BAB 2 LANDASAN KEPUSTAKAAN

Bab ini akan membahas kajian pustaka berdasarkan penelitian sebelumnya terkait dengan metode *file sharing* dan dasar teori yang terkait dengan penelitian yang dilakukan penulis.

2.1 Kajian Pustaka

Berdasarkan judul skripsi yang dibahas, penulis menggunakan penelitian-penelitian yang terkait yang telah dilakukan sebelumnya sebagai acuan. Penelitian terkait yang pertama adalah penelitian dari Pietro Braione (2002) yang berjudul "*A Semantical and Implementative Comparison of File Sharing Peer-to-Peer Application*". Dalam penelitian tersebut, dilakukan perbandingan pada aplikasi *peer-to-peer* terhadap fitur semantik dan fitur implementatif. Aplikasi yang digunakan dalam penelitian tersebut adalah Gnutella, Napster, Freenet, Free Haven, PAST, dan Publius. Penelitian tersebut menghasilkan perbandingan terhadap bagaimana *file* dipublikasikan, ditempatkan, dan dikirimkan pada setiap aplikasi *peer-to-peer* yang digunakan.

Penelitian lainnya yang dijadikan sebagai acuan dari penelitian ini adalah penelitian dari Vidal Martins(2007) yang berjudul "*Data Replication in P2P Systems*". Penelitian tersebut meneliti tentang replikasi data yang terjadi pada sistem jaringan *peer-to-peer* dan menjelaskan mekanisme replikasi data yang menggunakan berbagai macam arsitektur jaringan *peer-to-peer* yang berbeda. Protokol sistem yang digunakan dalam penelitian ini adalah Napster, JXTA, Gnutella, Chord, CAN, Tapestry, Pastry, Freenet, PIER, OceanStore, PAST, dan P-Grid. Penelitian ini menghasilkan solusi replikasi data yang memenuhi beberapa kebutuhan yaitu *Data Type, Autonomy, Replication Type, Conflict Detection, Consistency,* dan *Network Assumption* pada setiap protokol sistem yang digunakan.

2.2 File Sharing

File Sharing adalah suatu teknologi yang digunakan untuk melakukan distribusi konten data antara perangkat satu dengan perangkat lainnya yang saling terhubung dengan memanfaatkan protokol jaringan komputer sebagai media transfer konten data. Data yang didistribusikan adalah data *digital* seperti dokumen, multimedia, program komputer, *e-book*, gambar, dan grafis.

Berdasarkan metodenya, teknologi/aplikasi *file sharing* dibedakan menjadi dua, yaitu dengan menggunakan teknologi *peer-to-peer* atau *client-server*. *File Sharing* menggunakan *peer-to-peer* dilakukan oleh pengguna komputer yang bertindak sebagai *peer* dalam suatu jaringan komputer. *Peer* dapat melakukan *share* terhadap konten sehingga *peer* dapat berperan sebagai *client* sekaligus *server* (Department of Communications, Climate Action, & Environment, 2012).

Sedangkan pada c*lient-server* konten didapatkan dari *server* sebagai penyedia konten yang melibatkan suatu *websites* untuk melakukan *upload* konten data

pada *server*. *Client* dapat mengunduh konten data pada *server* dengan melakukan klik pada *link* yang tersedia pada *websites* yang menyediakan konten data (Department of Communications, Climate Action, & Environment, 2012).

File sharing yang dilakukan pada jaringan komputer tentu memiliki resiko-resiko yang dapat merugikan *users*. Resiko-resiko yang didapatkan dari penggunaan teknologi *file sharing* adalah (United States Computer Emergency Readiness Team, 2016):

- Installasi program berbahaya - saat menggunakan aplikasi *file sharing*, *attacker* dapat menyerang pengguna komputer dengan menyisipkan virus kedalam *file* yang di*share*.
- Pemaparan terhadap informasi pribadi
- Kerentanan terhadap serangan – aplikasi *file sharing* mungkin meminta pengguna untuk membuka *port* tertentu pada *firewall* untuk mengirimkan data. Tetapi *port* yang terbuka juga memberikan akses kepada *attackers* kepada komputer pengguna.
- *Denial of service* – kegiatan *file sharing* menyebabkan kepadatan terhadap *traffic* pada jaringan sehingga dapat membatasi akses pada internet atau mengurangi ketersediaan *resource* terhadap sejumlah program pada komputer.

2.3 *Peer-to-peer*

Peer-to-peer(P2P) adalah sebuah model jaringan alternatif terhadap model jaringan *client-server*. Jaringan P2P menggunakan model terpusat pada setiap mesin atau komputer yang disebut *peer* yang berfungsi sebagai *client* dan *server* sekaligus (The Government of Hong Kong Special Administration Region, 2008).

Berdasarkan karakteristik dari arsitekturnya P2P dibagi menjadi dua, antara lain (Androutsellis-Theotokis & Spinellis, 2004):

- *Sharing* yang dilakukan terhadap *resource* komputer secara langsung tanpa intermediasi dari *server* terpusat. *Server* terpusat terkadang dapat digunakan hanya untuk melakukan tugas-tugas tertentu seperti *bootstrapping* sistem, menambahkan node baru pada jaringan dan mendapatkan *global key* untuk enkripsi data.
- Kemampuan P2P dalam memperlakukan ketidakstabilan pada jaringan, menyesuaikan diri terhadap kegagalan pada koneksi jaringan, komputer, dan antar node secara otomatis.

Sedangkan berdasarkan *service* yang diberikan, P2P dibagi menjadi beberapa fungsi, yaitu (Camarillo, 2009):

- *Data Indexing Function*: fungsi yang digunakan untuk melakukan proses *indexing* pada data yang disimpan dalam sistem.

- *Data Storage Function*: fungsi yang digunakan untuk melakukan proses penerimaan dan penyimpanan data dari sistem.
- *Computation Function*: fungsi yang digunakan untuk melakukan proses komputasi pada sistem.
- *Message Transport Function*: fungsi yang digunakan untuk melakukan proses pertukaran pesan antar *peer*.

2.4 JXTA

JXTA adalah sebuah projek penelitian kolaboratif terkait dengan *peer-to-peer*(P2P) *computing*. JXTA merupakan kependekan dari *Juxtapose* (Gong, et al., 2002). JXTA adalah platform P2P yang dikembangkan dalam model Apache *open-source* oleh Sun Microsystem dibawah arahan dari Bill Joy dan Mike Clary. Tujuan dasar dibuatnya JXTA adalah sebagai berikut (Gradecki, 2002):

- Setiap *peer* dapat menemukan *peer* lainnya.
- *Peer* dapat melakukan *self-organize* menjadi *peer groups.*
- *Peer* dapat melakukan *advertise* dan *discover* terhadap *resources* dalam jaringan.
- *Peer* dapat berkomunikasi dengan *peer* lain.
- Platform dapat digunakan tanpa memerlukan penggunaan tertentu dari bahasa komputer atau sistem operasi.
- Platform dapat digunakan tanpa memerlukan penggunaan tertentu dari *authentication*, *security*, dan *encryption model*.

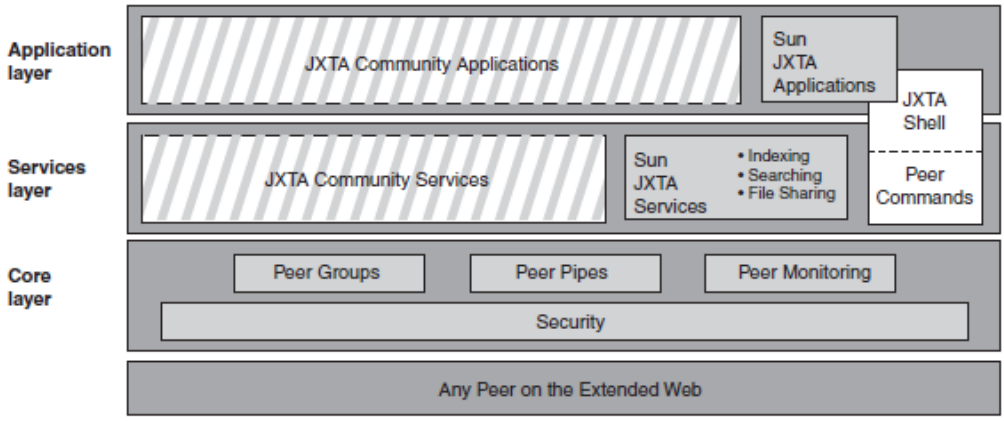
Pertukaran informasi pada jaringan protocol JXTA dilakukan dengan cara mengirimkan *advertisement* pada *peer* yang berperan sebagai *RDV/Relay* yang kemudian dikirimkan kembali pada *peer* lain yang membutuhkan informasi tersebut. *Advertisement* berupa dokumen XML yang berisi informasi *resources* dari setiap *peer* yang terhubung (Gong, et al., 2002).

Seluruh sistem JXTA dimodelkan menggunakan protokol-protokol kecil yang menangani *service* dari JXTA. Protokol-protokol ini dapat diimplementasikan menggunakan bahasa pemrograman apapun, sehingga memperbolehkan berbagai macam *device* untuk ada dan dapat berkomunikasi satu sama lain didalam sistem P2P yang besar. Didalam *core* JXTA terdapat enam protokol, yaitu sebagai berikut (Gradecki, 2002):

- *Peer Resolver Protocol*(PRP): digunakan untuk mengirim *query* kepada *peer-peer* untuk menerima sebuah respons.
- *Peer Discovery Protocol*(PDP): digunakan untuk melakukan *advertise* terhadap konten dan menemukan konten.
- *Peer Information Protocol*(PIP): digunakan untuk mendapatkan informasi terhadap status dari *peer*.

- *Pipe Binding Protocol*(PBP): digunakan untuk membuat jalur komunikasi antar *peer*.
- *Peer Endpoint Protoco*(PEP): digunakan untuk mencari rute dari satu *peer* ke *peer* lainnya.
- *Rendezvous Protocol*(RVP): digunakan untuk menyebarkan pesan didalam jaringan.

Agar protokol-protokol JXTA dapat berjalan pada sistem diperlukan adanya arsitektur. Arsitektur JXTA dapat dilihat pada Gambar 2.1 (Gradecki, 2002).



Gambar 2.1 Arsitektur *peer-to-peer* JXTA

Sumber: (Gradecki, 2002)

Core layer adalah tempat *code* untuk pengimplementasian protokol. Protokol-protokol ini menyediakan fungsionalitas untuk *peers*, *peer groups*, *security*, *monitoring*, *message-passing*, dan protokol jaringan. Pada *Service layer* terdapat *service-service* yang disediakan oleh JXTA. Sedangkan pada *Application layer* adalah tempat untuk pengembangan aplikasi *peer-to-peer* yang didalamnya terdapat *code-code* yang menarik *peer-peer* individu menjadi suatu fungsionalitas (Gradecki, 2002).

JXTA bekerja pada semua *layer* pada model OSI dimana *core layer* dari JXTA diimplementasikan pada *transport*, *network*, dan *data link layer*. Sedangkan pada *service* dan *application layer* diimplementasikan pada *session, transportation,* dan *application layer* (Airamo, 2005).

2.5 Gnutella

Gnutella adalah protokol *peer-to-peer* yang berbasis *open-source* dan *decentralized* yang penggunaannya diutamakan untuk melakukan pencarian *file* (Ripeanu, 2002). Gnutella juga merupakan protokol yang digunakan untuk melakukan *file sharing* pada jaringan *peer-to-peer* yang didesain dengan tujuan berikut ini (Ripeanu, 2002):

- Kemampuan untuk dapat beroperasi dalam sebuah *dynamic environment*. Gnutella harus dapat beroperasi pada sebuah *dynamic environment*, dimana

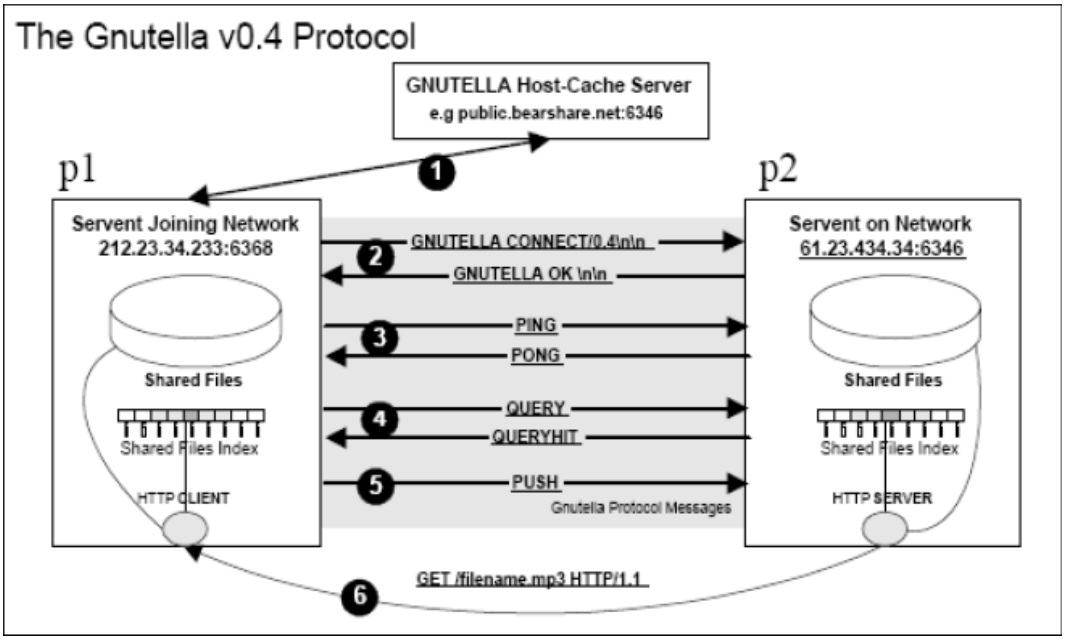
peer-peer memungkinkan untuk dapat bergabung atau meninggalkan jaringan yang terjadi berulang kali.

- *Performance and Stability*. Gnutella harus dapat menghasilkan performansi yang lebih kuat dibandingkan dengan model *client-server* saat dijalankan pada jaringan yang besar. Gnutella juga harus dapat menjaga stabilitas performansinya, terlebih saat pengguna dalam suatu jaringan bertambah banyak, *response time* yang dihasilkan harus tetap konstan dan *throughput* yang dihasilkan harus tetap tinggi.
- *Reliability*. Serangan dari luar seharusnya tidak dapat mempengaruhi kinerja Gnutella seperti kehilangan data penting atau berkurangnya performansi yang dihasilkan.
- *Anonimity*. *Anonimity* diperlukan untuk menjaga privasi dari seseorang yang mencari atau menyediakan informasi.

Protokol Gnutella hanya beroperasi pada *session layer* pada OSI model yang berguna untuk menemukan *peer* yang memiliki *resource* yang dicari oleh *peer* lain. Pertukaran informasi yang dilakukan antar *peer* dilakukan dengan metode *flooding* yaitu dengan meneruskan setiap respon yang diterima kepada *peer* lain yang terhubung secara terus menerus. Sedangkan pada pertukaran data pada jaringan protocol Gnutella menggunakan protocol HTTP (Airamo, 2005).

Untuk bergabung dalam jaringan Gnutella, *peer* harus terhubung pada satu *known-host* yang hampir selalu tersedia dalam jaringan(contoh: gnutellahost.com), selanjutnya *peer* mengirimkan pesan untuk dapat berinteraksi dengan *peer* lain. Pesan dapat dikirimkan dengan cara *broadcast* atau *back-propagate* yang difasilitasi oleh beberapa fitur yang disediakan oleh protokol. Fitur pertama, setiap pesan mempunyai *identifier* yang dibuat secara acak. Fitur kedua, setiap *peer* menyimpan *routed message* terbaru untuk mencegah terjadinya *re-broadcasting* dan mengimplementasikan *back-propagation*. Fitur ketiga, setiap pesan diberikan keterangan *time-to-live*(TTL) dan sebuah *field* yang berisikan *hop* yang telah dilalui. Pesan-pesan yang diperbolehkan didalam jaringan adalah seperti dibawah ini (Ripeanu, 2002) dan digambarkan pada Gambar 2.2 (Tribhuvan, 2007):

- *Group Membership*(PING dan PONG) *Messages.*
- *Search*(*Query* dan *Query Response*) *Messages.*
- *File Transfer*(GET dan PUSH) *Messages.*



Gambar 2.2 Arsitektur *peer-to*-peer Gnutella

Sumber: (Tribhuvan, 2007)

2.6 Wine

Wine adalah *compatibility layer* untuk menjalankan aplikasi Windows pada sistem operasi berbasis Linux, macOS, dan BSD. Wine bekerja dengan cara menerjemahkan API *calls* dari Windows menjadi POSIX *calls*, dengan menghilangkan penalti pada performa dan memory sehingga dapat terintegrasi dengan aplikasi Windows secara halus (Wine, n.d.). Keunggulan dari program Wine dibandingkan dengan Windows antara lain (Mesio, 2016):

- Wine memungkinkan untuk menggunakan kelebihan dari sistem operasi berbasis Unix yaitu *stability*, *flexibility*, dan *remote administration* saat menggunakan aplikasi Windows
- Wine memungkinkan untuk mengakses aplikasi Windows secara jarak jauh
- Wine dapat menghemat penggunaan secara ekonomi pada penggunaan terhadap *thin client*
- Wine dapat digunakan untuk membuat aplikasi Windows yang sudah ada tersedia pada Web dengan menggunakan VNC dan JAVA/HTML5 client
- Wine adalah *software* berbasis *open-source* jadi dapat digunakan sesuai dengan kebutuhan secara bebas

Dalam penelitian ini, Wine digunakan untuk menjalankan program/aplikasi yang berbasis sistem operasi Windows dapat berjalan pada sistem operasi yang

digunakan dalam penelitian yaitu Ubuntu. Wine akan diimplementasikan pada setiap VM yang digunakan untuk pengujian yang menggunakan sistem operasi Ubuntu LTS 14.04 sehingga aplikasi *file sharing* yang digunakan untuk pengujian, yaitu aplikasi Gnucleus dapat dijalankan.

2.7 Java Runtime Environment

Java Runtime Environment(JRE) merupakan *software* untuk mengembangkan aplikasi *Java* (Stoltzfus, 2017). JRE berfungsi sebagai platform yang diperlukan untuk menjalankan sebuah aplikasi *java* pada sistem operasi apapun termasuk Windows dan Linux. Didalam JRE terdapat komponen-komponen yang menunjang JRE agar dapat digunakan, yaitu antara lain (Stoltzfus, 2017):

- *Deployment technologies*, termasuk didalamnya ada *deployment*, Java Web Start dan Java Plug-in.
- *User Interface toolkits*, yang didalamnya terdapat Abstract Window Toolkit (AWT), Swing, Java 2D, Accessibility, Image I/O, Print Service, Sound, drag and drop (DnD), dan *input methods*.
- *Integration libraries*, yang mengandung Interface Definition Language (IDL), Java Database Connectivity (JDBC), Java Naming and Directory Interface (JNDI), Remote Method Invocation (RMI), Remote Method Invocation Over Internet Inter-Orb Protocol (RMI-IIOP) dan *scripting*.
- *Java Virtual Machine* yang didalamnya terdapat Java HotSpot Client dan Server Virtual Machines.

Dalam penelitian ini, JRE diperlukan untuk menjalankan aplikasi *file sharing* yang digunakan dalam pengujian yaitu myjxta karena aplikasi myjxta merupakan aplikasi berbasis java. JRE akan diimplementasikan pada setiap komputer dan VM yang digunakan pada pengujian aplikasi *file sharing* sehingga aplikasi dapat berjalan.

2.8 VMware Workstation

VMware Workstation adalah aplikasi untuk melakukan virtualisasi terhadap sistem operasi yang digunakan untuk membuat dan menjalankan *virtual machine* seperti *dekstop* dan *server* pada satu *physical machine* (Mohtasin, et al., 2016). Jumlah dan kemampuan *virtual machine* yang divirtualisasikan oleh VMware Workstation dibatasi dengan kemampuan *physical machine* yang digunakan seperti kapasitas *harddisk* dan *memory*. *Virtual machine* yang divirtualisasikan oleh VMware dapat membuat beberapa *network adapter* dari satu *physical network card* sehingga *resource* dapat dengan mudah didapatkan. Fitur-fitur yang diberikan oleh aplikasi VMware workstation adalah sebagai berikut:

- Dapat dijalankan dalam berbagai platform sistem operasi seperti Linux dan Windows

- Dapat menentukan sendiri spesifikasi dari *virtual machine* sesuai dengan kebutuhan yang dibatasi oleh kemampuan *physical machine*
- Menyediakan beberapa konfigurasi *network adapter* seperti Bridge, NAT, Host-only dan LAN segment
- Dapat melakukan *clone* terhadap *virtual machine* untuk menggandakan *virtual machine* yang sudah dibuat dengan konfigurasi yang sama
- Menyimpan keadaan sementara dari *virtual machine* dengan fitur Snapshot
- Menyediakan *virtual network editor* yang berfungsi untuk melakukan konfigurasi terhadap *network adapter* yang digunakan

Dalam penelitian ini, VMware Workstation digunakan karena keterbatasan komputer yang digunakan dalam pengujian dengan melakukan virtualisasi terhadap sistem operasi yang digunakan yaitu sistem operasi Ubuntu LTS 14.04. Virtualisasi sistem operasi Ubuntu dilakukan untuk menjalankan aplikasi yang digunakan untuk melakukan pengujian *file sharing*. VMware Worstation diimplementasikan pada setiap komputer yang digunakan dalam penelitian.

2.9 Wireshark

Wireshark adalah aplikasi berbasis *open-source* yang berguna untuk melakukan analisis paket data. Wireshark digunakan dalam jaringan untuk melakukan *troubleshooting* jaringan, melakukan pemeriksaan terhadap permasalahan *security* pada jaringan, melakukan *debug* terhadap implementasi suatu protokol, dan sebagai media pembelajaran cara kerja protokol pada suatu jaringan. Wireshark menyediakan fitur-fitur berikut seperti dibawah ini (Wang, et al., 2010):

- Mendukung sistem operasi Unix, Linux, dan Windows
- Mekanisme *capture* data secara *real-time* pada suatu *network interface* tertentu
- Membuka *file* berisikan hasil *capture* paket data menggunakan tcpdump/ Windump, Wireshark , dan beberapa aplikasi *capture* paket lainnya.
- *Import* paket yang berisi *hex dumps* dari suatu *file* berupa *text file*
- Menampilkan informasi detail dari protokol-protokol yang terdapat pada paket data
- Menyimpan hasil *capture* paket data
- *Export* paket data dalam sejumlah format *capture file*
- Melakukan *filter* terhadap paket dan menandai tampilan paket dalam berbagai macam kriteria yang dapat disesuaikan menurut kebutuhan
- Pencarian terhadap paket data yang telah di*capture*

- Menampilkan berbagai macam data statistik yang terkait dengan trafik jaringan komputer

Salah satu fitur yang disediakan oleh Wireshark yang digunakan pada penelitian ini adalah analisis terhadap paket data berdasarkan protokol transport yang digunakan yaitu TCP. Fitur ini akan menampilkan hasil *capture* terhadap trafik data masuk dan keluar pada suatu *network interface*. Hasil yang ditampilkan berupa informasi detail dari aliran paket data yang berjalan melalui *network interface* yang terdiri dari *source* IP *address*, *destination* IP *address*, *source port*, *destination port*, jumlah paket data, informasi paket data, *length* dari paket, dan lain sebagainya. Dengan menggunakan fitur ini, pengguna dapat mengetahui dan melakukan analisis terhadap kondisi trafik suatu jaringan pada *interface* yang digunakan saat suatu node melakukan komunikasi dengan node laininya.

Wireshark diperlukan untuk melakukan *capture* terhadap *traffic data* yang terjadi saat pengujian aplikasi *file sharing* berlangsung. Dari aplikasi wireshark didapatkan beberapa nilai yang akan dilakukan perhitungan sehingga didapatkan nilai untuk parameter-parameter yang sudah ditentukan sebelumnya. Implementasi wireshark akan dilakukan pada setiap komputer yang berfungsi sebagai penerima *file* pada saat pengujian aplikasi *file sharing* berlangsung.