ISSN : 2087 - 0930

PERANCANGAN SIMULASI UNTUK EVALUASI EFEK MOBILITAS NODE BERBASIS VBVA PADA *UNDERWATER SENSOR NETWORK*

Eka Purwa Laksana[1], Riri Fitri Sari[2]

[1] Fakultas Teknik
Universitas Budi Luhur, Jakarta, 12260
Telp : (021) 5853753 ext 255
E-mail : eka.purwalaksana@budiluhur.ac.id

[2] Fakultas Teknik
Universitas Indonesia, Jakarta,
E-mail : riri@ui.ac.id

ABSTRAK

Underwater Sensor Network (UWSN) dirancang untuk memungkinkannya berkomunikasi dibawah air dengan menggunakan sinyal akustik sebagai pengiriman datanya. Pada thesis ini dilakukan pembangunan simulasi performa operasi routing Vector Based Void Avoidance (VBVA) untuk aplikasi mobilitas node pada jaringan UWSN.

Metode penelitian yang dilakukan adalah dengan membangun simulasi menggunakan aplikasi Network Simulator (NS-2) versi 2.30 yang berjalan pada sistem operasi Linux. Pada NS-2.30 yang diinstall ditambahkan modul aqua-sim yang dikembangkan oleh Peng Xie. Hasil output akhir pada NS-2 akan divisualisasikan berupa tabel dan grafik yang kemudian akan dianalisa lebih lanjut menggunakan metrik pengukuran packet delivery ratio, throughput, delay dan packet loss dengan menggunakan script AWK beserta beberapa tambahan modifikasinya.

Kata Kunci : UWSN, Routing, VBVA, Network Simulator

1. PENDAHULUAN

Bumi merupakan planet yang lebih dari 71% permukaannya ditutupi oleh laut, yang sebagian besar belum pernah diteliti. Sebagai salah satu cara untuk mengetahui lingkungan perairan, adalah menggunakan *Underwater Sensor Network* (UWSN). UWSN telah menjadi bahan riset yang menantang bagi para peneliti. Terdapat perbedaan antara UWSN dengan jaringan sensor *terestrial*.

Salah satu perbedaan mendasar antara UWSN dengan jaringan sensor *terestrial* adalah metode komunikasi. Jaringan *terestrial* menggunakan gelombang elektromagnetik atau gelombang radio untuk mengirimkan data. Sedangkan UWSN menggunakan sinyal akustik. Untuk penggunaan di bawah air, karakteristik sinyal akustik lebih baik dari pada gelombang radio atau gelombang elektromagnetik. Hal ini dikarenakan pada medium air, gelombang elektromagnetik dan radio tidak dapat mencapai jarak yang jauh. Sebagian sinyal akustik dapat mencapai jarak yang jauh walaupun dengan kecepatan dan *bandwidth* yang terbatas. Oleh karena itu, komunikasi akustik telah dianggap sebagai metode yang paling praktis untuk UWSN.

Perbedaan selanjutnya adalah node sensor bawah air biasanya akan bergerak karena arus air. Selanjutnya, jika dibandingkan dengan node sensor *terrestrial*, node bawah air biasanya akan jauh lebih besar, lebih banyak mengkonsumsi energi, sulit untuk mengisi ulang dan sehingga menjadi lebih mahal. Alasan biaya besar dan mahal ini yang menyebabkan penyebaran di area luas tidak akan banyak penuh. Ketika node disebarkan pada daerah tertentu dengan jarak antar node yang telah ditentukan, maka arus lah yang memungkinkan jarak antar node berubah. Dengan berubahnya posisi node dari posisi awal, maka skema *routing* pun akan berubah. Berubahnya skema *routing* ini akan mempengaruhi keberhasilan paket yang dikirimkan.

ISSN : 2087 - 0930

Pada penelitian ini penulis membangun sebuah simulasi untuk mengukur performa operasi *routing* untuk aplikasi mobilitas node pada UWSN dengan menggunakan *Network Simulator* 2 (NS-2)

2. TINJAUAN PUSTAKA

2.1 Ciri Saluran Underwater Sensor Network (UWSN)

Ciri yang paling besar dari *Underwater Sensor Network* (UWSN) adalah gelombang pembawanya menggunakan gelombang suara. Selain gelombang suara, gelombang radio dan optik pernah dicoba diterapkan pada UWSN. Namun, gelombang radio mengalami atenuasi tinggi saat digunakan di dalam air laut. *Radio Frequency* (RF) sinyal dapat merambat pada jarak jauh melalui media air hanya pada frekuensi extra rendah (30-300 Hz), yang memerlukan antena besar dan daya transmisi yang tinggi[5]. Oleh karena itu, gelombang radio menjadi tidak layak digunakan pada UWSN.

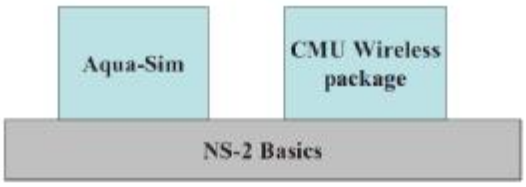
Untuk sinyal optik, kekurangannya adalah sinyal hanya dapat merambat pada jarak pendek dan *Line Of Sight*(LOS). Jangkauan transmisi maksimum gelombang optik adalah 8 meter pada air jernih (kolam renang). Namun dalam air yang sangat keruh dengan jarak pandang maksimum 1,4 meter, jangkauan maksimumnya adalah 1 m [6]. Maka, saluran komunikasi fisik yang paling tepat untuk UWSN adalah saluran akustik.

2.2 Aqua-Sim

Di dalam NS-2, paket *wireless* CMU dikembangkan untuk jaringan nirkabel *terrestrial*. Sedangkan untuk jaringan bawah air harus ditambahkan dengan aqua-sim [3], karena lingkungan bawah air berbeda dengan lingkungan *terrestrial*. Secara spesifik, jaringan bawah air mempunyai delay propagasi yang lama dan redaman yang tinggi. Model redaman saluran radio *terestrial* tidak dapat diterapkan lagi untuk saluran jaringan bawah air. Selain itu, delay propagasi yang lama membuat perilaku tabrakan sangat berbeda dengan perilaku tabrakan di jaringan radio yang membutuhkan cara yang berbeda untuk mensimulasikan tabrakan dalam jaringan sensor bawah air.

Di dalam NS-2, Aqua sim dipararelkan dengan paket simulasi *wireless Carnegie Mellon University* (CMU) seperti terlihat pada Gambar 1 [3].

Aqua-Sim tidak tergantung pada paket simulasi nirkabel dan tidak terpengaruh apabila paket nirkabel mengalami perubahan. Perubahan Aqua-Sim tidak akan mempengaruhi paket lain di NS-2, hanya berpengaruh terhadap Aqua-Sim nya saja. Dengan demikian, Aqua-Sim dapat berkembang secara bebas.



Gambar 1: Hubungan antara Aqua – Sim dan paket lainnya dari NS-2[3]

2.2.1 Routing Aqua-sim

Pelaksanaan semua protokol *routing* mengikuti standar protokol *routing* yang ada di NS-2. Parameter untuk protokol dapat diatur melalui skrip tcl. Ada tiga macam protokol *routing* pada jaringan bawah air yaitu, *Vector Based Forwading* (VBF), *Depth Based Routing* (DBR) dan *Q-Learning Based Routing* (QELAR).

Protokol VBF merupakan sebuah protokol *routing* geografis. Setiap node dalam jaringan diasumsikan mengetahui akan posisinya. Pada VBF, jalur *forwarding* mengikuti vektor dari sumber menuju target yang disebut *forwading vector*. Informasi posisi dari sumber, target dan pengirim dibawa dalam *header* dari paket data. Ketika node menerima paket, ia akan menghitung jarak *forwading vector*. Jika jaraknya kurang dari ambang batas yang telah ditentukan, yang disebut dengan radius, maka node ini memenuhi syarat untuk meneruskan paket. Di VBF, jalur *forwarding* seperti pipa dari sumber ke target, yang disebut *forwarding pipe*.

Protokol DBR merupakan protokol *routing* yang didasarkan pada algoritma *greedy forwarding* [13]. Ini menggunakan informasi kedalaman node sensor bawah air untuk mengirim paket dari node sumber menuju *sink* yang disebarkan di permukaan air. Paket mengikuti aturan mengurangi kedalaman node

ISSN : 2087 - 0930

forwarding pada setiap langkah *routing* menuju permukaan air. Dalam DBR, ketika paket diterima, node pertama mendapat informasi tentang kedalaman dari node sebelumnya yang disimpan dalam paket. Node penerima kemudian membandingkan kedalaman sendiri dengan kedalaman hnode sebelumnya. Jika kedalaman node penerima lebih kecil dengan kedalaman node sebelumnya, yang berarti node penerima lebih dekat dengan permukaan air, maka akan dianggap sebagai calon yang memenuhi syarat untuk meneruskan paket. Sebaliknya apa bila node penerima lebih jauh dengan permukan air maka akan membuang paket.

Protokol QELAR merupakan protokol *routing* yang adaptif, hemat energi, dan *lifetime-aware*. Dengan penggalian informasi yang dibutuhkan dari lingkungan bawah air pada saat dijalankan, agen belajar untuk mampu membuat keputusan yang optimal dalam hal routing untuk mengurangi jumlah hop ke *sink* serta membuat energi sisa didistribusikan lebih merata, dan oleh karena itu *lifetime* seluruh jaringan sebagian besar belangsung lama. Pada saat yang sama, protokol dapat dengan mudah diatur untuk menyeimbangkan konsumsi energi dan distribusi energi sisa untuk membuat dirinya lebih fleksibel.

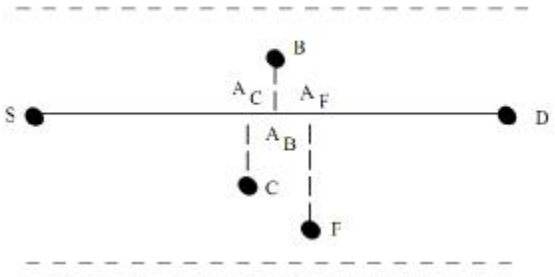
2.3 Vector-based Void Avoidance (VBVA)

VBVA merupakan pengembangan dari VBF dengan kemampuan untuk mengatasi masalah *void*. VBVA memiliki asumsi yang sama seperti VBF, yaitu node dapat mengetahui transmisi pada node sekitarnya. Hal ini dapat dengan mudah dipenuhi karena sifat *broadcast* dari saluran akustik bawah air. Sama halnya dengan VBF, jalur *forwarding* dari sebuah paket ini juga digambarkan sebagai vektor dan dibawa di dalam paket pada VBVA. VBVA akan berperilaku sama seperti VBF jika tidak ada *void* di jalur *forwarding*. Akan tetapi, VBVA secara signifikan berbeda dari VBF dalam hal VBVA mampu mendeteksi keberadaan *void* pada jalur *forwarding* dan melewati *void*. Dengan mekanisme *void avoidance*, VBVA berpotensi bisa menemukan beberapa vektor *forwarding* untuk sebuah paket data, sehingga secara signifikan meningkatkan ketahanan jaringan.

2.3.1 Mendeteksi Void

Node pada VBVA dapat mendeteksi adanya sebuah *void* dengan cara sengaja mendengarkan transmisi sebuah paket dari node sekitarnya. Pada VBVA, informasi akan vektor *forwarding* dari sebuah paket dibawa dalam paket. Saat node mendengarkan transmisi sebuah paket data, node akan menyimpan informasi posisi dari node *forwarding*.

Penggambarannya dapat diibaratkan titik awal dari vektor *forwarding* adalah S dan D sebagai titik akhir. Node akan disebut sebagai *nodevoid*jika semua peningkatan dari sekitarnya terhadap *forwarding* vektor paket tertentu kurang dari kemajuan sendiri. Seperti ditunjukkan pada Gambar 2, *forwarding* vektor dari sebuah paket adalah $\overrightarrow{SD}$, hasil pencapaian node B, C dan F pada *forwarding* vektor dilambangkan masing masing sebagai AB, AC dan AF.



Gambar 2 : contoh node void[2]

Terlihat bahwa semua node yang berdampingan dengan node F memiliki peningkatan yang kurang dari F pada *forwarding* vektor $\overrightarrow{SD}$, sehingga simpul F adalah *void node*. Dalam VBVA, jika sebuah node tahu kalau ia memiliki peningkatan terbesar pada saat *forwarding* vektor diantara semua node yang berdampingan, node akan menyimpulkan bahwa itu adalah *void node* dan mendeteksi *void* dalam *forwarding* vektor nya.

VBVA beda dengan VBF, dimana vektor *forwarding* dari sebuah paket didefinisikan sebagai vektor dari sumber ke *sink* dan dijaga tidak mengalami perubahan selama pengiriman paket. Pada VBVA, saat node *forwarding* mendeteksi *void* node mencoba untuk melewati *void* dengan mengganti vektor *forwarding* dari

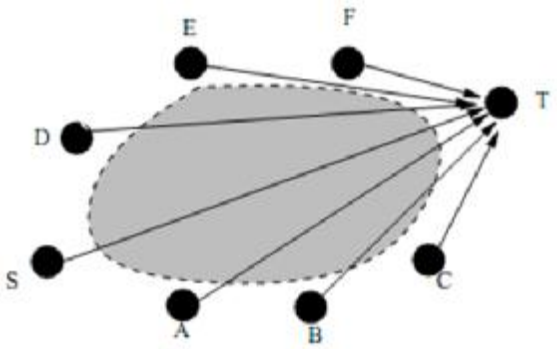
ISSN : 2087 - 0930

sebuah paket untuk menemukan rute alternatif. Yang digunakan dalam VBVA untuk melewati *void* adalah dengan dua mekanisme, yaitu *vector-shift* dan *back-pressure.*

2.3.2 Mekanisme Vector-Shift

Pada VBVA, ketika node menentukan bahwa itu adalah *void node* untuk sebuah paket, node akan mencoba melewati *void* dengan cara menggeser vektor *forwarding* dari sebuah paket. Untuk melakukan pergesaran vektor, node akan melakukan *broadcast* paket *vector-shift* ke node sekitarnya. Setelah menerima paket kotrol ini, semua node diluar pipa *forwarding* akan mencoba untuk meneruskan paket data yang sesuai mengikuti vektor *forwarding* yang baru yang berasal dari node tersebut menuju ke *sink.* Proses ini disebut sebagai *vector-shift* dan *void node* menggeser vektor *forwarding*.

Seperti terlihat pada Gambar 3, area yang bertanda garis adalah area *void.* Node S adalah node pengirim dan node T adalah node *sink*. Pada mulanya, node S mengirim paket sepanjang vektor *forwarding* $\overrightarrow{ST}$, lalu tetap menyimak jalur selama beberapa saat. Karena node yang berada disekitar node S yaitu node D dan A tidak berada dalam pipa *forwarding*, maka mereka tidak akan meneruskan paket ini. Dengan demikian, node S tidak dapat mendengar setiap tranmisi paket yang sama dan menyimpulkan bahwa node itu berada di pinggir *void* jaringan.



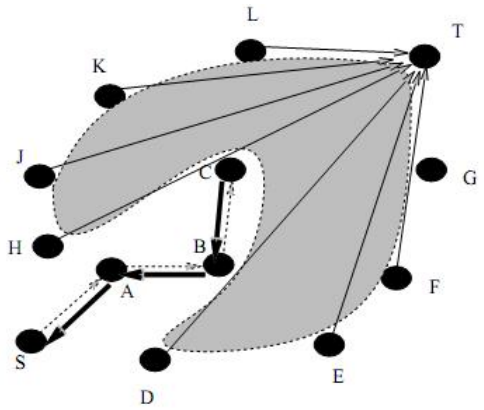
Gambar 3 : Mekanisme pergeseran vektor[2]

Maka dari itu kemudian, node mem*broadcast* paket kontrol *vector-shift,* meminta node sekitarnya untuk mengubah vektor *forwarding* semula menjadi $\overrightarrow{DT}$ dan $\overrightarrow{AT}$. Node D dan A akan mengulangi proses yang sama. Garis panah pada Gambar 3 adalah vektor *forwarding* yang digunakan oleh node *forwarding*. Dari gambar terlihat bahwa area *void* adalah cembung. Ini bisa dilewati dengan mekanisme *vector-shift*. Setelah menggeser vektor *forwarding* dari sebuah paket, node akan tetap menyimak jalur untuk mengetahui apakah ada node sekitarnya yang meneruskan paket dengan vektor *forwarding* yang baru. Jika node tidak mendengar paket diteruskan walaupun saat ini telah bergeser vektor *forwarding*, maka node diartikan sebagai node akhir. Untuk node akhir, mekanisme *vector-shift* tidak dapat menemukan jalur routing alternatif dan harus menggunakan mekanisme *back-pressure*.

2.3.3 Mekanisme Back-Pressure

Ketika sebuah node mengetahui bahwa itu adalah sebuah node akhir, node akan mem *broadcast* paket kontrol yang disebut paket BP (*Back-Pressure*). Ketika menerima paket BP, setiap node sekitarnya mencoba untuk menggeser vektor *forwarding* untuk paket yang sesuai jika sebelumnya belum pernah bergeser vektor *forwarding* paket ini. Jika tidak node akan mem*broadcast* paket lagi. Paket BP akan diarahkan kembali ke arah menjauhi dari *sink* sampai mencapai node yang dapat melakukan pergeseran vektor untuk meneruskan paket menuju *sink.*

ISSN : 2087 - 0930



Gambar 4 : Mekanisme back-pressure[2]

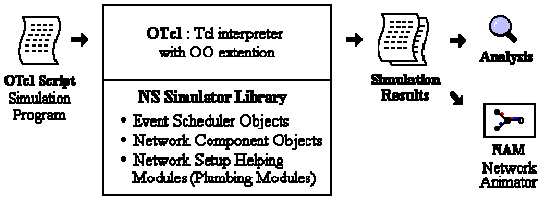
Gambar 4 memperlihatkan contoh proses *back-pressure*. Terlihat bahwa daerah yang gelap merupakan daerah *void* yang cekung. Node S adalah sebagai node pengirim dan node T adalah *sink.* Ketika node S mengirim paket melalui vektor *forwarding* $\overrightarrow{ST}$ menuju node C, karena node C tidak dapat meneruskan paket sepanjang vektor $\overrightarrow{ST}$ lagi, node C akan mencoba terlebih dahulu menggunakan mekanisme *vector-shift* untuk menemukan jalur alternatif untuk paket data. Karena node C adalah node terakhir, node C tidak dapat mendengar transmisi paket yang lain. Lalu node C akan mem *broadcast* paket BP. Setelah menerima paket BP, node B akan mencoba menggeser vektor *forwarding* terlebih dahulu, namun gagal untuk menemukan jalur untuk paket data. Kemudian node B akan mem *broadcast* paket BP ke node A dan seterusnya. Akhirnya, paket BP disalurkan dari node A ke sumber S. Kemudian Node S menggeser vektor *forwarding* ke $\overrightarrow{HT}$ dan $\overrightarrow{DT}$. Paket data tersebut kemudian diterukan ke *sink* dengan metode *vector-shift* dari node H dan D.

2.4 Network Simulator 2 (NS-2)

Network Simulator (NS) pertama kali dibangun pada tahun 1989 di University of California Barkeley (UCB). Pada tahun 1995 *Network Simulator*-2 (NS-2) pertama kali dikembangkan yang didukung oleh *Defense Advanced Research Projects Agency* (DARPA). NS-2 merupakan suatu sistem yang bekerja pada sistem operasi Unix/Linux.NS-2 dapat juga dijalankan pada sistem operasi Windows namun harus menggunakan *Cygwin* sebagai *enviroment* Linux-nya [8].

Pada dasarnya NS-2 adalah *interpreter* OTcl dengan objek *library network simulation*. Simulator ini mendukung hierarki *class* dalam C++ (*compiled hierarchy*) dan hierarki *class* yang serupa pada *interpreter* OTcl (*interpreted hierarchy*) [9]. Dua hierarki ini saling terkait satu sama lain. *Root* dari hierarki iniadalah Tcl Object. *User* membuat obyek Simulator (dari *class Simulator*) baru melalui *interpreter*, dan dicerminkan dengan obyek sebanding pada *compiled hierarchy* [9]. Setelah obyek Simulator dibuat, maka metode-metode untuk membuat topologi, node dan komponen jaringan lainnya dipanggil.

NS-2 juga merupakan simulator yang dipicu oleh *event* (*event-driven simulator*) [9]. *Scheduler* berjalan dengan cara mengeksekusi *event* berikutnya yang paling dahulu sampai selesai, kemudian kembali menjalankan *event* berikutnya. Walaupun jaringan berkomunikasi dengan melewatkan paket, namun itu tidak menghabiskan banyak waktu. Jika komponen yang dibuat memerlukan *delay*, maka digunakan *event scheduler* untuk menerbitkan *event* untuk paket tersebut dan menunggu *event* berhenti sebelum *event* berikutnya dijalankan [9].Flowchart sistem kerja NS secara global dapat dilihat pada Gambar 5.



Gambar 5:Flowchart Sistem Kerja NS-2 [14]

ISSN : 2087 - 0930

3. PERANCANGAN SISTEM

3.1 Skenario Simulasi

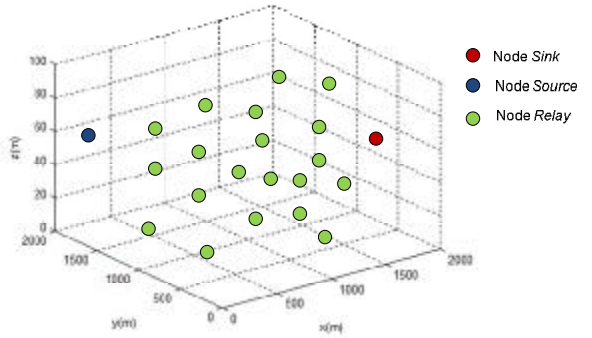
Dalam melakukan analisa kinerja *operasi routing* pada jaringan UWSN, penulis melakukan beberapa skenario dari berbagai kondisi lalu lintas dengan beberapa parameter variabel. Variabel parameter dalam simulasi ini adalah jumlah node dan kecepatan node.

Uji coba dilakukan dengan membuat sebuah area pengujian dengan dimensi 2000 m x 2000 m x 100 m. Didalam area pengujian akan ditempatkan sebanyak 10 sampai 50 node dengan perbedaan tiap pengujian 5 node. Penyebaran dan pergerakan node akan dilakukan secara acak. Mobilitas node akan bergerak dengan kecepatan 3 m/s dan 5 m/s. Untuk setiap kombinasi variabel parameter dijalankan sebanyak sepuluh kali percobaan untuk mendapatkan nilai rata-rata.

Skenario yang akan dijalankan pada saat simulasi ini digunakan untuk melihat pengaruh dari operasi *routing* VBVA untuk aplikasi mobilitas node. Skenario ini juga bertujuan untuk memperoleh pengaruh dari variasi jumlah node dan kecepatan node terhadap *packet delivery ratio, throughput*, *delay* dan *packet loss.*

3.1.1 Skenario Simulasi 1 : Lalu lintas sumber tunggal

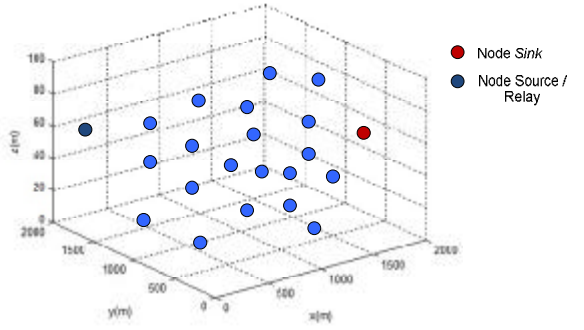
Dalam skenario pertama ini terdapat 1 node *source* dan 1 node *sink* yang diam di posisinya. Node sisanya akan berfungsi sebagai node *relay* yang akan disebar secara acak dan melakukan mobilitas secara acak pula. Node *relay* ini berfungsi sebagai node perantara dalam pengiriman paket dari node *source* yang dikirim menuju node *sink.* Topologi dari skenario pertama ini dapat dilihat pada Gambar 6.



Gambar 6 : Skenario pertama

3.1.2 Skenario Simulasi 2 : Lalu lintas beberapa sumber

Dalam skenario kedua terdapat 1 node yang berfungsi sebagai node *sink* dan node sisanya akan berfungsi sebagai node *source* yang juga bisa berfungsi sebagai node *relay*. Topologi dari skenario kedua ini dapat dilihat pada Gambar 7.



Gambar 7 : Skenario kedua

3.2 Paramater Simulasi UWSN

Untuk mendapatkan hasil simulasi yang tepat terhadap sistem yang akan dilakukan, maka perlu ditentukan beberapa parameter simulasi. Simulasi *operasi routing* pada jaringan UWSN menggunakan parameter-parameter yang diperlihatkan pada Tabel 1.

ISSN : 2087 - 0930

Tabel 1 : Parameter simulasi

Parameter	Nilai
Bit rate	10 Kbps
Frequensi	25 KHz
Jumlah node	50
Delay	25 µs
Antenna	Omni directional
Daya Tx	0,6 mWatt
Daya Rx	0,3 mWatt
Kecepatan node	3 m/s dan 5 m/s

3.3 Spesifikasi Perangkat

Dibawah ini spesifikasi perangkat yang akan digunakan dalam membangun simulasi jaringan ini:

a. InterfaceHardware

Hardware yang digunakan adalah 1 buah PC (laptop) dengan spesifikasi sebagai berikut:

- HP Pavilion G4
- Processor: AMD Dual-Core A4-3300M 1.9 GHz
- Memory: 4 Gb DDR3
- HDD: 500 GB

b. InterfaceSoftware

Simulasi ini harus berjalan baik pada spesifikasi software sebagai berikut:

- Sistem Operasi : Linux Ubuntu 12.04
- *Simulation Tool*: NS-2.30
- Text Editor: Notepad++
- Grafik : Gnuplot
-

3.4 Metrik Kinerja

Di dalam simulasi ini, ada beberapa ukuran kinerja yang diambil sebagai acuan dalam menganalisa kinerja operasi *routing* terhadap aplikasi mobilitas node pada jaringan UWSN. Ukuran atau metrik kinerja yang diukur pada penelitian ini antara lain:

1. ***Packet Delivery Ratio***

   Packet delivery ratio merupakan perbandingan banyaknya jumlah paket yang diterima oleh node penerima dengan total paket yang dikirimkan dalam suatu periode waktu tertentu. Atau bisa juga dihitung dengan cara mengurangi jumlah paket keseluruhan yang dikirim dengan paket yang *loss* atau hilang. Secara matematis *packet delivery ratio* dapat dicari dengan persamaan berikut :
   Packet delivery ratio (%) = (Σ Paket diterima / Σ Paket dikirim) x 100 %

2. ***Throughput***

   Throughput merupakan laju data aktual yang terukur pada suatu ukuran dalam waktu tertentu. Walaupun *throughput* memiliki satuan dan rumus yang sama dengan laju data, tetapi *throughput* lebih pada menggambarkan laju data yang sebenarnya (aktual) pada suatu waktu tertentu dan pada kondisi jaringan tertentu yang digunakan untuk men*download* suatu *file* atau data dengan ukuran tertentu. *Throughput* dapat dihitung dengan membagi jumlah paket data yang diterima dengan waktu yang dibutuhkan selama pengiriman data. Satuan dari *throughput* adalah Kbps atau Mbps.

   $$Throughput = \frac{Ukuran\ data\ yang\ diterima}{Total\ waktu\ pengiriman\ data}$$

3. ***Delay***

   Pada saat komunikasi akan terdapat *delay*, dalam hal ini adalah *delay* transmisi. Suatu paket data dari pengirim menuju ke penerima akan menghasilkan *delay* waktu yang berbeda tergantung parameter-

ISSN : 2087 - 0930

parameternya antara lain jarak kendaraandengan kendaraan yang lain, pergerakan dan kecepatan kendaraan dan masih banyak lagi yang dapat menyebabkan *delay* tersebut. *Delay Time* merupakan waktu yang diperlukan oleh suatu *node* untuk mengirimkan data ke *node* yang lain pada saat simulasi berlangsung.

4. ***Packet Loss***
Packet loss dapat diketahui dengan menghitung selisih jumlah paket yang dikirim dengan jumlah paket yang diterima. Pada perhitungan *packet loss* akan menghasilkan jumlah *packet loss* yang merupakan representasi jumlah paket yang terbuang atau tidak sampai pada *node sink*.

4. PENUTUP

1. Simulasi skema operasi *routing* untuk aplikasi *node mobility* pada *underwater sensor network* dilakukan dengan menggunakan *Network Simulator* 2 (NS-2).
2. Pengukuran kinerja operasi *routing* dari metode *routing VBVA* pada aplikasi mobilitas node dilakukan dengan menggunakan 4 parameter: *packet delivery ratio*, *throughput*, *delay*, dan *packet loss*.

REFERENSI

[1] E.H. Cherkaoui, L. Toni, L. Rossi, J.G. Fontaine, N. Agoulmine, “On the Effect of Node Mobility On Energy Eficiency and Delay Sensitive Applications In Underwater Sensor Networks, in Proc of MTS/IEEE OCEANS’11, June 2011.

[2] P. Xie, Z. Zhou, Z. Peng, J.H Cui, Z. Shi, “Void Avoidance In Three-Dimensional Mobile Underwater Sensor Networks”. in Proc of Wireless Algorithms, Systems, and Applications (WASA) 2009, Boston, USA, August 2009

[3] P. Xie, Z. Zhou, Z. Peng, H. Yan, T. Hu, J.H Cui, Z. Shi, Y. Fei, S. Zhou, “Aqua-Sim An NS-2 Based Simulator for Underwater Sensor Networks”, in Proc of MTS/IEEE OCEANS, pp. 1-7, 2009.

[4] M. Tran, M. Zuba, S. Le, Y. Zhu, Z. Peng, J.H. Cui, “Aqua-3D : An Underwater Network Animator”, in Proc of MTS/IEEE OCEANS’12, 2012.

[5] I.F. Akylidiz, D. Pompili, T. Melodia, “State of the Art in Protocol Research for Underwater Acoustic Sensor Networks”, in ACM International Workshop on Underwater Networks (WUWNet), September 2006.

[6] I. Vasilescu, K. Kkotay, D. Rus, M. Dunbabin, P. Corke, “Data Collection, Storage, and Retrieval with an Underwater Sensor Network”, in Proc SenSys ’05, pp 154-165, 2005

[7] P. Xie, J.H. Cui, L. Lao, “VBF : Vector-Based *Forwarding* Protocol for Underwater Sensor Networks”, in Proc of IFIP Networking pp 1216-1221, 2005

[8] “The Network Simulator ns-2”, http://www.isi.edu/nsnam/ns, diakses 22 Februari 2013.

[9] Jae Chung and Mark Claypool, “NS by Example”, Wurchester Polytechnic Insitute, http://nile.wpi.edu/NS, diakses 22 Februari 2013.

[10] T. Van Dam, K. Langendoen, “An Adaptive Energy-Efficient MAC Protocol for wireless sensor networks, in ACM SenSys’03, Los Angeles, California, USA, November 2003.

[11] “Nam Official website”, http://www.isi.edu/nsnam/nam/, diakses 22 Februari 2013.

[12] “Oceanviz”, http://blog.mikepan.com/post/19323379104/oceanviz-a-realtime - blender-underwater-visualization

[13] H. Yan, Z. Shi, J. H. Cui, “DBR: Depth-Based Routing for Underwater Sensor Networks”, in Networking 2008, 2008