/trendy-gl group-5/project-conf.h"
329         ;;
        "11")
331         saFile="$CONTIKI/examples/ipv6/rpl-border-router/project-conf.h"
            ;;
333     *)
            echo "Ending"
335         ;;
    esac
337 echo "selected file: $saFile and nullRDC = $nullRDC"

339 linetochange="#define NULLRDC"
    newlinewithcoojascript="#define NULLRDC  $nullRDC"
341 sed -i "s/$linetochange.*/$newlinewithcoojascript/" $saFile
    done
343
    IFS=$SAVEIFS
345
    #--------------Loading new seed values in a simulation-----------
347 linetochange="<randomseed>generated</randomseed>"
    newlinewithrandomseed="<randomseed>$newrandomseed</randomseed>"
349
    awk -v var1="$linetochange" -v var2="$newlinewithrandomseed" '{
351   gsub(var1,var2)
      print
353 }' $SIMULATION > temp
    mv temp $SIMULATION
355
    echo $SIMULATION" with random seed = $newrandomseed is starting"
357
    #Specific to Simulation
359 SDIR="$FDIR/$NUMBEROFNODES-$SIMNAMEFORMATTED-for-$simIterations-[$VMNo]"
    mkdir -p $SDIR
361 echo $SDIR" is created for Simulation record"

363 unset DISPLAY
    cd $LOWPAN
365


367 #Log file for Simulation's terminal output
    LOGFILE="$SDIR/sim$j.log"
369 echo "GIT ID: $simRef by VM#$VMNo" >>$LOGFILE
    echo "The Simulation Duration = $totalSimulationDuration" >>$LOGFILE
371 echo "Total Nodes $NUMBEROFNODES" >>$LOGFILE
    echo " of $j th of $simIterations iterations" >>$LOGFILE
373 echo "DA last for $DATotalTimeWindows time windows each with $DATimeWindow seconds" >>$LOGFILE
    echo "UA will query $queryInOneSet queries with interval of $queryInterval seconds" >>$LOGFILE
375 echo "UA will repeat above query set for $querySetRepeatTimes times each with
        $querySetInterval seconds interval" >>$LOGFILE

377 #Cleaning up before ending
    rm *.testlog
379 rm *.txt
    rm *.log
381 rm *.dat
    rm *.pcap
383
    #----------Starting and scheduling processes-----------------------
385 sleep 120 && sudo make connect-router-cooja >> $LOGFILE&
    pid_cooja_router=$!
```

```
sleep $UAStartTime && java -jar $UA $queryInOneSet $queryInterval $queryRandomInterval
    $querySetRepeatTimes $querySetInterval $ratio >> $LOGFILE&
pid_ua=$!

java -jar $DA $DATimeWindow $DATotalTimeWindows $DAMode $ratio $timerMin $timerMax
    $timerThreshold $timerStep >> $LOGFILE&
pid_da=$!
/usr/bin/time --verbose -ocoojatime.log -a java -jar $COOJA -nogui=$SIMULATION -contiki=
    $CONTIKI >> $LOGFILE&
pid_cooja=$!

wait $pid_cooja

sleep 10
kill -9 $pid_ua
kill -9 $pid_da
kill -9 $pid_cooja_router

#General detailed log
cat daDetail.txt >> "$SDIR/$SIMNAMEFORMATTED-da-detail.log"
cat daxtradetail.txt >> "$SDIR/$SIMNAMEFORMATTED-daFullDetail.log"
cat uacompleteLog.txt >> "$SDIR/$SIMNAMEFORMATTED-ua-detail.log"
cat COOJA.testlog >> "$SDIR/$SIMNAMEFORMATTED-Lowpan-detail.log"
cat daperformance.txt >> "$SDIR/$SIMNAMEFORMATTED-daperformance.log"
cat uaperformance.txt >> "$SDIR/$SIMNAMEFORMATTED-uaperformance.log"
cat uaperformance-processed.txt >> "$SDIR/$SIMNAMEFORMATTED-uaperformance-processed.log"
cat l-energy-all.dat >> "$SDIR/$SIMNAMEFORMATTED-l-energy-all.dat"
cat l-energy-ind.dat >> "$SDIR/$SIMNAMEFORMATTED-l-energy-ind.dat"
cat l-raw-energy-ind.dat >> "$SDIR/$SIMNAMEFORMATTED-l-raw-energy-ind.dat"
cat l-packet-all.dat >> "$SDIR/$SIMNAMEFORMATTED-l-packet-all.dat"
cat l-packet-ind.dat >> "$SDIR/$SIMNAMEFORMATTED-l-packet-ind.dat"
cat radio-packets.dat >> "$SDIR/$SIMNAMEFORMATTED-radio-packets.log"
cat "radiolog-"*".pcap" >> "$SDIR/$SIMNAMEFORMATTED-packets.pcap"
cat coojatime.log >> "$SDIR/$SIMNAMEFORMATTED-coojaTime.log"


#Saving Log for generating graphs
cat uaperformance.txt >> "$GDIR/case-$i-uaperformance.log"
cat uaperformance-processed.txt >> "$GDIR/case-$i-uaperformance-processed.log"
cat daperformance.txt >> "$GDIR/case-$i-daperformance.log"
cat l-energy-all.dat >> "$GDIR/case-$i-energy-all.dat"
cat l-energy-ind.dat >> "$GDIR/case-$i-energy-ind.dat"
cat l-raw-energy-ind.dat >> "$GDIR/case-$i-raw-energy-ind.dat"
cat l-packet-all.dat >> "$GDIR/case-$i-packet-all.dat"
cat l-packet-ind.dat >> "$GDIR/case-$i-packet-ind.dat"
cat radio-packets.dat >> "$GDIR/case-$i-radio-packets.log"
cat "radiolog-"*".pcap" >> "$GDIR/case-$i-packets.pcap"


#Cleaning up before ending
rm *.testlog
rm *.txt
rm *.log
rm *.dat
rm *.pcap

done #End of multiple iterations of one Simulation
done #End of one Unique Simulation - Loop for all simulations
sudo service dropbox start
```

A.2 Script for 6LoWPAN data gathering

Each COOJA simulation requires some Javascript to produce results in log files with different level of details. This automates the process of result gathering in a specified format for each experiment, which aids the process of debugging, validation and measurements of different performance metrics for 6LoWPANs. Following is an example of such a Javascript:

```
importPackage(java.io);

// Function to record statistics after each time window
```

```
function
print_stats()
{
total_energy_consumption = total_cpu_energy_consumption + total_lpm_energy_consumption
            + total_listen_energy_consumption + total_transmit_energy_consumption;

    log.log("----------------Time = "+ time +"------------------------\n");
    log.log("Total Nodes = " + (nrNodes-1) +":\n");
    log.log("Total reports = " + total_reports +":\n");
    log.log("Total Update Messages =" + total_updates_msgs + "\n");
    log.log("Total Grouping Messages =" + total_grouping_msgs + "\n");

    log.log("Total CPU Consumption =" + Math.round (total_cpu_energy_consumption)
                    + " [" + Math.round ((total_cpu_energy_consumption/
                        total_energy_consumption)*100, 5) + "%]");

    log.log("Total LPM Consumption =" + Math.round (total_lpm_energy_consumption)
                    + " [" + Math.round ((total_lpm_energy_consumption/
                        total_energy_consumption)*100, 5) + "%]");

    log.log("Total LISTEN Consumption =" + Math.round (total_listen_energy_consumption)
                    + " [" + Math.round ((total_listen_energy_consumption/
                        total_energy_consumption)*100, 5) + "%]");

    log.log("Total TRANSMIT Consumption =" + Math.round (total_transmit_energy_consumption)
                        + " [" + Math.round ((total_transmit_energy_consumption/
                            total_energy_consumption)*100, 5) + "%]");

    log.log("Total Energy Consumption =" + (total_energy_consumption)/1000  + " Joules [milli:
        "+total_energy_consumption);
    log.log("-------------------------------------------------------------\n");

gtime = time/1000000;
remainder = gtime%60;

if(remainder<200){
gtime = gtime-remainder;
}

// File to save aggregated energy statistics: lAllEnergyDescfile
writeinfile = gtime +"  "+total_energy_consumption +"   "+total_cpu_energy_consumption+"    "+
    total_lpm_energy_consumption +"   "+total_listen_energy_consumption+" "+
    total_transmit_energy_consumption+"\n";
output1[lAllEnergyDescfile].write(writeinfile);
log.log("lAllEnergyDescfile:"+ writeinfile);

// File to save aggregated packets statistics: lAllPacketDescfile
writeinfile = gtime +"  "+ (total_updates_msgs + total_grouping_msgs) + "   "+
    total_updates_msgs + "   "+ total_grouping_msgs +"\n";
output3[lAllPacketDescfile].write(writeinfile);
log.log("lAllPacketDescfile:"+ writeinfile);

}


TIMEOUT(MAIN_TIMEOUT);
/* override simulation speed limit to specified one*/
sim.setSpeedLimit(RATIO_TO_CHANGE1);

log.log("--------------------------------------------------------------------------------\n");
/* Configurations at start */
nrNodes = 0;
node_reported = new Array();
cpu_value = new Array();
lpm_value = new Array();
listen_value = new Array();
transmit_value = new Array();
energy_value = new Array();
updates_value = new Array();
grouping_value = new Array();
data = new Array();
total_updates_msgs = 0;
total_grouping_msgs = 0;
total_cpu_energy_consumption = 0;
total_lpm_energy_consumption = 0;
total_listen_energy_consumption = 0;
total_transmit_energy_consumption = 0;
total_energy_consumption = 0;
total_reports = 0;
total_reports_needed = 0;
```

```
all_reported = false;

output1 = new Object();
output2 = new Object();
output3 = new Object();
output4 = new Object();
output5 = new Object();
saNodes = nrNodes-1;
lAllEnergyDescfile =  "l-energy-all.dat";
lIndEnergyDescfile =  "l-energy-ind.dat";
lIndRawEnergyDescfile =  "l-raw-energy-ind.dat";
lAllPacketDescfile =  "l-packet-all.dat";
lIndPacketDescfile =  "l-packet-ind.dat";

nodes_starting = true;
for(i = 2; i <= nrNodes; i++) {
    node_reported[i] = false;
    energy_value[i] = 0;
    cpu_value[i] = 0;
    lpm_value[i] = 0;
    listen_value[i] = 0;
    transmit_value[i] = 0;
    updates_value[i] = 0;
    grouping_value[i] = 0;
}

// This loop will run till the end and wait for different messages to record them
while (true) {

// Creating different log files
if(!output1[lAllEnergyDescfile]){
output1[lAllEnergyDescfile] = new FileWriter(lAllEnergyDescfile);
slinetowrite =  "#Time: calculated_energy_consumption   calculated_cpu_energy_consumption
    calculated_lpm_energy_consumption   calculated_listen_energy_consumption
    calculated_transmit_energy_consumption"+ "\n";
//output1[lAllEnergyDescfile].write(slinetowrite);
log.log(slinetowrite);
}

if(!output2[lIndEnergyDescfile]){
output2[lIndEnergyDescfile] = new FileWriter(lIndEnergyDescfile);
slinetowrite =   "#Time:    i   energy_value[i] cpu_value[i]    lpm_value[i]    listen_value[i
    ] transmit_value[i]"+ "\n";
output2[lIndEnergyDescfile].write(slinetowrite);
log.log(slinetowrite);
}

if(!output3[lAllPacketDescfile]){
output3[lAllPacketDescfile] = new FileWriter(lAllPacketDescfile);
slinetowrite =   "#Time:    Total Trendy Packets:   UPD GROUPING"+ "\n";
output3[lAllPacketDescfile].write(slinetowrite);
log.log(slinetowrite);
}

if(!output4[lIndPacketDescfile]){
output4[lIndPacketDescfile] = new FileWriter(lIndPacketDescfile);
slinetowrite =   "#Time:    Node-ID:    Total Trendy Packets:   UPD GROUPING"+ "\n";
output4[lIndPacketDescfile].write(slinetowrite);
log.log(slinetowrite);
}

if(!output5[lIndRawEnergyDescfile]){
output5[lIndRawEnergyDescfile] = new FileWriter(lIndRawEnergyDescfile);
slinetowrite =   "#Time: i  raw_energy_value[i] raw_cpu_value[i] raw_lpm_value[i]
    raw_listen_value[i] raw_transmit_value[i]"+ "\n";
log.log(slinetowrite);
}


try{
    YIELD();
}catch(e){
        output1[lAllEnergyDescfile].close();
        output2[lIndEnergyDescfile].close();
        output3[lAllPacketDescfile].close();
        output4[lIndPacketDescfile].close();
        output5[lIndRawEnergyDescfile].close();
        log_func();
        throw('test script killed');
    }
```

```
/* Enforcing simulation speed limit to specified one */
sim.setSpeedLimit(RATIO_TO_CHANGE2);

// Recording error messages
if(msg.contains("off-link")){
nmsg=msg.substr(msg.indexOf("slip-bridge"));
log.log( "\n" + id +" at "+ time +  ":" + nmsg + "\n");
continue;
}
log.log( "\n" + id +" at "+ time +  ":" + msg + "\n");

// Recording Statistics message
if(!msg.contains("S:")){continue;}

    node_reported[id] = true;
    log.log( id +" at "+ time +  ":" + msg + "\n");
    data = msg.split(":");
    updates_value[id] = Number(data[1]);
    grouping_value[id] = Number(data[2]);
    cpu_value[id]= Number(data[3])
    lpm_value[id]= Number(data[4]);
    listen_value[id]= Number(data[5]);
    transmit_value[id]= Number(data[6]);
    energy_value[id]= cpu_value[id]+lpm_value[id]+listen_value[id]+transmit_value[id];


// Time is rounded
gtime = time/1000000;
remainder = gtime%60;

if(remainder<20){
gtime = gtime-remainder;
}

// Synchronisation with the DA after each time window
syncWithDA = new Object();
if(!syncWithDA["sync"]){
syncWithDA["sync"] = new FileWriter("sync.dat");
log.log( "Sync File Created");
}
syncWithDA["sync"].write(""+gtime+"");
syncWithDA["sync"].close();

// #1: lIndEnergyDescfile
log.log( id + " Before Conversion:[CPU,LPM,LSN,TRN]=[" + cpu_value[id] +" "+ lpm_value[id]+" "
     + listen_value[id]+" " + transmit_value[id]+ "]\n");

writeinfile = gtime +"  "+id +" "+energy_value[id]+"    "+cpu_value[id]+"   "+lpm_value[id]+"
       "+listen_value[id]+"    "+transmit_value[id]+"\n";
    output5[lIndRawEnergyDescfile].write(writeinfile);
    log.log("lIndRawEnergyDescfile:"+ writeinfile);

            cpu_value[id] = ((cpu_value[id]*(0.5*3.9))*3)/32768;
            lpm_value[id] = ((lpm_value[id]*(0.0545))*3)/32768;
            listen_value[id] = ((listen_value[id]*19.7)*3)/32768;
            transmit_value[id] = ((transmit_value[id]*17.4)*3)/32768;
        energy_value[id] =  cpu_value[id] + lpm_value[id] + listen_value[id] + transmit_value[
            id];
log.log( id +" After Conversion :[CPU,LPM,LSN,TRN=TOTAL]=[" + cpu_value[id] +" "+ lpm_value[id
    ]+" " + listen_value[id]+" " + transmit_value[id]+" ="+ energy_value[id]+"]\n");

    writeinfile = gtime +"  "+id +" "+energy_value[id]+"    "+cpu_value[id]+"   "+lpm_value[id
        ]+"   "+listen_value[id]+"    "+transmit_value[id]+"\n";
    output2[lIndEnergyDescfile].write(writeinfile);
    log.log("lIndEnergyDescfile:"+ writeinfile);

// #2: lIndPacketDescfile
    writeinfile = gtime +"  "+id +" " +(updates_value[id]+grouping_value[id]) +"    "+
        updates_value[id]+"   "+grouping_value[id]+"\n";
    output4[lIndPacketDescfile].write(writeinfile);
    log.log("lIndPacketDescfile:"+ writeinfile);

    total_reports++;

// Recording details till and reset after each time window
if(total_reports >= total_reports_needed) {
            all_reported = true;
  }
```

```
    if(all_reported){
227         all_reported = false;

229         for(i = 2; i <= nrNodes; i++) {
                total_updates_msgs += updates_value[i];
231             total_grouping_msgs += grouping_value[i];
                total_cpu_energy_consumption += cpu_value[i];
233             total_lpm_energy_consumption += lpm_value[i];
                total_listen_energy_consumption += listen_value[i];
235             total_transmit_energy_consumption += transmit_value[i];
            }
237
            print_stats();
239         total_updates_msgs = 0;
            total_grouping_msgs = 0;
241         total_energy_consumption = 0;

243 if(sim.getSimulationTimeMillis()>TIME_OUT-FINALIZE){
            log.log( "Ended at "+ time +  "\n");
245         output1[lAllEnergyDescfile].close();
            output2[lIndEnergyDescfile].close();
247         output3[lAllPacketDescfile].close();
            output4[lIndPacketDescfile].close();
249 log_func();
    SCRIPT_TIMEOUT();
251 }

253         for(i = 2; i <= nrNodes; i++) {
                node_reported[i] = false;
255             updates_value[i] = 0;
                grouping_value[i] = 0;
257             cpu_value[i]= 0;
                lpm_value[i]= 0;
259             listen_value[i]= 0;
                transmit_value[i]= 0;
261             energy_value[i]= 0;
                total_reports = 0;
263             total_updates_msgs = 0;
                total_grouping_msgs = 0;
265             total_energy_consumption = 0;
                total_cpu_energy_consumption = 0;
267             total_lpm_energy_consumption = 0;
                total_listen_energy_consumption = 0;
269             total_transmit_energy_consumption = 0;
            }
271
    }
273
}
```