# An Efficient Algorithm for Improved Energy Management in Software Defined Network (SDN)

S. Sathish[1]
*[1]Assistant professor,
Department of Computer Science and Engineering,
Sri Ranganathar Institute of Engineering and Technology.

C. Nivendiran[2], V.Muralidharan[3],
S. Kathiresan[4], S. Jeevanantham[5]
*[2,3,4,5] UG Scholar,
Department of Computer Science and Engineering,
Sri Ranganathar Institute of Engineering and Technology.

## Abstract

This paper we propose an efficient algorithm for improved energy management in software defined network. There are many algorithm to be used in energy consumption, But our proposed algorithm not only consuming battery life, it increase high throughput, life time, traffic load, and packet data ratio(PDR). The algorithm has to be finding the efficient path for routing, and to check the high energy node in routing. That node to identified, and re-routing the another path which is efficient one. Send a data from source to destination. This all to be done using the branch and bound algorithm for increasing throughput, battery life, packet data delivery(PDR).

**Keywords: SDN, PDR, DSR, branch and bound algorithm, re-routing, energy management.**

## I. INTRODUCTION

**Software Defined Network** is a wireless network. In traditional network has a collection of node that node to select a cluster head for some coverage. All the cluster head has separate control unit, this each cluster head to control all nodes under cluster head. In SDN has three layers which is application layer, control layer, infrastructure layer.

**Infrastructure layer**

It is the responsible for forwarding to the data.

**Control layer**

It controls the network view and overall monitoring process.

**Application layer**

It is an user interaction layer, to provide control layer for access the network.

The main purpose of SDN is to provide a centralized control unit for each cluster head. In traditional network has the main disadvantage in energy management, because of mobility in network.

## II. EXISTING SYSTEM

In SDN has issue in energy management. When the node to send a data from source to destination. The node send a route request for a neighbor node using DSR protocol.

### Dynamic Source Routing protocol

It is done by the following steps;

**step1:** To send a route request to the neighbor node.
**Step2:** To select a source and destination node.
**Step3:** Calculate the entire possible routing path.
**Step4**: To select an efficient path from all the possible routing path.
**Stpe5:** Send a data from selected routing path.

### Issues

In existing system has the issue in selecting a routing path. Is the node to select a routing path using DSR the selected path node should consume high energy. If that routing path was efficient also it was the issue to send a data. When a send a data to the efficient path, the data node may failure due to the loss of energy. So that we proposed the efficient algorithm for re-routing the path with low energy consumed.

## III. PROPOSED SYSTEM

We proposed the efficient algorithm in DSR protocol which is to select a routing path and to check a routing path, if any node to consume high energy and then re-routing the path which is

consumed low energy. Our proposed algorithm not only consuming the energy, it give a high throughput, traffic load, packet data ratio.

**Advantages**

- To select an efficient energy routing path.
- High throughput.
- Packet data ratio.
- Energy consumed.

**Algorithm explanation**

Our proposed algorithm was branch and bound algorithm.

A branch and bound algorithm is an optimization technique to get an optimal solution to the problem. It looks for the best solution for a given problem in the entire space of the solution. A branch and bound algorithm is similar to backtracking but is used to

Optimize the problems. We compute a bound on best possible solution that we can get if we down this node. If bound on best possible solution itself is worse than current best, then we ignore the sub tree rooted with the node. Apply the branch and bound algorithm to improve the routing path to efficiency energy and prolong of the network

Lifetime, throughput, PDR and minimize the overhead, delay.

**Step-1**

Generate and initial solution and nodes values are initialized and then source node is

forward the route request packets to neighbor nodes.

A = src

If current node is src

Then append current node broadcast the packet to neighboring nodes.

src – source node

**Step-2**

Calculate the lower bound of all the nodes in S by calling the bounding procedure

and non decreasing order of lower bound. Traverse the all new nodes. Forward packets or

given request and reply to next node. if the lower bound of current new node is smaller than

global upper bound.

min = S

A=src

B=dst

if src!=dst

srcd = sqrt((x2-x1)+(y2-y1)^2)

dstd = sqrt((x22-x11)+(y22-y11)^2)

if srcd<F And dstd<min

Then print the shortest path nodes

src – Source node, dst – destination node

x2 – source node x2 value, x1 – source node x1 value

y2 – source node y2 value, y1 – source node y1 value

x22 – destination node x22 value, x11 – destination node x11 value

y22 – destination node y22 value, y11 – destination node y1 value

S – minimum distance value, F – path finding set a fitness value.

**Step-3**

Calculate the lower bound of current node. Then initialize the maximum energy value

to compare the node energy value .In this step finding the highest energy value of the node.

max = S

For C < T

if e(a) < S

S=e(a)

at=a

Then forward the route request packet to routing path node

S – maximum energy value set for node, e(a) – energy value of the node

a –highest energy value of the node ,at – store the highest energy value of the node.

C – Current node, T – Total no of nodes.

**Step-4**

All branching procedures are completed. So find the highest energy value of the node

to travel the route path are broken so choose optimal path. Then terminating process to user

put source and destination to routing process are process.

F = 0

it=1

If a=at And F=0

Then break the path

Elseif

Then Choose the optimal path

F – Flag value for node count checking, it – interval time increase

a – Highest energy value of node, at – node energy value.

**Algorithm implementation**

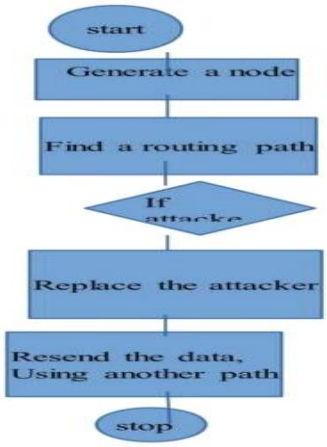


Fig : 1 algorithm implantation.

# IV. LIST OF MODULES

This is implemented in three modules. Module1 module2 module3

**Module1**

To generate a node and send a hello message for check active state. To generate a number of node for the user convenient. Set a range of each node which node is to placed in networking. To send a hello message for each neighbor node, this is to be check the node was active or not.

**Module2**

Send a route request for each node, find the routing path. After checking the active state, all the node to send a route request for all possible neighbor nodes. User to select a source and destination node which is from where the node to be start and end. After selecting source and destination it find the all the possible routing path.

**Module3**

Find the attacker node and re-routing the path. Using the branch and bound algorithm we check the routing path which node is consuming high energy is called Attacker. Because when the data send to that path the data may loss due to loss of energy. So our proposed algorithm to re-routing the path which is contain no high energy consuming path. Finally send a data from the efficient finding routing path.

**Graph for energy consumption**

```
BEGIN {
print "Markers: true" > "Energy.xg"
print "BoundBox: true" > "Energy.xg"
print "LabelFont: Monospace" > "Energy.xg"
print "TitleFont: Monospace" > "Energy.xg"
print "0 100" > "Energy.xg"
i=1
esum=0
avg=0
tim=0
itval=5
}
{
if($1=="N") {
if($3<tim) {
esum=esum+$7
i++
} else {
avg=esum/i
esum=0
i=1
if(avg!=0) {
print tim" "avg > "Energy.xg"
}
tim=tim+itval
}
}
}

END {

}
```

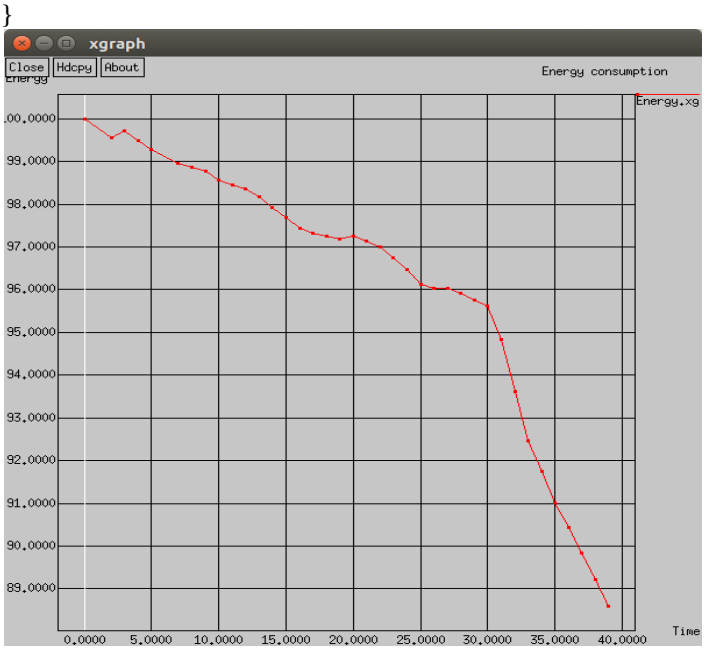


Fig :2 energy consumption graph

In this fig we set a initial energy level 100, so our proposed algorithm to consuming the energy level 100 to 40 shown in fig.

**Graph for high throughput:**

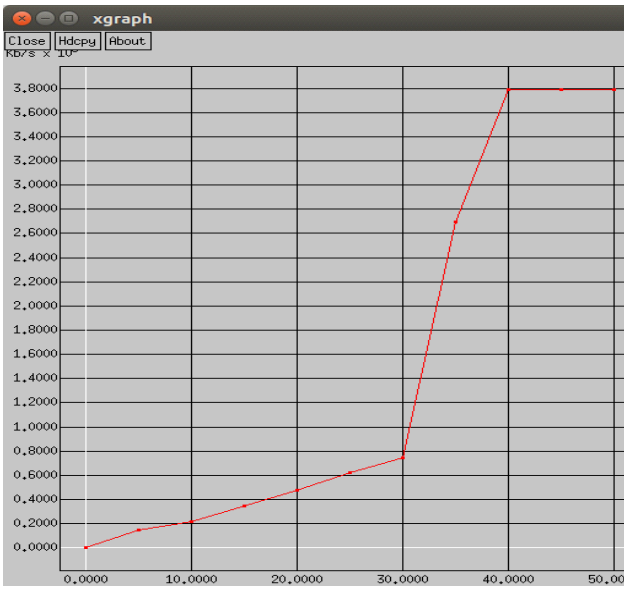


Fig.:3 High throughput graph

This graph to show the throughput level for using our proposed algorithm.

**Coding**

```
BEGIN {
print "LabelFont: Monospace" > "Throughput.xg"
print "TitleFont: Monospace" > "Throughput.xg"
print "Markers: true" > "Throughput.xg"
print "BoundBox: true" > "Throughput.xg"
t=0 #start time
e=50 #end time
ss=0
rr=0
dd=0
bits=0
tput=0
pdr=0
drops=0
itval=5
d=0
dst=0
engg=1
eng[engg,1]=0
eng[engg,2]=100
sum=0
{
if(FILENAME=="connect") {
src=$1
dst=$2}
if(FILENAME=="Trace.tr") {
if($1=="s" && $19=="AGT" && $37==512) {
s++
sent[s]=$3
#print $3 "\t" s}
if($1=="r" ) {
r++
res[r]=$3
byte[r]=$37
print $3 "\t" r
}
if($1=="d" && $19=="IFQ") {
d++
drop[d]=$3
#print $3 "\t" d}
if($1=="N") {engg++
eng[engg,1]=$3
eng[engg,2]=$7
eng[engg,3]=$5
#print $3 "\t" d}}}
END {
# Calculate sent
for(i=1;i<=s;i++) {
if(sent[i]<=t) {
ss=ss+1
}
else {while(sent[i]>t) {
sr[t,1]=ss
#print t "\t" ss > "sentP.xg"
t=t+itval
ss=0}
ss=ss+1  }}
sr[t,1]=ss
#print t "\t" ss > "sentP.xg"
ss=0
while (t<e) {
t=t+itval
#print t "\t" ss > "sentP.xg"}
# Calculate recieved and tput
t=0
for(i=1;i<=r;i++) {
if(res[i]<=t) {
rr=rr+1
bits=bits+(byte[i]*8)+30}
else {
tput=bits/itval/1000
while(res[i]>t) {
sr[t,2]=rr
print t "\t" tput > "Throughput.xg"
t=t+itvalrr=0#bits=0
#tput=0
}
rr=rr+1   bits=bits+(byte[i]*8)
}}tput=bits/itval/1000
sr[t,2]=rr
#print t "\t" rr
print t "\t" tput > "Throughput.xg"
#bits=0
```

```
#tput=0 rr=0
while (t<e) {
t=t+itval
print t "\t" tput > "Throughput.xg"
}
# Calculate drops
}
```

**Graph for packet data ratio**

**Coding**

```
BEGIN {
print "LabelFont: Monospace" > "PDR.xg"
print "TitleFont: Monospace" > "PDR.xg"
print "MARKERS: TRUE" > "PDR.xg"
print "0  9.25" > "PDR.xg"
t=5.0
d=10
itval=5.0
}
{if($3<=t && $32=="-Id") {
if($1=="r") {r--}
if($1=="d" && $21!="COL") {
d--}}
if($3>t && $32=="-Id") {
if($1=="r" ) {r--
}
if($1=="d")
print $1" "$21" "$23
if($1=="d" && $21!="COL") {
d--
}s=r+d
if(s!=0) {
pd1=s/r * 9.20 + 0.20
print t " \t"pd1 > "PDR.xg"}
t=t+itval }
}
END {
s=r+d
if(s!=0) {
pd1=r/s * 9.7
print t " \t"pd1  > "PDR.xg"}}
```

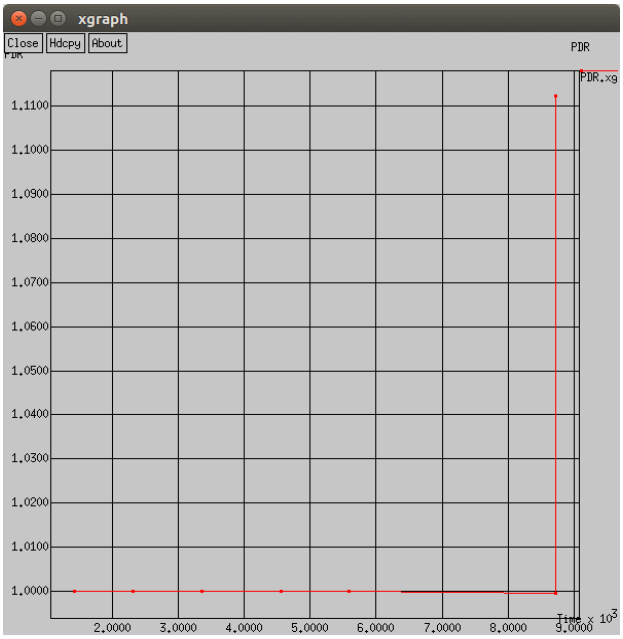


**Fig:4** packet data ratio diagram

# V. CONCLUSION

There are several algorithm to be used for energy efficient, but using our proposed algorithm only efficient because of consuming less energy for the routing node, and high throughput, and high packet data ratio.

# VI. REFERENCES

1] N. Feamster, J. Rexford, and E. Zegura, "The Road to SDN," Queue,
vol. 11, pp. 20:20–20:40, Dec. 2013.
[2] M. F. Bari, R. Boutaba, R. Esteves, L. Z. Granville, M. Podlesny,
M. G. Rabbani, Q. Zhang, and M. F. Zhani, "Data center network
virtualization: A survey," IEEE Communications Surveys & Tutorials,
vol. 15, pp. 909–928, Sept. 2013.
[3] S. Yu, M. Liu, W. Dou, X. Liu, and S. Zhou, "Networking for Big
Data: A Survey," IEEE Communications Surveys & Tutorials, vol. PP,
Sept. 2016.

[4] N. A. Pantazis, S. A. Nikolidakis, and D. D. Vergados, ''Energy-ef cient routing protocols in wireless sensor networks: A survey,'' IEEE