BAB 1
USULAN GAGASAN

1.1 Latar Belakang Masalah

Dalam beberapa tahun terakhir ini arsitektur internet telah diusulkan sebagai alternatif untuk arsitektur TCP/IP. Arsitektur ini mengusulkan solusi untuk menangani keterbatasan masalah internet berbasis IP. *Named Data Networking* (NDN) adalah salah satu arsitektur yang mengusulkan desain arsitektur jaringan internet pada masa yang akan mendatang yang saat ini sedang dalam pengembangan, mengubah paradigma jaringan yang sebelumnya berfokus pada *host-centric* menjadi *content-centric* [1]. Pada arsitektur NDN paket data diberi *identifier* (pengenal) berupa nama dari konten tersebut, bukan alamat sumber penyedia konten data (*server*) ataupun tujuan. Berbeda dengan jaringan berbasis IP di mana setiap *device* harus mengetahui alamat IP dari penyedia konten data (*server*) tersebut.

Hal ini dapat terjadi dikarenakan pada *router* di arsitektur NDN memiliki algoritma *caching* yang berfungsi untuk melakukan duplikasi konten data yang sudah pernah diakses oleh pengguna (*client*) dari penyedia konten data (*server*) dan algoritma *forwarding* yang berfungsi untuk proses transfer data yang membuat arsitektur NDN dapat melakukan pertukaran informasi yang lebih efisien. Setiap node NDN mencakup *Content Store* (CS) yaitu memori untuk menyimpan konten, *Forwarding Information Base* (FIB) yaitu tabel dengan entri (awalan nama dan interface jaringan) untuk meneruskan *interest* (misalnya, tabel perutean), dan *Pending Interest Table* (PIT) yaitu tabel pencatatan pada *interface* mana untuk meneruskan data kembali ke pemohon [2].

Meskipun NDN memiliki protokol *routing* [3], namun permasalahan yang terjadi saat ini ialah memiliki *overhead* yang tinggi dikarenakan routing yang digunakan NDN saat ini bersifat *decentralized* yang setiap proses sinkronisasi dan *update* jalur menggunakan pengiriman paket secara *broadcast*. Hal ini untuk mengurangi *broadcast* dan menurunkan *overhead* dibuatlah sebuah kontroler yang kontroler ini memiliki jalur terpisah (*signaling*) dari jalur data. Dengan demikian, *overhead* yang dihasilkan oleh *routing centralized* lebih rendah jika dibandingkan dengan *routing decentralized.*

Selain itu, cara menggabungkan NDN dengan kontrol terpusat serta merancang keseluruhan arsitektur untuk mencapai kinerja dan *overhead* yang baik telah banyak dilakukan oleh peneliti-peneliti sebelumnya seperti [4]. Pada literatur Anwar dkk. [5] yang membahas cara penggabungan NDN dengan SDN agar dapat meningkatkan skalabilitas dengan

memisahkan *data plane* dan *control plane.* Fungsi *control plane* terpusat membuat mekanisme kompleks lebih mudah diterapkan. Maka dari itu, ide inti penelitian Tugas akhir ini adalah memusatkan kontroler untuk mengurangi *overhead* pada *routing*

1.2 Informasi Pendukung Masalah

Hal yang didesain untuk mengelola dan mengontrol sistem *centralized* dapat digunakan teknologi arsitektur *Software Defined Network* (SDN). SDN merupakan paradigma pengembangan jaringan teknologi baru untuk menyelesaikan masalah pada mekanisme jaringan. SDN mampu melakukan kontrol terpusat yang memisahkan *control plane* dengan *data plane. Control-plane* berfungsi untuk mengatur dan mengendalikan aliran data jaringan sedangkan *data plane* berfungsi untuk memindahkan data dari satu *end-point* ke *end-point* lainnya [6]. Tidak seperti solusi berbasis SDN, dalam Tugas Akhir kami ini hanya meneruskan paket *named data* dan menghilangkan ketergantungan IP pada *router* untuk berkomunikasi.

1.3 Analisis Umum

1.3.1 Aspek Teknologi

NDN adalah rancangan arsitektur jaringan internet masa depan yang mampu mengubah sudut pandang dalam jaringan yang sebelumnya *host-centric* menjadi *data-centric*.

1.3.2 Aspek Sosial

Perkembangan teknologi infromasi dan komunikasi telah menyebabkan hubungan dunia menjadi tanpa batas dan saat ini menjadi pedang bermata dua.

1.3.3 Aspek Keberlanjutan

Dengan adanya kontrol terpusat pada arsitektur jaringan NDN membuat *overhead* yang dihasilkan oleh routing *centralized* akan lebih rendah jika dibandingkan dengan *routing decentralized.*

1.4 Kebutuhan yang Harus Dipenuhi

Penggunaan jaringan kontroler secara terpusat diperlukan pengetahuan mengenai topologi jaringan secara global. Oleh karena itu, setiap *node* mengirimkan informasi yang berisi *node* tetangganya, sehingga kontroler mengetahui informasi yang berkaitan dengan *node* tersebut. Selain itu dengan adanya kontroler terpusat dapat memanfaatkan *link* secara maksimal serta meningkatkan nilai *Throughput* dan RTT pada jaringan tersebut.

1.4.1 Spesifikasi Hardware

Pembuatan desain pada penelitian Tugas Akhir ini menggunakan perangkat keras laptop pribadi dengan spesifikasi yang memadai untuk menjalankan virtualisasi.

1.4.2 Spesifikasi Software

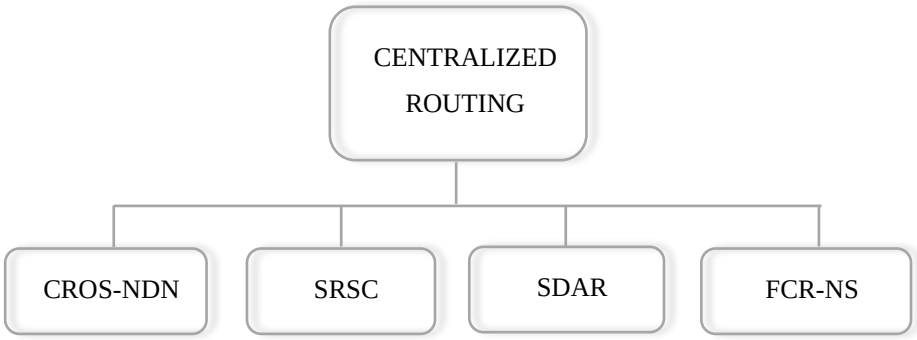
Pembuatan desain pada penelitian Tugas Akhir ini mengunakan perangkat lunak dengan emulasi jaringan *Packet Network Emulator Tool Lab* (PNETLAB).

1.5 Tujuan

Penelitian ini bertujuan untuk menurunkan *overhead* yang tinggi dikarenakan *routing* saat ini bersifat *decentralized*. Sehingga dibuatlah sebuah kontroler terpusat.

1.6 Solusi Sistem yang Diusulkan

Dalam beberapa tahun terakhir paradigma baru jaringan terpusat telah mendapat daya tarik bagi peneliti-peneliti sebelumnya yang mana kontrol jaringan didistribusikan ke kontrol terpusat dengan memisahkan *control plane* dan *data plane*. Pada Tugas Akhir kami ini menawarkan 4 solusi sistem yang dapat diusulkan untuk *Centralized NDN* seperti yang tertera pada Gambar 1.1 [4] dan berikut penjabaran dari setiap solusi.



Gambar 1. 1 Skema Centralized NDN

1.6.1 Karakteristik Produk

1.6.1.1 CRoS-NDN

Skema CRoS-NDN ini memanfaatkan paradigma SDN itu sendiri yang hanya meneruskan *named data* dan menghilangkan ketergantungan pada IP untuk *router*. Simulasi CRoS-NDN ini menggunakan ndnSIM. Hasil penelitian ini dapat mengurangi perutean yang *overhead* dengan membatasi jaringan *interest flooding*, meningkatkan efisiensi mobilitas konten produsen serta dapat memantau *node* tetangga satu *hop*-nya. Selain itu CRoS-NDN dapat menambahkan tindakan khusus untuk menghapus aturan *forwarding* yang tidak valid di FIB

setelah entri PIT *timeout*, yang mana fitur NDN *default* tidak memiliki sarana protokol perutean untuk memberikan umpan balik berdasarkan kedaluwarsa PIT [7]

1.6.1.2 SRSC

Skema perutean berbasis SDN untuk NDN ini menggunakan pesan data dan membangun saluran komunikasi antara pengontrol dan *node*. SRSC ini telah beroperasi di lingkungan NDN asli (yaitu tanpa TCP/IP). Skema ini menggunakan *Testbed Experiment Docker* dan di implementasikan ke dalam NDNx dengan menggunakan Topologi Abilene menggunakan 4 skenario mendapatkan hasil penelitian yang dapat mengurangi panjang jalur untuk mengakses konten, meneruskan *interest* ke *node* terdekat serta kinerja *cache* yang tinggi dibandingkan dengan strategi *forwarding* NDN default [2].

1.6.1.3 SDAR

Model komunikasi SDAR diimplementasikan oleh paradigma SDN dalam jaringan berbasis IP, tetapi SDAR ini sudah bekerja di atas protokol NDN serta menggunakan paket NDN untuk berkomunikasi satu sama lain. Sistem SDAR terdiri dari dua jenis komponen yaitu; *controller* dan *node*. Skema ini menggunakan ndnSIM untuk merancang dan mengimplementasikan model komunikasi SDAR, menggunakan modul 'topologyReader' di ndnSIM yang menghasilkan enam *node* yang akhirnya mendapatkan hasil penelitian yang dapat mengurangi *overhead* lalu lintas ke jaringan dan pengontrol, meningkatkan kecepatan respon setelah perubahan topologi dan kegagalan link dengan bantuan *adaptive* serta *forwarding* multijalur [8].

1.6.1.4 FCR-NS

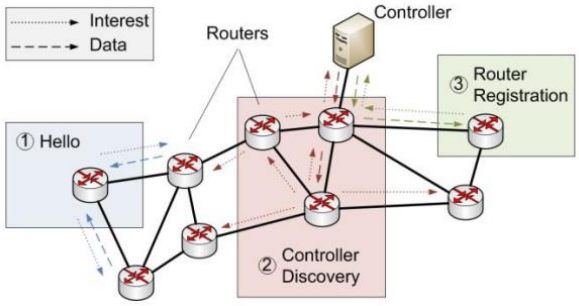
FCR-NS adalah protokol perutean berbasis SDN yang menggunakan kebijakan penggantian *cache* baru yang menghitung popularitas data lokal di *switch* dan struktur *filter bloom*, untuk mempercepat *forwarding*. Skema ini menggunakan ndnSIM dalam pengerjaanya mendapatkan hasil penelitian yang dapat melakukan penggantian *cache* yang sangat efektif berdasarkan perhitungan *real-time* dari popularitas data oleh *switch*, proses perutean intra-zona yang dikelola oleh pengontrol SDN sangat cepat Begitupun dengan perutean antar zona dengan menggunakan *filter bloom* di pengontrol SDN [5].

1.6.2 Skenario Penggunaan

1.6.2.1 CRoS-NDN

CRoS memperkenalkan elemen jaringan khusus yang disebut pengontrol yang bertanggung jawab atas perutean lokasi data bernama. Skema ini terdari dari 2 strategi [9] yaitu:

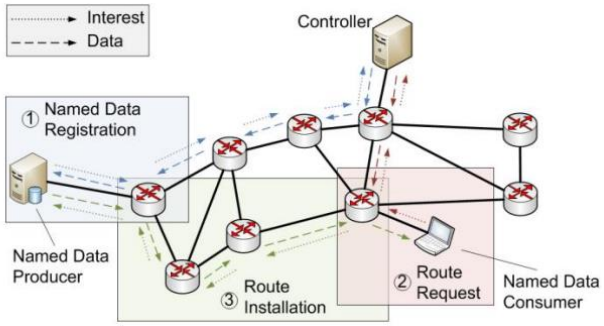
1. Fase Bootstrap



Gambar 1. 2 Fase Bootstrap CRoS-NDN

Fase ini bertujuan memonitor *router* untuk membangun pengetahuan tentang topologi jaringan global. Penggambaran *flow* pada fase ini seperti pada Gambar 1.2 [9] yang memiliki tiga tahapan yaitu *Hello Protocol*, *Controller Discovery*, dan *Router Registration*.

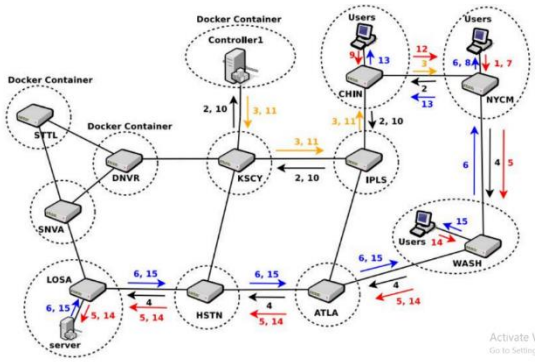
2. Fase Named Data Routing



Gambar 1. 3 Fase Named Data Routing CRoS-NDN

Fase ini menjamin lokalisasi dan akses ke konten yang diminta. Penggambaran *flow* pada fase ini seperti pada Gambar 1.3 [9] yang memiliki tiga tahapan *Named Data Routing*, *Route Installation*, dan *Route Request*.

1.6.2.2 SRSC



Gambar 1. 4 SRSC diimplementasikan ke dalam NDN

Topologi yang digunakan pada metode SRSC ini adalah Topologi Abilene yang diterapkan pada *testbed experiment docker* seperti yang terlihat pada Gambar 1.4. SRSC bergantung pada dua fase berbeda untuk mencapai tujuannya [2] yaitu:

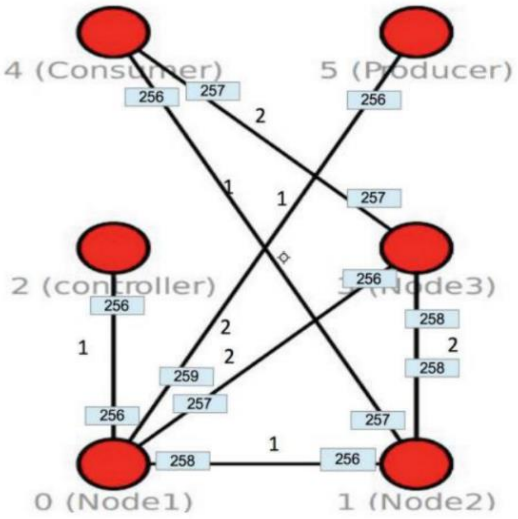
1. Bootstrap

   Fase *bootstrap* ini untuk mengetahui *node* ke pengontrol, sehingga pengontrol mendapatkan bentuk topologi jaringan.

2. Forwarding

   Fase ini ialah langkah penting untuk *forward interest* yaitu, meminta aturan ke pengontrol, menambah aturan ke dalam *node* FIB dan meneruskan *interest* ke CS terdekat.

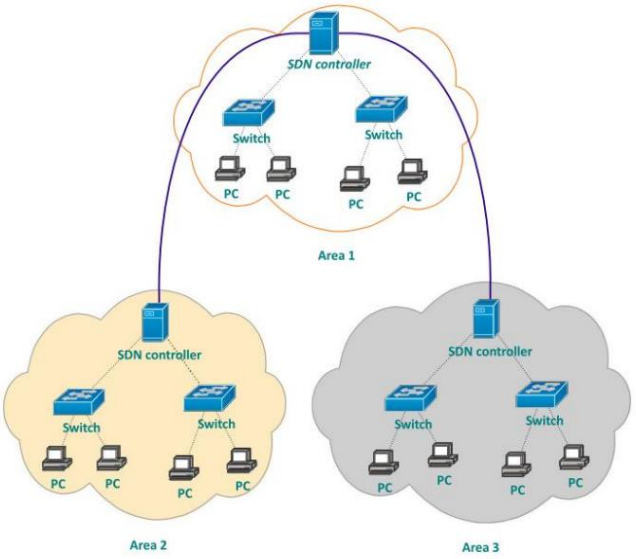
1.6.2.3 SDAR



Gambar 1. 5 SDAR Topologi Jaringan

SDAR diimplementasikan berdasarkan paradigma SDN dan dirancang untuk memisahkan kontrol perutean dari *forwarding*. Metode pendekatan yang dilakukan pada SDAR yaitu setiap *node* memulai komunikasi yang berfungsi memberi tahu *controller* untuk melakukan pengambilan data dengan cara mengirimkan *request interest*. Setelah *controller* menerima notifikasi, ia akan menerima *request* dengan paket data yang kosong untuk memenuhi kebutuhan *interest* untuk mengambil data dari *node*. Pada model komunikasi ini, meskipun menimbulkan beban lalu lintas tambahan, akan lebih efisien jika jaringan tidak mengalami perubahan *link* dan pembaruan topologi [8].

1.6.2.4 FCR-NS



Gambar 1. 6 FCR-NS Topologi Jaringan

FCR-NS adalah protokol perutean berbasis SDN yang menggunakan kebijakan penggantian *cache* baru yang menghitung popularitas data lokal di *switch*. Penggunaan arsitektur SDN ini, bidang data dipisahkan dari bidang kontrol dan menggabungkannya dengan *filter bloom*. Kemudian setiap *switch* membuat keputusan untuk mengganti data secara mandiri, berdasarkan permintaan yang dibuat oleh *node* langsung [5].

1.7 Kesimpulan dan Ringkasan CD-1

Named Data Networking (NDN) adalah arsitektur yang mengubah paradigma jaringan yang sebelumnya berfokus pada *host-centric* menjadi *content-centric* [1]. Meskipun NDN memiliki *protocol routing* [3], namun persmasalahan yang terjadi saat ini ialah memiliki *overhead* yang tinggi dikarenakan routing yang digunakan saat ini bersifat *decentralized*. Oleh sebab itu dibuatlah kontroler terpusat untuk menurunkan *overhead* dan mengurangi *system broadcast.*

Selain itu, cara menggabungkan NDN dengan kontrol terpusat serta merancang keseluruhan arsitektur untuk mencapai kinerja dan *overhead* yang baik telah banyak dilakukan oleh peneliti-peneliti sebelumnya [4]. Hal yang didesain untuk mengelola dan mengontrol sistem *centralized* dapat digunakan teknologi arsitektur *Software Defined Network* (SDN). SDN merupakan paradigma pengembangan jaringan teknologi baru untuk menyelesaikan masalah pada mekanisme jaringan. SDN mampu melakukan kontrol terpusat yang memisahkan *control plane* dengan *data plane* [6].