



LAPORAN PRAKTEK KERJA LAPANGAN

APLIKASI PACKET SNIFFER DENGAN BAHASA PYTHON

Oleh:

I PUTU KUSWARA ADI PRADANA

NIM : 1308605017

Pembimbing:

I DEWA MADE BAYU ATMAJA DARMAWAN, S.KOM., M.CS.

Program Studi Teknik Informatika

Jurusan Ilmu Komputer

Fakultas Matematika Dan Ilmu Pengetahuan Alam

Universitas Udayana

2016

HALAMAN PENGESAHAN
APLIKASI PACKET SNIFFER DENGAN BAHASA PYTHON

Oleh:

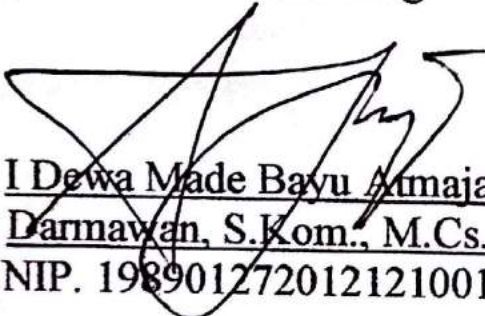
I Putu Kuswara Adi Pradana

NIM : 1308605017

Bukit Jimbaran, 5 Januari 2017


Menyetujui,

Dosen Pembimbing




I Dewa Made Bayu Atmaja
Darmawan, S.Kom., M.Cs.
NIP. 198901272012121001

Pembimbing Lapangan



I Gede Eka Gatria Atitama
S.Kom., M.Kom.
NIP. 1991022620160312001

Penguji



Luh Arida Ayu Rahning
Putri, S.Kom., M.Cs
NIP. 198209182008122002

Mengetahui,

Ketua Jurusan Ilmu Komputer
FMIPA Universitas Udayana



Agus Muliantara, S.Kom., M.Kom.
NIP. 198006162005011001

KATA PENGANTAR

Puji dan syukur dipanjatkan kehadiran Tuhan Yang Maha Esa karena atas segala berkat dan karunia-Nya sehingga dapat terselesaikannya laporan praktek kerja lapangan (PKL) dengan judul “Aplikasi Packet Sniffer Dengan Bahasa Python”.

1. Bapak Agus Muliantara, S.Kom., M.Kom. selaku ketua Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Udayana
2. Bapak I Dewa Made Bayu Atmaja Darmawan, S.Kom., M.Cs. selaku pembimbing yang telah memberikan bimbingan, arahan, dan masukan selama penyusunan laporan ini.
3. Bapak I Gede Oka Gatria Atitama, S.Kom., M.Kom., selaku pembimbing lapangan yang telah memberikan bimbingan dan arahan selama kegiatan praktek kerja lapangan.
4. Semua rekan – rekan Praktek Kerja Lapangan di lingkungan Jurusan Ilmu Komputer yang mendukung dan memberikan saran – saran kepada penulis selama melakukan Praktek Kerja Lapangan
5. Semua pihak yang telah membantu hingga laporan ini dapat terselesaikan.

Disebabkan keterbatasan pengetahuan dan kemampuan yang dimiliki, menyadari laporan ini jauh dari sempurna. Kritik dan saran yang bersifat membangun sangat diharapkan dari pembaca.

Akhir kata terima kasih dan mohon maaf apabila terdapat kesalahan baik yang disengaja maupun tidak disengaja.

Jimbaran, 1 Desember 2016

Penulis

DAFTAR ISI

	Halaman
HALAMAN PENGESAHAN.....	ii
KATA PENGANTAR	iii
DAFTAR ISI.....	iv
DAFTAR TABEL.....	vi
DAFTAR GAMBAR.....	vii
DAFTAR LAMPIRAN.....	viii
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Tujuan	2
1.3 Manfaat.....	2
1.4 Waktu Dan Tempat Pelaksanaan.....	3
BAB II GAMBARAN UMUM.....	5
2.1 Sejarah Jurusan Ilmu Komputer	5
2.2 Kegiatan Jurusan Ilmu Komputer	6
2.3 Struktur Kepengurusan Jurusan Ilmu Komputer	6
2.4 Visi, Misi dan Tujuan Jurusan Ilmu Komputer	7
BAB III KAJIAN PUSTAKA.....	9
3.1 Packet Sniffer.....	9
3.2 Paket Data Jaringan.....	10
3.3 TCP/IP.....	11
3.4 UDP.....	12
3.5 ICMP.....	14
3.6 IP Address.....	15
3.7 MAC Address.....	20
3.8 Bahasa Pemrograman Python	21
3.9 LINUX	22
BAB IV PELAKSANAAN PKL.....	23
4.1 Gambaran Umum Aplikasi Packet Sniffer	23
4.2 Perancangan Aplikasi.....	24
4.2.1 Skema Paket Data Jaringan	24
4.2.2 Use Case Diagram.....	32
4.2.3 Flowchart.....	33
4.3 Implementasi dan Hasil Aplikasi.....	34

4.4	Pengujian Aplikasi	41
BAB V KESIMPULAN DAN SARAN.....		31
5.1	Kesimpulan	43
5.2	Saran.....	43
DAFTAR PUSTAKA		44
LAMPIRAN.....		45

DAFTAR TABEL

	Halaman
Tabel 4.1 Penjelasan IP Header	27
Tabel 4.2 Penjelasan TCP Header	28
Tabel 4.3 Penjelasan UDP Header	30
Tabel 4.4 Source code inisialisasi socket awal	34
Tabel 4.5 Source Code Parsing data RAW.....	35
Tabel 4.6 Parsing paket TCP	36
Tabel 4.7 Parsing paket UDP.....	37
Tabel 4.8 Parsing Paket ICMP.....	38
Tabel 4.9 Pesan khusus untuk protokol paket lain	38
Tabel 4.10 Command line menyimpan report.....	39
Tabel 4.11 Pengujian Fungsionalis Program.....	41

DAFTAR GAMBAR

	Halaman
Gambar 2.1 Struktur organisasi Jurusan Ilmu Komputer	7
Gambar 3.1 Struktur IP Address	15
Gambar 3.2 Pembagian IP Address	16
Gambar 3.3 IP Address kelas A	17
Gambar 3.3 IP Address kelas B.....	17
Gambar 3.3 IP Address kelas C.....	18
Gambar 3.6 Ilustrasi IP Header secara umum.....	18
Gambar 3.7 Struktur MAC Address	21
Gambar 4.1 Skema IP Header	23
Gambar 4.2 Skema TCP Header.....	24
Gambar 4.3 Skema UDP Header	24
Gambar 4.4 Skema Ethernet Header	25
Gambar 4.5 Skema ICMP Header.....	25
Gambar 4.6 Use Case Diagram Packet Sniffer	32
Gambar 4.7 Flowchart Aplikasi.....	33
Gambar 4.8 Hasil Scan Paket data dalam Hex.....	35
Gambar 4.9 Hasil Scan paket data yang telah diparsing.....	39
Gambar 4.9 Hasil report berupa teks	40

DAFTAR LAMPIRAN

	Halaman
Lampiran 1. Form aktivitas harian PKL.....	A-1
Lampiran 2. Dokumentasi Kegiatan PKL.....	B-1

BAB I

PENDAHULUAN

1.1 Latar Belakang

Traffic data (lalu-lintas data) yang ada pada jaringan perlu diawasi untuk memudahkan admin dalam control atau mencegah terjadinya kegagalan dalam jaringan. Penggunaan jaringan internet bervariasi, seperti download file, streaming, browsing dan lain-lain. Disisi lain bandwidth yang dimiliki terbatas. Oleh karena itu, dibutuhkan pengaturan yang baik untuk menggunakan bandwidth yang terbatas itu.

Ide pembuatan aplikasi packet sniffer bermula saat Jurusan Ilmu Komputer FMIPA Universitas Udayana, mengembangkan beberapa fasilitas penunjang proses perkuliahan seperti: membuat 4 laboratorium baru, membangun sebuah taman internet dan pemasangan beberapa akses point jaringan wifi disetiap gedung yang ada di lingkungan Jurusan Ilmu Komputer. Namun setelah fasilitas selesai dibangun dan digunakan oleh mahasiswa, muncul masalah berupa koneksi internet yang tidak stabil. Atas dasar ini, maka dibuatlah sebuah aplikasi packet sniffer untuk mengawasi dan memberikan laporan dari lalu lintas paket data jaringan yang ada di Jurusan Ilmu Komputer demi keperluan pengelolaan jaringan.

Aplikasi dikembangkan dengan bahasa pemrograman python berbasis desktop. Pemilihan bahasa pemrograman python dikarenakan bahasa pemrograman python merupakan bahasa pemrograman tingkat tinggi mendukung banyak library, sehingga pengembangan aplikasi menjadi lebih mudah. Aplikasi dibuat untuk dapat berjalan pada system operasi berbasis LINUX, disesuaikan system operasi yang digunakan oleh komputer server yang ada di Jurusan Ilmu Komputer FMIPA Universitas Udayana yaitu system operasi berbasis LINUX (CentOs), yang juga telah secara default mendukung bahasa pemrograman python. Aplikasi menghasilkan laporan meliputi destination MAC address, source MAC address, versi IP, jenis protokol, source address, destination address, source port, destination port, sequence number, acknowledgement, panjang header, data paket

Informasi ini tentu saja diharapkan sebagai masukan yang dapat dipahami dengan mudah oleh admin. Informasi-informasi tersebut dapat digunakan dalam memblokir suatu alamat IP atau mengurangi traffic bandwidth ke suatu alamat IP, atau mungkin dapat membuat kebijakan-kebijakan dalam sisi firewall atau sisi keamanan tingkat atas berdasarkan informasi yang diberikan oleh sistem yang dibangun, sehingga admin dengan cepat dapat membuat kebijakan dalam mengelola bandwidth yang dimilikinya.

1.2 Tujuan

Adapun tujuan yang ingin dicapai dalam pembuatan aplikasi paket snifer dengan bahasa python adalah sebagai berikut:

1. Membantu administrator server dalam mengetahui lalu lintas paket data yang sedang berlangsung.
2. Membantu administrator server dalam membuat kebijakan-kebijakan lalu lintas data berdasarkan hasil aplikasi packet sniffer

1.3 Manfaat

Adapun tujuan yang ingin dicapai dalam pembuatan aplikasi paket snifer dengan bahasa python adalah sebagai berikut :

1.3.1 Manfaat Bagi Penulis

Beberapa manfaat yang diperoleh penulis dalam pembuatan aplikasi sniffer adalah :

1. Menyesuaikan diri dalam menghadapi lingkungan kerja setelah menyelesaikan studi
2. Melihat secara langsung penggunaan / penerapan teknologi dan komunikasi di tempat praktek kerja

1.3.2 Manfaat Bagi Instansi PKL

Manfaat yang didapatkan bagi instansi dari adanya program packet sniffer adalah :

1. Membantu memberikan data mengenai lalu-lintas paket data untuk digunakan dalam kebijakan-kebijakan pada firewall dan lalu lintas data pada instansi.

1.4 Waktu Dan Tempat Pelaksanaan

Pelaksanaan praktek kerja lapangan bertempat di Jurusan Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Udayana, di Jalan Kampus Bukit Jimbaran. Dimulai pada tanggal 5 September 2016 sampai dengan 25 November 2016. Pelaksanaan jam praktek kerja lapangan disesuaikan dengan jam kuliah di Jurusan Ilmu Komputer, FMIPA, Universitas Udayana yaitu pukul 08.30 wita – 16.00 wita.

BAB II

GAMBARAN UMUM

2.1 Sejarah Jurusan Ilmu Komputer

Ilmu Komputer merupakan ilmu terapan dari ilmu – ilmu dasar yang mengalami perkembangan sangat pesat seiring dengan pesatnya perkembangan Ilmu Pengetahuan dan Teknologi (IPTEK). Penguasaan bidang ilmu komputer belakangan ini sangatlah dirasa perlu dalam meningkatkan sumber daya manusia sebagai tuntutan dari perkembangan teknologi. Khususnya dalam mendukung peningkatan kualitas Tri Dharma Perguruan Tinggi di dalam institusi dan untuk menunjang proses – proses pembangunan masyarakat (daerah dan nasional), bidang ilmu komputer sangat dirasa perlu dikembangkan di Universitas Udayana (Unud).

Gejala meningkatnya kebutuhan terhadap tenaga – tenaga terdidik, terampil, dan profesional di bidang ilmu komputer dan terapannya telah diantisipasi oleh pimpinan Unud sejak tahun 2005. Berawal dari persetujuan Senat Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Udayana (FMIPA Unud) tanggal 13 Agustus 2005 tentang Pembentukan Program Studi Ilmu komputer di Fakultas MIPA Unud yang kemudian dilanjutkan ketingkat Universitas melalui persetujuan Rapat Pimpinan Universitas Udayana tanggal 15 September 2005 yang menyetujui pendirian Jurusan Ilmu Komputer di Fakultas MIPA Unud.

Seiring dengan perjalanan waktu, akhirnya pada tanggal 12 April 2006 dikeluarkanlah Ijin Penyelenggaraan PS Ilmu komputer dari DIRJEN DIKTI dengan Surat Keputusan DIKTI No.1193/D/T/ 2006 yang berlaku selama 2 tahun terhitung dari tahun pertama akademik, maka Jurusan/PS Ilmu komputer FMIPA Unud secara resmi menyelenggarakan perkuliahan untuk mahasiswa angkatan I (tahun akademik 2006/2007) pada tanggal 3 September 2006 dengan jumlah mahasiswa terdaftar 100 (seratus) orang dari kapasitas sebenarnya

yang hanya 50 (lima puluh) orang. Animo masyarakat untuk mendalami bidang ilmu komputer memang sangat tinggi, hal ini dapat dilihat dari banyaknya pendaftar pada angkatan pertama sebanyak 291 orang.

Begitu juga pada tahun ajaran 2007/2008 dimana Jurusan Ilmu Komputer sebagai jurusan baru sudah dapat mensejajarkan diri dengan jurusan - jurusan favorit lainnya dalam penerimaan mahasiswa dengan masuknya Jurusan Ilmu Komputer sebagai salah satu jurusan yang memperoleh mahasiswa sesuai dengan kuota penerimaan sehingga tidak ada bangku kosong.

2.2 Kegiatan Jurusan Ilmu Komputer

Jurusan Ilmu Komputer merupakan salah satu jurusan yang berada di bawah naungan Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Udayana, yang memiliki beberapa aktivitas – aktivitas akademik maupun non akademik, yaitu antara lain : belajar mengajar, seminar publikasi ilmiah, pengabdian masyarakat, kegiatan organisasi mahasiswa (Himakom dan SIC), dan lain sebagainya.

2.3 Struktur Kepengurusan Jurusan Ilmu Komputer

Jurusan Ilmu Komputer, FMIPA Unud memiliki struktur kepengurusan sebagai berikut :

2.4 Visi, Misi dan Tujuan Jurusan Ilmu Komputer

2.4.1 Visi Jurusan Ilmu Komputer

2.4.2 Misi Jurusan Ilmu Komputer

Adapun Misi Jurusan Ilmu Komputer Fakultas MIPA Universitas Udayana dapat dijabarkan sebagai berikut:

1. Menyelenggarakan proses pembelajaran yang mampu menghasilkan lulusan yang berkualitas, mandiri, professional, dan berbudaya dalam bidang Ilmu Komputer/Informatika.
2. Menyelenggarakan dan mengorganisasikan pendidikan di bidang ilmu komputer/Informatika yang adaptif dan responsif pada perkembangan riset yang menunjang pembangunan nasional dan internasional.
3. Mengembangkan riset dan penyebarluasan hasil-hasil riset di bidang Ilmu Komputer/Informatika melalui program pengabdian kepada masyarakat.

2.4.3 Tujuan Jurusan Ilmu Komputer

Jurusan Ilmu Komputer FMIPA Unud memiliki beberapa tujuan yang ingin dicapai, yaitu sebagai berikut :

1. Menghasilkan lulusan yang berkualitas, mandiri, dan berbudaya serta memiliki wawasan luas dengan penguasaan bidang Ilmu Komputer/Teknik Informatika yang kompeten.
2. Menghasilkan lulusan yang memiliki kemampuan problem solving, kreatif, dan inovatif sehingga mampu berpartisipasi dalam pengembangan riset di bidang Ilmu Komputer/Tenik Informatika.
3. Meningkatkan kualitas dan kuantitas penelitian di bidang Ilmu Komputer/Tenik Informatika dalam mendukung pembangunan nasional.
4. Meningkatkan kuantitas dan kualitas pengabdian masyarakat untuk penyebarluasan perkembangan riset di bidang Ilmu Komputer/Tenik Informatika.

BAB III

KAJIAN PUSTAKA

3.1 Packet Sniffer

Packet Sniffer yang juga dikenal sebagai Network Analyzers atau Ethernet Sniffer ialah sebuah aplikasi yang dapat melihat lalu lintas paket data pada jaringan komputer. Dikarenakan data mengalir secara bolak-balik pada jaringan, aplikasi ini menangkap tiap-tiap paket dan bisa menguraikan isi dari RFC (Request for Comments) atau spesifikasi yang lain.

Sebuah sniffer paket memungkinkan untuk menyadap data pada lalu lintas perjalanan antar jaringan komputer. Packet sniffer akan menangkap data yang ditujukan ke mesin lain dan menyimpannya untuk analisis nanti. Sebuah packet sniffer bisa dipasang pada setiap titik di sepanjang jalan. Bisa juga secara sembunyi-sembunyi diinstal pada sebuah server yang bertindak sebagai gateway atau mengumpulkan informasi pribadi penting.

Teknisi jaringan dapat menggunakan informasi ini untuk menentukan di mana kesalahan terletak, seperti menentukan perangkat gagal menanggapi permintaan jaringan. Namun ditangan orang yang salah dapat menggunakan aplikasi sniffer untuk menyadap data yang tidak terenkripsi di paket untuk melihat informasi apa yang sedang dipertukarkan antara dua pihak. Mereka juga dapat menangkap informasi seperti password dan token otentikasi. Namun, di tangan yang salah, sniffer paket dapat menangkap informasi pribadi sensitif yang dapat menyebabkan pelanggaran privasi, pencurian identitas, dan eventualities serius lainnya.

3.1.1 Fungsi Packet Sniffer

Secara umum aplikasi packet sniffer dapat dimanfaatkan untuk hal-hal berikut:

- a. Mengatasi permasalahan pada jaringan komputer.
- b. Mendeteksi adanya penyelundup dalam jaringan (Network Intusion).
- c. Memonitor penggunaan jaringan dan menyaring isi isi tertentu.

- d. Memata-matai pengguna jaringan lain dan mengumpulkan informasi pribadi yang dimilikinya.
- e. Dapat digunakan untuk Reverse Engineer pada jaringan.

3.1.2 Cara Kerja Packet Sniffer

Packet sniffer bekerja dengan mencegat dan mencatat lalu lintas data pada jaringan bahwa mereka dapat melihat melalui antarmuka jaringan kabel atau nirkabel dimana aplikasi packet sniffer memiliki akses ke komputer host.

Pada jaringan kabel, apa yang dapat ditangkap tergantung pada struktur jaringan. Sebuah paket sniffer mungkin dapat melihat lalu lintas di seluruh jaringan atau hanya segmen tertentu, tergantung pada bagaimana switch jaringan dikonfigurasi, ditempatkan, dll. Pada jaringan wireless, sniffer paket dapat biasanya hanya menangkap satu saluran pada waktu kecuali komputer host memiliki beberapa antarmuka nirkabel yang memungkinkan untuk multichannel capture.

Setelah data paket raw ditangkap, paket sniffing software harus menganalisis, mengkonversi, dan menyajikannya dalam bentuk yang dapat dibaca oleh manusia sehingga orang yang menggunakan aplikasi packet sniffer dapat memahaminya.

3.2 Paket Data Jaringan

Paket data jaringan atau network packet adalah satuan informasi dasar yang dapat ditransmisikan di atas jaringan atau melalui saluran komunikasi digital. Sebuah paket berisi packet header yang berisi informasi mengenai protokol tersebut (informasi mengenai jenis, sumber, tujuan, atau informasi lainnya), data yang hendak ditransmisikan yang disebut dengan data payload, dan packet trailer yang bersifat opsional. Sebuah paket memiliki struktur logis yang dibentuk oleh protokol yang digunakannya. Ukuran setiap paket juga dapat bervariasi, tergantung struktur yang dibentuk oleh arsitektur jaringan yang digunakan. Paket jaringan juga dapat disebut datagram, frame, atau cell.

3.3 TCP/IP

3.3.1 Pengertian TCP/IP

TCP/IP adalah sekumpulan protokol yang terdapat didalam jaringan komputer (network) yang digunakan untuk berkomunikasi atau bertukar data antar komputer. TCP/IP merupakan protokol standar pada jaringan internet yang menghubungkan banyak komputer yang berbeda jenis mesin maupun sistem operasi agar dapat berinteraksi satu sama lain.

Protokol merupakan himpunan aturan yang memungkinkan komputer untuk berhubungan antara satu dengan yang lain, biasanya berupa bentuk waktu, barisan, pemeriksaan error saat transmisi data. Komputer yang terhubung ke internet berkomunikasi dengan protokol ini. Karena menggunakan bahasa yang sama, yaitu protokol TCP/IP, perbedaan jenis komputer dan sistem operasi tidak menjadi masalah. Komputer PC dengan sistem operasi Windows dapat berkomunikasi dengan komputer Sun-SPARC yang menjalankan Solaris. Jadi, jika sebuah komputer menggunakan protokol TCP/IP dan terhubung ke internet, maka komputer tersebut dapat berhubungan langsung dengan komputer lain dibelahan dunia manapun yang juga terhubung dengan internet.

TCP/IP berfungsi melakukan komunikasi data pada jaringan komputer. TCP/IP terdiri atas sekumpulan protocol yang masing-masing bertanggung jawab atas bagian-bagian tertentu dari komunikasi data. Jadi tugas masing-masing protokol menjadi jelas dan sederhana. Protokol yang satu tidak perlu mengetahui cara kerja protokol yang lain, sepanjang ia masih bisa saling mengirim dan menerima data.

3.3.2 Sejarah TCP/IP

Konsep TCP/IP berawal dari kebutuhan DoD (Departement of Defense) USA akan suatu komunikasi di antara berbagai variasi komputer yang telah ada. Komputer-komputer DoD ini seringkali harus menghubungkan antara satu organisasi peneliti dengan organisasi peneliti lainnya. Komputer tersebut harus tetap berhubungan karena terkait dengan pertahanan negara dan sumber informasi harus tetap berjalan meskipun terjadi bencana alam besar,

seperti ledakan nuklir. Oleh karenanya pada tahun 1969 dimulailah penelitian terhadap serangkaian protokol TCP/IP. Adapun tujuan penelitian tersebut adalah sebagai berikut.

- a. Terciptanya protokol-protokol umum, (DoD memerlukan suatu protokol yang dapat dipergunakan untuk semua jenis jaringan).
- b. Meningkatkan efisiensi komunikasi data.
- c. Dapat dipadukan dengan teknologi WAN (Wide Area Network) yang telah ada.
- d. Mudah dikonfigurasi.

Protokol-protokol TCP/IP dikembangkan lebih lanjut pada awal 1980 dan menjadi protokol standar untuk ARPAnet pada tahun 1983. Protokol-protokol ini mengalami peningkatan popularitas di komunitas pemakai ketika TCP/IP dapat diimplementasikan dengan sangat baik pada versi 4.2 BSD (Berkeley Standard Distribution) UNIX. Versi ini digunakan secara luas pada institusi penelitian dan pendidikan serta digunakan sebagai dasar dari beberapa penerapan UNIX komersial, termasuk SunOS dari Sun dan Ultrix dari Digital.

3.4 UDP

3.4.1 Pengertian UDP

Dalam Buku I Putu Agus Eka Pratama tahun 2014 dijelaskan bahwa UDP (User Datagram Protocol) merupakan salah satu protokol utama di dalam jaringan komputer, khususnya pada Transport Layer, yang bersifat Connectionless dan Unreliable. Connectionless dapat diartikan bahwa UDP tidak memerlukan adanya persiapan (setup) koneksi terlebih dahulu untuk memulai proses dan layanan di dalamnya. Unreliable atau tidak andal, memiliki bahwa UDP tidak melakukan pengecekan untuk keandalan didalam jaringan layaknya seperti protokol TCP. Serta memiliki Header UDP yang didalamnya memuat SPI (Source Process Identification) dan DPI (Destination Process Identification).

3.4.2 Karakteristik UDP

Karakteristik dari UDP antara lain, yaitu sebagai berikut :

- a. Connectionless (tanpa koneksi): Pesan-pesan UDP akan dikirimkan tanpa harus dilakukan proses negosiasi koneksi antara dua host yang hendak berukar informasi.
- b. Unreliable (tidak andal): Pesan-pesan UDP akan dikirimkan sebagai datagram tanpa adanya nomor urut atau pesan acknowledgment. Protokol lapisan aplikasi yang berjalan di atas UDP harus melakukan pemulihan terhadap pesan-pesan yang hilang selama transmisi. Umumnya, protokol lapisan aplikasi yang berjalan di atas UDP mengimplementasikan layanan keandalan mereka masing-masing, atau mengirim pesan secara periodik atau dengan menggunakan waktu yang telah didefinisikan.
- c. UDP menyediakan mekanisme untuk mengirim pesan-pesan ke sebuah protokol lapisan aplikasi atau proses tertentu di dalam sebuah host dalam jaringan yang menggunakan TCP/IP. HeaderUDP berisi field Source Process Identification dan Destination Process Identification.
- d. UDP menyediakan perhitungan checksum berukuran 16 bit terhadap keseluruhan pesan UDP.

3.4.3 Kegunaan UDP

UDP sering digunakan dalam beberapa tugas berikut:

- a. Protokol yang “ringan” (lightweight): Untuk menghemat sumber daya memori dan prosesor, beberapa protokol lapisan aplikasi membutuhkan penggunaan protokol yang ringan yang dapat melakukan fungsi-fungsi spesifik dengan saling bertukar pesan. Contoh dari protokol yang ringan adalah fungsi query nama dalam protokol lapisan aplikasi Domain Name System (DNS).
- b. Protokol lapisan aplikasi yang mengimplementasikan layanan keandalan: Jika protokol lapisan aplikasi menyediakan layanan transfer data yang andal, maka kebutuhan terhadap keandalan yang ditawarkan oleh TCP pun menjadi tidak ada. Contoh dari protokol seperti ini

adalah Trivial File Transfer Protocol (TFTP) dan Network File System (NFS)

- c. Protokol yang tidak membutuhkan keandalan. Contoh protokol ini adalah protokol Routing Information Protocol (RIP).
- d. Transmisi broadcast: Karena UDP merupakan protokol yang tidak perlu membuat koneksi terlebih dahulu dengan sebuah host tertentu, maka transmisi broadcast pun dimungkinkan. Sebuah protokol lapisan aplikasi dapat mengirimkan paket data ke beberapa tujuan dengan menggunakan alamat multicast atau broadcast. Hal ini kontras dengan protokol TCP yang hanya dapat mengirimkan transmisi one-to-one. Contoh: query nama dalam protokol NetBIOS Name Service.

3.5 ICMP

ICMP (Internet Control Message Protocol) adalah protokol yang bertugas mengirimkan pesan-pesan kesalahan dan kondisi lain yang memerlukan perhatian khusus. Pesan/ paket ICMP dikirim jika terjadi masalah pada layer IP dan layer atasnya (TCP/UDP). Pada kondisi normal, protokol IP berjalan dengan baik. Namun ada beberapa kondisi dimana koneksi IP terganggu, misalnya karena Router crash, putusnya kabel, atau matinya host tujuan. Pada saat ini ICMP membantu menstabilkan kondisi jaringan, dengan memberikan pesan-pesan tertentu sebagai respons atas kondisi tertentu yang terjadi pada jaringan tersebut.

3.6 IP Address

IP Address adalah alamat yang diberikan untuk jaringan komputer yang menggunakan protokol TCP IP. IP address terdiri dari 32 bit angka biner yang dapat dituliskan menjadi 4 kelompok bilangan decimal yang dipisahkan dengan tanda titik.

3.6.1 Struktur IP Address

Pada IP versi 4 Alamat IP terdiri dari bilangan biner sepanjang 32 bit yang dibagi atas 4 segmen. Tiap segmen terdiri atas 8 bit yang berarti memiliki nilai desimal dari 0 - 255. Luas area dari alamat IP (range address) yang bisa digunakan adalah dari 00000000.00000000.00000000.00000000 sampai dengan 11111111.11111111.11111111.11111111. Jadi, ada sebanyak 232 kombinasi address yang bisa dipakai diseluruh dunia (walaupun pada kenyataannya ada sejumlah IP Address yang digunakan untuk keperluan khusus). Jadi, jaringan TCP/IP dengan 32 bit address ini mampu menampung sebanyak 232 atau lebih dari 4 milyar host. Untuk memudahkan pembacaan dan penulisan, IP Address biasanya direpresentasikan dalam bilangan desimal. Jadi, range address di atas dapat diubah menjadi address 0.0.0.0 sampai address 255.255.255.255. Nilai desimal dari IP Address inilah yang dikenal dalam pemakaian sehari-hari. Format IP Address

Binary	Decimal
00000000.00000000.00000000.00000000	= 0.0.0.0
s/d	
11111111.11111111.11111111.11111111	= 255.255.255.255

Gambar 3.1 Struktur IP Address

Alamat IP yang dimiliki oleh sebuah host dapat dibagi dengan menggunakan subnet mask jaringan ke dalam dua buah bagian, yakni :

- Network Identifier/NetID* atau *Network Address* (alamat jaringan) yang digunakan khusus untuk mengidentifikasi alamat jaringan di mana host berada. Semua sistem di dalam sebuah jaringan fisik yang sama harus memiliki alamat *network identifier* yang sama. *Network identifier* juga harus bersifat unik dalam sebuah *internetwork*. Alamat *network identifier* tidak boleh bernilai 0 atau 255.

- b. *Host Identifier/HostID* atau *Host address* (alamat host) yang digunakan khusus untuk mengidentifikasi alamat host di dalam jaringan. Nilai *host identifier* tidak boleh bernilai 0 atau 255 dan harus bersifat unik di dalam *network identifier* di mana ia berada.



Gambar 3.2 Pembagian IP Address

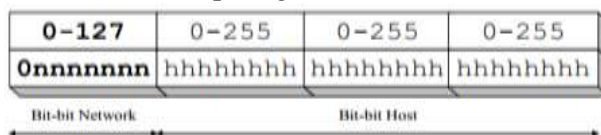
3.6.2 Pembagian Kelas IP Address

Jumlah IP address yang tersedia secara teoritis adalah $255 \times 255 \times 255 \times 255$ atau sekitar 4 milyar lebih yang harus dibagikan ke seluruh pengguna jaringan. Pembagian kelas-kelas ini ditujukan untuk mempermudah alokasi IP Address, baik untuk host/jaringan tertentu atau untuk keperluan tertentu.

Address dapat dipisahkan menjadi 2 bagian, yakni bagian network (net ID) dan bagian host (host ID). Net ID berperan dalam identifikasi suatu network dari network yang lain, sedangkan host ID berperan untuk identifikasi host dalam suatu network. Jadi, seluruh host yang tersambung dalam jaringan yang sama memiliki net ID yang sama. Sebagian dari bit-bit bagian awal dari IP Address merupakan network bit/network number, sedangkan sisanya untuk host. Garis pemisah antara bagian network dan host tidak tetap, bergantung kepada kelas network. IP address dibagi ke dalam lima kelas, yaitu kelas A, kelas B, kelas C, kelas D dan kelas E. Perbedaan tiap kelas adalah pada ukuran dan jumlahnya. Contohnya IP kelas A dipakai oleh sedikit jaringan namun jumlah host yang dapat ditampung oleh tiap jaringan sangat besar. Kelas D dan E tidak digunakan secara umum, kelas D digunakan bagi jaringan multicast dan kelas E untuk keperluan eksperimental. Perangkat lunak Internet Protocol menentukan pembagian jenis kelas ini dengan menguji beberapa bit pertama dari IP Address. Penentuan kelas ini dilakukan dengan cara berikut :

a. Kelas A

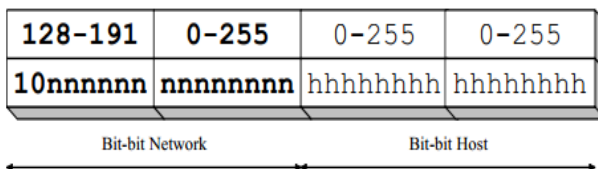
Bit pertama IP address kelas A adalah 0, dengan panjang net ID 8 bit dan panjang host ID 24 bit. Jadi byte pertama IP address kelas A mempunyai range. Jadi kelas A terdapat 127 network dengan tiap network dapat menampung sekitar 16 juta host ($255 \times 255 \times 255$). IP address kelas A diberikan untuk jaringan dengan jumlah host yang sangat besar, IP kelas ini dituliskan pada gambar berikut ini.



Gambar 3.3 IP Address kelas A

b. Kelas B

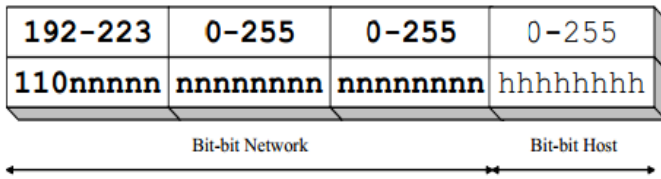
Dua bit IP address kelas B selalu diset 10 sehingga byte pertamanya selalu bernilai antara 128-191. Network ID adalah 16 bit pertama dan 16 bit sisanya adalah host ID sehingga kalau ada komputer mempunyai IP address 192.168.26.161, network ID = 192.168 dan host ID = 26.161. Pada IP address kelas B ini mempunyai range IP dari 128.0.xxx.xxx sampai 191.155.xxx.xxx, yakni berjumlah 65.255 network dengan jumlah host tiap network 255×255 host atau sekitar 65 ribu host.



Gambar 3.4 IP Address kelas B

c. Kelas C

IP address kelas C mulanya digunakan untuk jaringan berukuran kecil seperti LAN. Tiga bit pertama IP address kelas C selalu diset 111. Network ID terdiri dari 24 bit dan host ID 8 bit sisanya sehingga dapat terbentuk sekitar 2 juta network dengan masing-masing network memiliki 256 host.



Gambar 3.5 IP Address kelas C

d. Kelas D

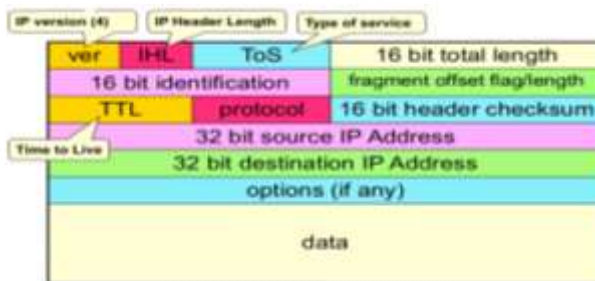
IP address kelas D digunakan untuk keperluan multicasting. 4 bit pertama IP address kelas D selalu diset pertamanya berkisar antara 224-247, sedangkan bit-bit berikutnya diatur sesuai keperluan multicast group 1110 sehingga byte yang menggunakan IP address ini. Dalam multicasting tidak dikenal istilah network ID dan host ID.

e. Kelas E

IP address kelas E tidak diperuntukkan untuk keperluan eksperimental. 4 bit pertama IP address kelas ini diset 1111 sehingga byte pertamanya berkisar antara 248-255.

3.6.3 IP Header

IP header adalah informasi dimana IP protocol menambahkan di depan transport klien layer X untuk membuat IP paket. Header ini panjangnya 32 byte dan mencakup source dan destination IP address. Header-header IP diilustrasikan dalam datagram dibawah ini.



Gambar 3.6 Ilustrasi IP Header secara umum

Fungsi dari masing-masing komponen diatas adalah sebagai berikut:

a. Version (4 bit)

Header ini mendefinisikan versi Internet Protocol yang digunakan, versi yang secara luas banyak digunakan adalah versi 4.

b. IHL (4 bit)

Header ini mendefinisikan panjang header IP dalam 32 bit word. Nilai minimum yang valid adalah 5 dan maksimumnya 6.

c. Type of Service (8 bit)

Merupakan header yang menentukan bagaimanapun proses transmisi datagram secara benar.

d. Packet Length (16 bit)

Header yang mendefinisikan total panjang header dan data pada IP

e. Identification (16 bit)

Header ini untuk mendukung fasilitas fragmentasi

f. DF (1 bit)

Header ini untuk mendefinisikan agar transmisi tidak di fragmentasi

g. DM (1 bit)

Header ini untuk mendefinisikan bahwa ada paket yang difragmentasi pada paket-paket berikutnya (More Fragment)

h. Fragment Offset (13 bit)

Header ini mendefinisikan lokasi dari paket-paket yang mengalami fragmentasi dalam urutan keseluruhan paket

i. TTL (16 bit)

Time to Live merupakan header yang mendefinisikan umur paket data, TTL akan berkurang 1 jika melewati sebuah router, demikian seterusnya sampai paket sampai ke host tujuan. Dengan mekanisme ini, dapat diantisipasi dimana paket bergentayangan terus di internet sehingga banyak terdapat paket sampah di internet jika ternyata host tujuan tidak ditemukan. Jika TTL telah habis (bernilai 0) sedangkan

paket data belum sampai ke host tujuan, maka paket data akan dibuang.

j. Transport (8 Bit)

Header ini mendefinisikan protokol pada layer transport yang digunakan, header ini bisa berupa TCP atau UDP

k. Header Checksum (32 Bit)

Header ini digunakan untuk mengecek apakah terjadi perubahan/kerusakan pada header IP. Header Checksum dikalkulasi pada tiap router.

l. Sending Address (32 Bit)

Sending Address atau Source Address merupakan header yang mendefinisikan IP address dari host asal.

m. Destination Address (32 Bit)

Destination Address merupakan header yang mendefinisikan IP address dari host tujuan.

n. Option (32 Bit)

Header ini digunakan untuk mendefinisikan informasi tambahan pada layer transport seperti source routing.

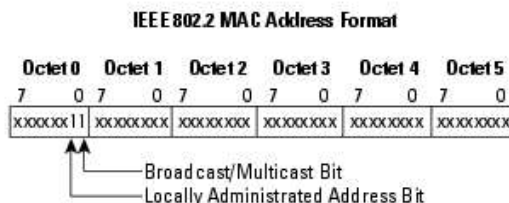
o. Padding (32 Bit)

Header ini sama dengan padding pada TCP, yaitu digunakan untuk memenuhi panjang header sehingga merupakan kelipatan 32 bit. Jika terdapat header yang kurang, maka padding ditambahkan sampai berjumlah 32 bit

3.7 MAC Address

MAC Address (Media Access Control address) adalah alamat fisik suatu interface jaringan (seperti ethernet card pada komputer, interface/port pada router, dan node jaringan lain) yang bersifat unik dan berfungsi sebagai identitas perangkat tersebut. Secara umum MAC Address dibuat dan diberikan oleh pabrik pembuat NIC (Network Interface Card) dan disimpan secara permanen pada ROM (Read Only Memory) perangkat tersebut. MAC address juga biasa disebut Ethernet Hardware Address (EHA), Hardware Address, atau Physical Address.

MAC Address memiliki panjang 48-bit (6 byte). Format standard MAC Address secara umum terdiri dari 6 kelompok digit yang masing-masing kelompok berjumlah 2 digit heksadesimal. masing-masing kelompok digit dipisahkan tanda (-) atau (:),



Gambar 3.7 Struktur MAC Address

Supaya komputer dan perangkat jaringan lain bisa berkomunikasi satu dengan yang lain, frame-frame / data yang dikirim melalui jaringan harus memiliki MAC Address. Tetapi agar komunikasi jaringan lebih mudah dan sederhana, digunakanlah IP Address. Karena komunikasi jaringan menggunakan MAC Address maka alamat IP tersebut harus diterjemahkan ke MAC Address. Maka dari itu diciptakanlah ARP (Address Resolution Protocol) yang bertugas untuk menerjemahkan IP Address menjadi MAC Address sehingga komputer pun bisa saling berkomunikasi.

3.8 Bahasa Pemrograman Python

Python merupakan bahasa pemrograman tingkat tinggi yang dapat berjalan di berbagai system operasi. Python adalah bahasa pemrograman interpretatif multiguna. Tidak seperti bahasa lain yang susah untuk dibaca dan dipahami, python lebih menekankan pada keterbacaan kode agar lebih mudah untuk memahami sintaks. Hal ini membuat Python sangat mudah dipelajari baik untuk pemula maupun untuk yang sudah menguasai bahasa pemrograman lain.

Pada laporan ini, Python dijalankan pada system operasi berbasis LINUX. Python pada LINUX tersedia secara default tanpa harus instalasi paket tambahan lagi. Untuk mengeceknya bisa dilakukan dengan mengetikkan “python” pada terminal.

Bahasa ini muncul pertama kali pada tahun 1991, dirancang oleh seorang bernama Guido van Rossum. Sampai saat ini Python masih dikembangkan oleh Python Software Foundation. Bahasa Python mendukung hampir semua sistem operasi, bahkan untuk sistem operasi Linux, hampir semua distronya sudah menyertakan Python di dalamnya.

Dengan kode yang simpel dan mudah diimplementasikan, seorang programmer dapat lebih mengutamakan pengembangan aplikasi yang dibuat, bukan malah sibuk mencari syntax error.

3.9 LINUX

Linux adalah nama yang diberikan kepada sistem operasi komputer bertipe Unix. Linux merupakan salah satu contoh hasil pengembangan perangkat lunak bebas dan sumber terbuka utama. Seperti perangkat lunak bebas dan sumber terbuka lainnya pada umumnya, kode sumber Linux dapat dimodifikasi, digunakan dan didistribusikan kembali secara bebas oleh siapa saja.

Nama "Linux" berasal dari nama pembuatnya, yang diperkenalkan tahun 1991 oleh Linus Torvalds. Linux telah lama dikenal untuk penggunaannya di server, dan didukung oleh perusahaan-perusahaan komputer ternama seperti Intel, Dell, Hewlett-Packard, IBM, Novell, Oracle Corporation, Red Hat, dan Sun Microsystems. Linux digunakan sebagai sistem operasi di berbagai macam jenis perangkat keras komputer, termasuk komputer desktop hingga supercomputer.

Saat ini terdapat banyak distribusi Linux (lebih dikenali sebagai distro) yang dibuat oleh individu, grup, atau lembaga lain. Masing-masing disertakan dengan program sistem dan program aplikasi tambahan, di samping menyertakan suatu program yang memasang keseluruhan sistem di komputer (installer program). Contoh-contoh distribusi linux antara lain:

- a. Ubuntu
- b. OpenSUSE
- c. Fedora
- d. BackTrack
- e. CentOS

BAB IV PELAKSANAAN PKL

4.1 Gambaran Umum Aplikasi Packet Sniffer

Dalam perancangan aplikasi packet sniffer agar dapat menjadi solusi dalam mengawasi traffic setiap paket yang melintasi jaringan guna membantu admin agar mudah mengawasi jaringan yang dikelolanya.

Aplikasi packet sniffer ini bekerja dengan melihat setiap paket yang dikirim dalam jaringan, termasuk paket tidak dimaksudkan untuk dirinya sendiri dengan membuka sebuah port socket yang digunakan untuk mencatat semua lalu lintas data pada jaringan yang sedang berjalan. Dalam aplikasi packet sniffer untuk mengawasi lalu lintas data pada jaringan, terdapat beberapa hal yang perlu diperhatikan seperti IP sumber dan tujuan paket jaringan, tipe paket jaringan seperti TCP/IP, UDP, hingga ICMP, juga port dan MAC address dari paket jaringan.

Pada tahap awal, informasi data yang diawasi dan dicatat masih berupa data paket RAW yang belum bisa dibaca. Setelah data paket RAW ditangkap, aplikasi packet sniffer harus mengkonversi (parsing) data tersebut dalam bentuk yang dapat dibaca oleh manusia sehingga admin yang menggunakan aplikasi packet sniffer ini dapat memahaminya. Hasil konversi data tersebut kemudian dianalisis berdasarkan tipe packet data dan disimpan dalam report berjenis teks.

Secara umum cara kerja dari packet sniffer yang dibuat dapat dibagi menjadi 3 yaitu collecting, conversion, analysis. Untuk penjelasannya adalah sebagai berikut:

1. Collecting

Cara kerja yang pertama dari packet sniffing adalah merubah interface yang digunakan menjadi "promiscuous mode", dan mulai mengumpulkan atau mengelompokkan semua paket data yang lewat melalui jaringan dalam bentuk raw binary.

2. Conversion

Cara kerja kedua adalah mengkonversi atau merubah data yang berbentuk binary kedalam data yang mudah dibaca atau mudah dipahami.

3. Analysis

Cara kerja ketiga adalah dimana bentuk data tersebut disimpan dalam report teks yang berisi informasi data, baik dari sumber dari transmisi berupa TCP, UDP dan lain-lain.

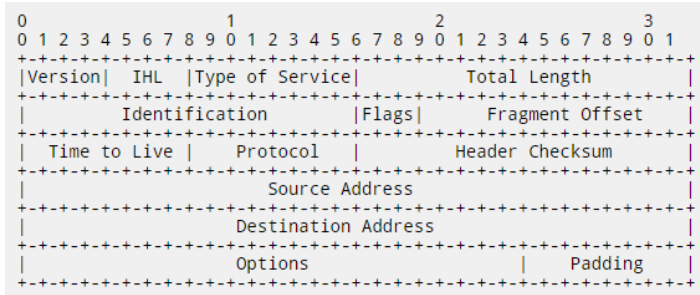
4.2 Perancangan Aplikasi

Aplikasi dikembangkan dengan bahasa pemrograman python. Bahasa pemrograman python dipilih karena python merupakan bahasa tingkat tinggi yang secara default telah terinstalasi pada system operasi CentOS yang digunakan oleh komputer server.

4.2.1 Skema Paket Data Jaringan

Sebelum membuat program paket sniffer kita perlu mengetahui bagian-bagian dari paket data jaringan yang ingin kita ambil. Berikut adalah skema dari beberapa tipe paket data jaringan berdasarkan protokol.

a. IP Header (berdasarkan RFC 791)



Gambar 4.1 Skema IP Header

Tabel 4.1 Penjelasan IP Header

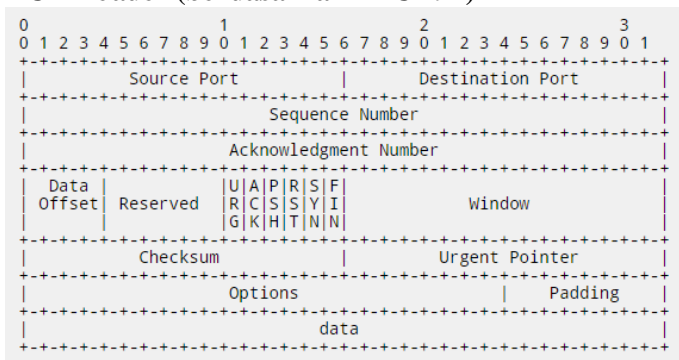
Field	Panjang	Keterangan
Version	4 bit	Digunakan untuk mengindikasikan versi dari header IP yang digunakan. Karena memiliki panjang 4 bit, maka terdapat $2^4=16$ buah jenis nilai yang berbeda-beda, yang berkisar antara 0 hingga 15. Meskipun begitu hanya ada dua nilai yang bisa digunakan, yakni 4 dan 6,

		mengingat versi IP standar yang digunakan saat ini dalam jaringan dan Internet adalah versi 4 dan 6.
Header length	4 bit	Digunakan untuk mengindikasikan ukuran header IP. Karena memiliki panjang 4 bit, maka terdapat $2^4=16$ buah jenis nilai yang berbeda-beda. Field header length ini mengindikasikan bilangan double-word 32-bit (blok 4-byte) di dalam header IP.
Type of Service (TOS)	8 bit	Field ini digunakan untuk menentukan kualitas transmisi dari sebuah datagram IP.
Total Length	16 bit	Merupakan panjang total dari datagram IP, yang mencakup header IP dan muatannya. Dengan menggunakan angka 16 bit, nilai maksimum yang dapat ditampung adalah 65535 byte.
Identifier	16 bit	Digunakan untuk mengidentifikasi sebuah paket IP tertentu yang dikirimkan antara node sumber dan node tujuan. Host pengirim akan mengeset nilai dari field ini, dan field ini akan bertambah nilainya untuk datagram IP selanjutnya. Field ini digunakan untuk mengenali fragmen-fragmen sebuah datagram IP.
Flag	3 bit	Berisi dua buah flag yang berisi apakah datagram IP mengalami fragmentasi atau tidak.
Fragment Offset	13 bit	Digunakan untuk mengidentifikasi offset dimana fragmen yang bersangkutan dimulai, dihitung dari permulaan muatan IP yang belum dipecah.
Time-to-Live (TTL)	8 bit	Digunakan untuk mengidentifikasi berapa banyak saluran jaringan di mana

		<p>sebuah datagram IP dapat berjalan-jalan sebelum sebuah router mengabaikan datagram tersebut. Field ini pada awalnya ditujukan sebagai penghitung waktu, untuk mengidentifikasi berapa lama (dalam detik) sebuah datagram IP boleh terdapat di dalam jaringan. Adalah router IP yang memantau nilai ini, yang akan berkurang setiap kali hinggap dalam router.</p>
Protocol	8 bit	<p>Digunakan untuk mengidentifikasi jenis protokol lapisan yang lebih tinggi yang dikandung oleh muatan IP. Field ini merupakan tanda eksplisit untuk protokol klien. Terdapat beberapa nilai dari field ini, seperti halnya nilai 1 (0x01) untuk ICMP, 6 (0x06) untuk TCP, dan 17 (0x11) untuk UDP. Field ini bertindak sebagai penanda multipleks (multiplex identifier), sehingga muatan IP pun dapat diteruskan ke protokol lapisan yang lebih tinggi saat diterima oleh node yang dituju.</p>
Header Checksum	16 bit	<p>Field ini berguna hanya untuk melakukan pengecekan integritas terhadap header IP, sementara muatan IP sendiri tidak dimasukkan ke dalamnya, sehingga muatan IP harus memiliki checksum mereka sendiri untuk melakukan pengecekan integritas terhadap muatan IP. Host pengirim akan melakukan pengecekan checksum terhadap datagram IP yang dikirimkan. Setiap router yang berada di dalam jalur transmisi antara sumber dan tujuan akan</p>

		melakukan verifikasi terhadap field ini sebelum memproses paket. Jika verifikasi dianggap gagal, router pun akan mengabaikan datagram IP tersebut. Karena setiap router yang berada di dalam jalur transmisi antara sumber dan tujuan akan mengurangi nilai TTL, maka header checksum pun akan berubah setiap kali datagram tersebut hinggap di setiap router yang dilewati. Pada saat menghitung checksum terhadap semua field di dalam header IP, nilai header checksum akan diset ke nilai 0.
Source IP Address	32 bit	Mengandung alamat IP dari sumber host yang mengirimkan datagram IP tersebut, atau alamat IP dari Network Address Translator (NAT).
Destination IP Address	32 bit	Mengandung alamat IP tujuan kemana datagram IP tersebut akan disampaikan, atau yang dapat berupa alamat dari host atau NAT.

b. TCP Header (berdasarkan RFC 791)



Gambar 4.2 Skema TCP Header

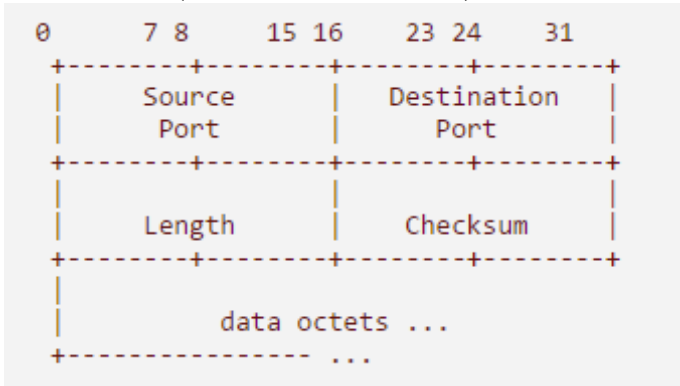
Tabel 4.2 Penjelasan TCP Header

Nama field	Ukuran	Keterangan
Source Port	2 byte (16 bit)	Menandakan sumber protokol lapisan aplikasi yang mengirimkan segmen TCP yang bersangkutan. Gabungan antara field Source IP Address dalam header IP dan field Source Port dalam field header TCP disebut juga sebagai source socket, yang berarti sebuah alamat global dari mana segmen dikirimkan. Lihat juga Port TCP.
Destination Port	2 byte (16 bit)	Mengindikasikan tujuan protokol lapisan aplikasi yang menerima segmen TCP yang bersangkutan. Gabungan antara field Destination IP Address dalam header IP dan field Destination Port dalam field header TCP disebut juga sebagai socket tujuan, yang berarti sebuah alamat global ke mana segmen akan dikirimkan.
Sequence Number	4 byte (32 bit)	Mengindikasikan nomor urut dari oktet pertama dari data di dalam sebuah segmen TCP yang hendak dikirimkan. Field ini harus selalu diset, meskipun tidak ada data (payload) dalam segmen. Ketika memulai sebuah sesi koneksi TCP, segmen dengan flag SYN (Synchronization) diset ke nilai 1, field ini akan berisi nilai Initial Sequence Number (ISN). Hal ini berarti, oktet pertama dalam aliran byte (byte stream) dalam koneksi adalah ISN+1.

Acknowledgment Number	4 byte (32 bit)	Mengindikasikan nomor urut dari oktet selanjutnya dalam aliran byte yang diharapkan oleh untuk diterima oleh pengirim dari si penerima pada pengiriman selanjutnya. Acknowledgment number sangat dipentingkan bagi segmen-segmen TCP dengan flag ACK diset ke nilai 1.
Data Offset	4 bit	Mengindikasikan di mana data dalam segmen TCP dimulai. Field ini juga dapat berarti ukuran dari header TCP.
Reserved	6 bit	Direservasikan untuk digunakan pada masa depan. Pengirim segmen TCP akan mengeset bit-bit ini ke dalam nilai 0.
Flags	6 bit	Mengindikasikan flag-flag TCP yang memang ada enam jumlahnya, yang terdiri atas: URG (Urgent), PSH (Push), ACK (Acknowledgment), RST (Reset), SYN (Synchronize), dan FIN (Finish).
Window	2 byte (16 bit)	Mengindikasikan jumlah byte yang tersedia yang dimiliki oleh buffer host penerima segmen yang bersangkutan. Buffer ini disebut sebagai Receive Buffer, digunakan untuk menyimpan byte stream yang datang. Tujuan hal ini adalah untuk mengatur lalu lintas data atau flow control.
Checksum	2 byte (16 bit)	Mampu melakukan pengecekan integritas segmen TCP (header-nya dan payload-nya). Nilai field

		Checksum akan diatur ke nilai 0 selama proses kalkulasi checksum.
Urgent Pointer	2 byte (16 bit)	Menandakan lokasi data yang dianggap "urgent" dalam segmen.
Options	4 byte (32 bit)	Berfungsi sebagai penampung beberapa opsi tambahan TCP. Setiap opsi TCP akan memakan ruangan 32 bit, sehingga ukuran header TCP dapat diindikasikan dengan menggunakan field Data offset.

c. UDP Header (berdasarkan RFC 768)



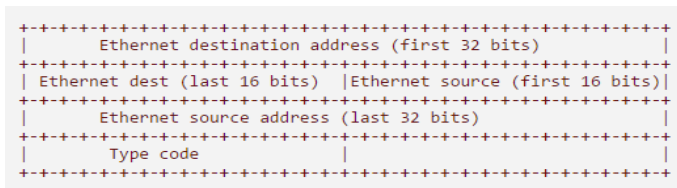
Gambar 4.3 Skema UDP Header

Tabel 4.3 Penjelasan Header UDP

Field	Panjang	Keterangan
Source Port	16 bit (2 byte)	Digunakan untuk mengidentifikasi sumber protokol lapisan aplikasi yang mengirimkan pesan UDP yang bersangkutan. Penggunaan field ini adalah opsional, dan jika tidak digunakan, akan diset ke angka 0. Beberapa protokol lapisan aplikasi dapat

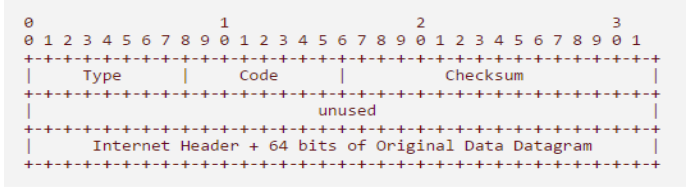
		menggunakan nilai field ini dari pesan UDP yang masuk sebagai nilai field port tujuan (Destination Port, lihat baris selanjutnya) sebagai balasan untuk pesan tersebut.
Destination Port	16 bit (2 byte)	Digunakan untuk mengidentifikasi tujuan protokol lapisan aplikasi yang menjadi tujuan pesan UDP yang bersangkutan. Dengan menggunakan kombinasi antara alamat IP dengan nilai dari field ini untuk membuat sebuah alamat yang signifikan untuk mengidentifikasi proses yang berjalan dalam sebuah host tertentu yang dituju oleh pesan UDP yang bersangkutan.
Length	16 bit (2 byte)	Digunakan untuk mengindikasikan panjang pesan UDP (pesan UDP ditambah dengan header UDP) dalam satuan byte.
Checksum	16 bit (2 byte)	Berisi informasi pengecekan integritas dari pesan UDP yang dikirimkan (header UDP dan pesan UDP). Penggunaan field ini adalah opsional. Jika tidak digunakan, field ini akan bernilai 0.

d. Ethernet Header



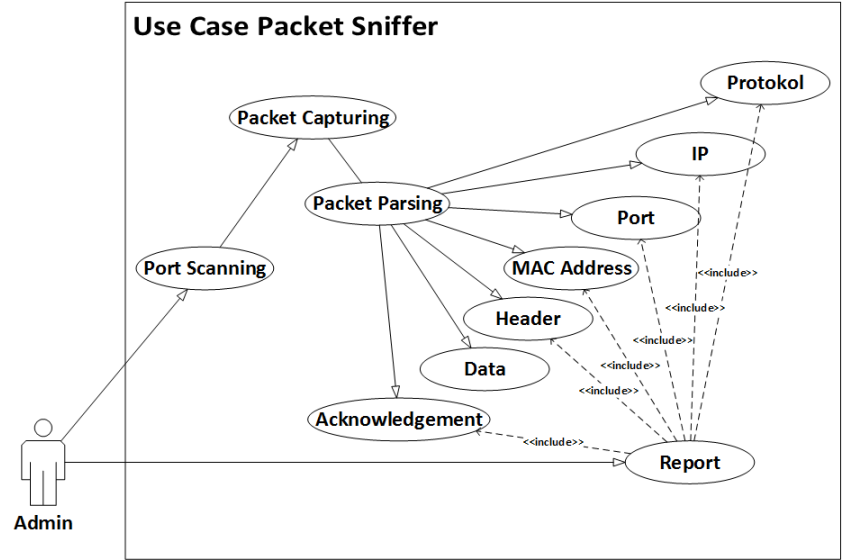
Gambar 4.4 Skema Ethernet Header

e. ICMP Header (berdasarkan RFC 792)



Gambar 4.5 Skema ICMP Header

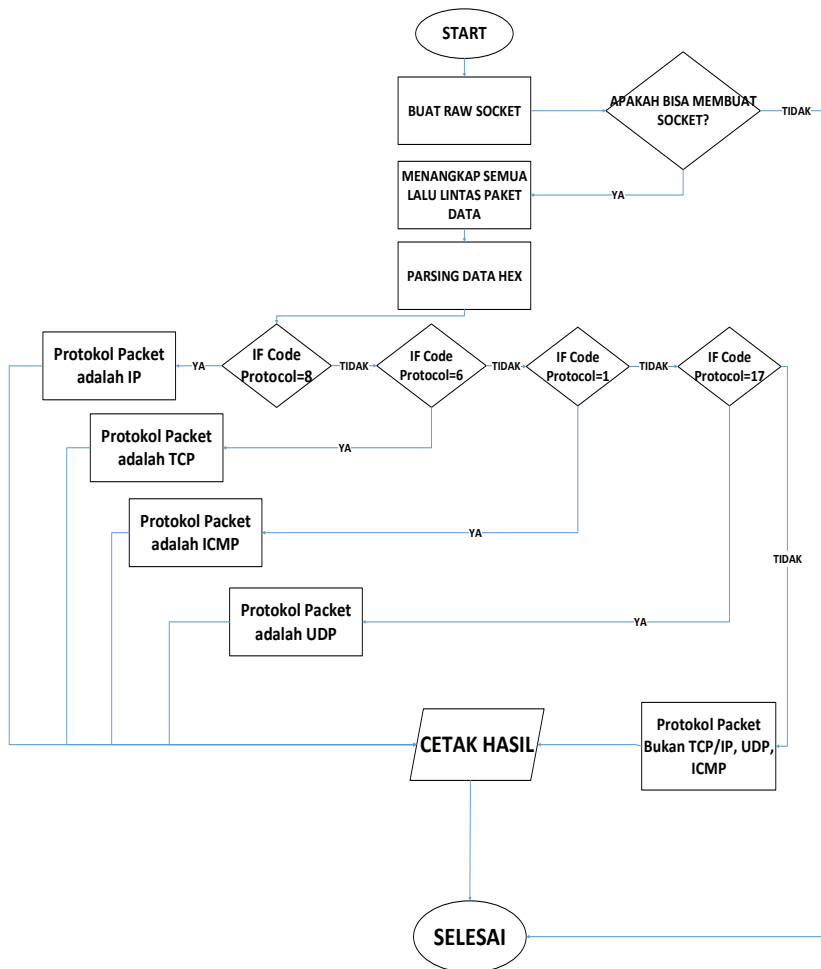
4.2.2 Use Case Diagram



Gambar 4.6 Use Case Diagram Packet Sniffer

Pada gambar 4.6 menunjukan use case diagram dari aplikasi packet sniffer. Diagram tersebut menunjukan interaksi aktor admin dengan fungsionalitas aplikasi.

4.2.3 Flowchart



Gambar 4.7 Flowchart Aplikasi

Pada gambar 4.7 menggambarkan merupakan flowchart dari jalannya aplikasi pada untuk dapat melakukan monitoring dan pencatatan lalu lintas packet data. Program akan membuat sebuah socket untuk dapat mencatat semua lalu lintas paket yang ada. Dari

data hasil monitoring kemudian dikonversikan agar dapat dibaca dan dianalisis berdasarkan tipe protokol header paket data. Hasil dari aplikasi akan menghasilkan report teks sebagai dokumentasi.

4.3 Implementasi, dan Hasil Aplikasi

Aplikasi dibangun dengan bahasa python versi 2.7 dengan menggunakan library “socket” dan “struct” python. Library “socket” digunakan digunakan karena mendukung low level networking interface. Sedangkan library “struct” digunakan sebagai struktur data dalam mengelola hasil parsing paket data.

a. Inisialisasi socket awal dan mengumpulkan packet data (proses collecting)

Tabel 4.4 Source code inisialisasi socket awal

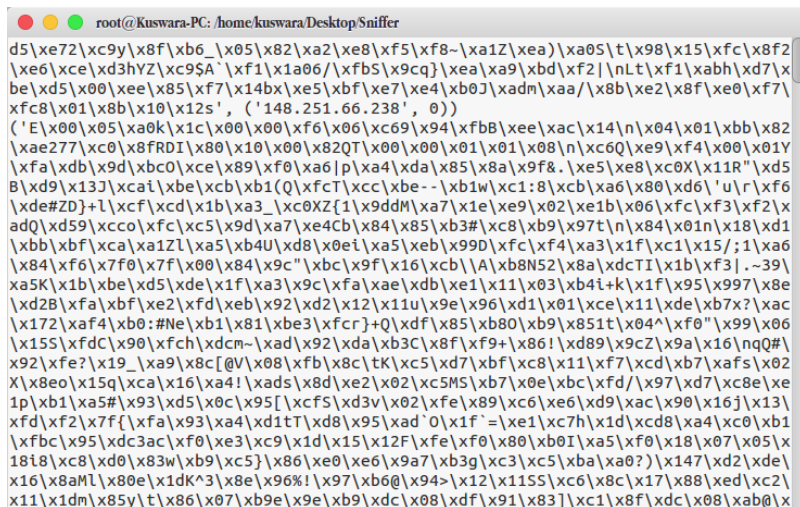
```
import socket, sys
from struct import *

def eth_addr (a) :
    b = "%.2x:%.2x:%.2x:%.2x:%.2x:%.2x" % ord(a[0]) ,
ord(a[1]), ord(a[2]), ord(a[3]), ord(a[4]) ,
ord(a[5]))
    return b

#create a AF_PACKET sebagai raw socket
try:
    s = socket.socket( socket.AF_PACKET ,
socket.SOCK_RAW , socket.ntohs(0x0003))
except socket.error , msg: print 'Socket could not be
created. Error Code : ' + str(msg[0]) + ' Message ' +
msg[1]
    sys.exit()
while True:
    packet = s.recvfrom(65565)
    packet = packet[0]
```

Source code pada table 4.4 berfungsi sebagai inisialisasi socket awal. Aplikasi membuka port 65565 sebagai mode listening. Jika port sudah digunakan dan tidak dapat membuka port maka akan muncul pesan error dan aplikasi akan berhenti. Ketika aplikasi dalam mode listening, aplikasi akan mengambil data packet jaringan dan

menyimpannya dalam index dengan menggunakan fungsi `packet = packet[0]`. Pada saat ini packet data jaringan yang didapatkan masih berupa data RAW yang merupakan hasil “low level networking interface” yang belum bisa dibaca oleh admin.



```

root@Kuswara-PC: /home/kuswara/Desktop/Sniffer
d5\xe72\xc9y\x8f\xb6_\x05\x82\xa2\xe8\xf5\xf8~\xa1Z\xea)\xa05\t\x98\x15\xfc\x8f2
\xe6\xce\xbd3hYz\xc95A` \xf1\x1a06/\xf5b5\x9cqj\xea\x9a\xbd\xf2/\nLt\xf1\xabhh\xd7\x
be\xbd5\x00\xee\x85\xf7\x14bx\xe5\xbf\xe7\xe4\xb0j\xadm\xaa/\x8b\xe2\x8f\xe0\xf7\
xfc8\x01\x8b\x10\x12s', ('148.251.66.238', 0))
('E\x00\x05\xa0k\x1c\x00\x00\xf6\x06\xc69\x94\xfbB\xee\xac\x14\n\x04\x01\xbb\x82
\xae277\xc0\x8fRDI\x80\x10\x00\x82QT\x00\x00\x01\x01\x08\n\xc6Q\x9e\x9f4\x00\x01Y
\xfa\xdb\x9d\xbc0\xce\x89\xf0\xa6\p\xa4\xda\x85\x8a\x9f8f.\xe5\xe8\x00X\x11R"\xd5
B\xd9\x13J\xcaI\xbe\xcb\xbb1(Q\xfcT\xcc\xbe--\xb1w\xci:8\xcb\xa6\x80\xde6\ 'u'\xf6
\xde#ZD]+l\xcf\xcd\x1b\xa3_\xc0XZ{1\x9ddM\xa7\x1e\xe9\x02\xe1b\x06\xfc\xfb3\xfb2\x
adQ\xd59\xcco\xfc\x5\x9d\xa7\xe4Cb\x84\x85\xb3#\xc8\xb9\x97t\n\x84\x0in\x18\xbd1
\xbb\xbf\xca\xa1Zl\xa5\xbd4U\xdb\x0eI\xa5\xeb\x99D\xfc\xfb4\xa3\x1f\xcc1\x15/;1\xa6
\x84\xf6\xf70\x00\x84\x9c"\xbcb\x9f\x16\xcb\A\x8b8N52\x8a\xdcTI\x1b\xfb3j.~39\
xa5K\x1b\xbe\xbd5\xde\x1f\xa3\x9c\xfa\xae\xdb\x1e1\x11\x03\xb4i+k\x1f\x95\x997\x8e
\xdb2B\xfa\xbf\x82\xfd\xeb\x92\xbd2\x12\x11u\x9e\x96\xbd1\x01\xce\x11\xde\xbd7x?\xac
\x172\xaf4\xb0:#Ne\xbb1\x81\xbe3\xfcrcj+Q\xdf\x85\xb80\xb9\x851t\x04^'\xf0"\x99\x06
\x155\xfdC\x90\xfc\xh\xdc~\xad\x92\xda\xbb3C\x8f\xfb9+\x86!\xd89\x9cZ\x9a\x16\nqQ#
\x92\xfe? \x19_\xa9\x8c[0V\x08\xfb\x8c\tK\x5\xbd7\xbf\xcb\x81\x11\xfb7\xcd\xbb7\xaf5\x02
X\x8e0\x15q\xca\x16\xa4!\xad5\x8d\xe2\x02\xc5M5\xb7\x0e\xbc\xfd/\x97\xbd7\xc8e\x
1p\xbb1\xa5#\x93\xbd5\x0c\x95[\xc5\xbd3v\x02\xfe\x89\xcc6\x0e\xbd9\xac\x90\x16j\x13\
\xfd\xfb2\x7f\xfa\x93\xa4\xdb1t\xdb8\x95\xad`0\x1f'=\xe1\xcc7h\x1d\xcd8\xa4\xcc0\xbb1
\xfbcb\x95\xdc3ac\xfb0\xe3\x9c\x9\x1d\x15\x12F\xfe\xfb0\x80\xbb0I\xa5\xfb0\x18\x07\x05\x
18i8\x8c\xbd0\x83w\xbb9\x5j\x86\x0e0\x0e6\x9a7\xb3g\xcc3\xcc5\xba\xa0?)\x147\xbd2\xde\
\x16\x8aMl\x80e\x1dk^3\x8e\x96%!\x97\xbb6@\x94>\x12\x1155\xcc6\x8c\x17\x88\xed\xcc2\
\x11\x1dm\x85y\t\x86\x07\xbb9e\x9e\xbb9\xdc\x08\xdf\x91\x83j\xcc1\x8f\xdc\x08\xab@\x

```

Gambar 4.8 Hasil Scan Paket data dalam Hex

b. Memparsing packet data RAW agar bisa dibaca (proses conversion)

Tabel 4.5 Source Code Parsing data RAW

```

eth_length = 14
eth_header = packet[:eth_length]
eth = unpack('!6s6sH' , eth_header)
eth_protocol = socket.ntohs(eth[2])

print 'Destination MAC : ' + eth_addr(packet[0:6])
+ ' Source MAC : ' + eth_addr(packet[6:12]) + '
Protocol : ' + str(eth_protocol)

if eth_protocol == 8 :
    #Parse IP header
    #ambil 20 karakter pertama untuk ip header
    ip_header =
packet[eth_length:20+eth_length]

```

```

#unpack
iph = unpack('!BBHHBBH4s4s' , ip_header)
version_ihl = iph[0]
version = version_ihl >> 4
ihl = version_ihl & 0xF
iph_length = ihl * 4
ttl = iph[5]
protocol = iph[6]
s_addr = socket.inet_ntoa(iph[8]);
d_addr = socket.inet_ntoa(iph[9]);
print 'Version : ' + str(version) + ' IP
Header Length : ' + str(ihl) + ' TTL : ' +
str(ttl) + ' Protocol : ' + str(protocol) + '
Source Address : ' + str(s_addr) + ' Destination
Address : ' + str(d_addr)

```

Pada source code diatas, paket yang telah berhasil dimonitor akan dilakukan parsing awal untuk mendapatkan versi dari paket data. Jika versi paket adalah 8, maka paket tersebut merupakan IP header, jika versi paket adalah 6, maka paket tersebut merupakan paket TCP, jika versi paket adalah 17, maka paket tersebut merupakan paket UDP, dan jika versi paket adalah 1, maka paket tersebut adalah paket ICMP.

Setelah didapatkan kode versi paket data, maka paket tersebut akan dilakukan parsing lebih lanjut dengan fungsi “unpack” sesuai dengan versi paket untuk diketahui informasi-informasi yang ada pada paket data. Informasi-informasi tersebut meliputi: destination MAC, source MAC, versi IP, jenis protokol, source address, destination address, source port, destination port, sequence number, acknowledgement, panjang header, isi data.

Tabel 4.6 Parsing paket TCP

```

#TCP protocol
if protocol == 6 :
    t = iph_length + eth_length
    tcp_header = packet[t:t+20]

    #now unpack them :)
    tcph = unpack('!HLLBBHHH',tcp_header)
    source_port = tcph[0]

```

```

        dest_port = tcph[1]
        sequence = tcph[2]
        acknowledgement = tcph[3]
        doff_reserved = tcph[4]
        tcph_length = doff_reserved >> 4
        print 'Source Port : ' + str(source_port)
+ ' Dest Port : ' + str(dest_port) + ' Sequence
Number : ' + str(sequence) + ' Acknowledgement : ' +
str(acknowledgement) + ' TCP header length : ' +
str(tcph_length)
        h_size = eth_length + iph_length +
tcph_length * 4
        data_size = len(packet) - h_size
        #get data from the packet
        data = packet[h_size:]
        print 'Data : ' + data

```

Source code pada tabel 4.6 merupakan source code yang akan mengkonversi paket TCP untuk didapatkan informasi yang ada pada paket sesuai dengan skema paket TCP sesuai RFC 791 seperti pada gambar 4.2.

Tabel 4.7 Parsing paket UDP

```

#UDP packets
    elif protocol == 17 :
        u = iph_length + eth_length
        udph_length = 8
        udp_header = packet[u:u+8]

        #unpack
        udph = unpack('!HHHH' , udp_header)
        source_port = udph[0]
        dest_port = udph[1]
        length = udph[2]
        checksum = udph[3]
        print 'Source Port : ' + str(source_port)
+ ' Dest Port : ' + str(dest_port) + ' Length : ' +
str(length) + ' Checksum : ' + str(checksum)
        h_size = eth_length + iph_length +
udph_length
        data_size = len(packet) - h_size
        #ambil informasi data dari packet
        data = packet[h_size:]
        print 'Data : ' + data

```

Source code pada tabel 4.7 merupakan source code yang akan mengkonversi paket UDP untuk didapatkan informasi yang ada pada paket sesuai dengan skema paket TCP sesuai RFC 768 seperti pada gambar 4.3.

Tabel 4.8 Parsing Paket ICMP

```
#ICMP Packets
    elif protocol == 1 :
        u = iph_length + eth_length
        icmp_length = 4
        icmp_header = packet[u:u+4]

        #unpack
        icmp_h = unpack('!BBH' , icmp_header)
        icmp_type = icmp_h[0]
        code = icmp_h[1]
        checksum = icmp_h[2]
        print 'Type : ' + str(icmp_type) + ' Code
: ' + str(code) + ' Checksum : ' + str(checksum)
        h_size = eth_length + iph_length +
icmp_length
        data_size = len(packet) - h_size
        #ambil informasi data paket
        data = packet[h_size:]
        print 'Data : ' + data
```

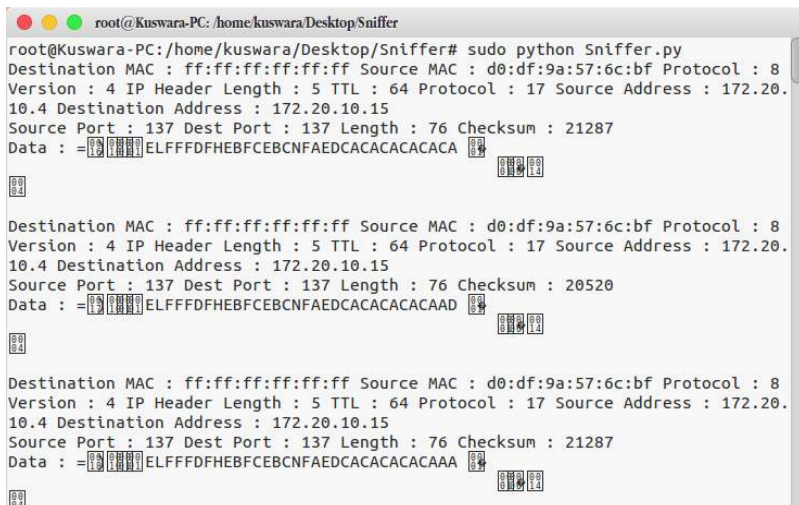
Source code pada tabel 4.8 merupakan source code yang akan mengkonversi paket ICMP untuk didapatkan informasi yang ada pada paket sesuai dengan skema paket TCP sesuai RFC 792 seperti pada gambar 4.5.

Namun jika versi paket yang didapatkan adalah selain dari TCP/IP, UDP, dan ICMP, maka akan diberikan pesan khusus yang menyatakan bahwa paket data tersebut diluar dari paket TCP/IP, UDP, dan ICMP seperti contohnya paket IGMP.

Tabel 4.9 Pesan khusus untuk protokol paket lain

```
else :
    print 'Protocol data bukan TCP/UDP/ICMP'
```

Hasil monitoring dan parsing paket data yang telah didapatkan, kemudian akan ditampilkan melalui layar terminal dari Linux.



```

root@Kuswara-PC: /home/kuswara/Desktop/Sniffer
root@Kuswara-PC:/home/kuswara/Desktop/Sniffer# sudo python Sniffer.py
Destination MAC : ff:ff:ff:ff:ff:ff Source MAC : d0:df:9a:57:6c:bf Protocol : 8
Version : 4 IP Header Length : 5 TTL : 64 Protocol : 17 Source Address : 172.20.
10.4 Destination Address : 172.20.10.15
Source Port : 137 Dest Port : 137 Length : 76 Checksum : 21287
Data : =\ELFFFDHBEBCBNFAEDCACACACACA
Destination MAC : ff:ff:ff:ff:ff:ff Source MAC : d0:df:9a:57:6c:bf Protocol : 8
Version : 4 IP Header Length : 5 TTL : 64 Protocol : 17 Source Address : 172.20.
10.4 Destination Address : 172.20.10.15
Source Port : 137 Dest Port : 137 Length : 76 Checksum : 20520
Data : =\ELFFFDHBEBCBNFAEDCACACACAAD
Destination MAC : ff:ff:ff:ff:ff:ff Source MAC : d0:df:9a:57:6c:bf Protocol : 8
Version : 4 IP Header Length : 5 TTL : 64 Protocol : 17 Source Address : 172.20.
10.4 Destination Address : 172.20.10.15
Source Port : 137 Dest Port : 137 Length : 76 Checksum : 21287
Data : =\ELFFFDHBEBCBNFAEDCACACACAAA

```

Gambar 4.8 Hasil Scan paket data yang telah diparsing

c. Proses menyimpan report (proses analysis)

Selain ditampilkan pada layar terminal, aplikasi juga bisa memberikan report berbasis teks untuk dapat dikelola dan dianalisis lebih lanjut oleh admin.

Tabel 4.10 Command line menyimpan report

```
sudo python Sniffer.py >> hasil.txt
```

Command line diatas akan menghasilkan file report dengan judul “hasil” berekstensi teks. Command line tersebut secara default bisa digunakan pada Linux.

```

hasil.txt  x  Openstack.txt  interfaces.txt  x  Sniffer.py  1.py
Destination MAC : 01:00:5e:7f:ff:fa Source MAC : cc:2d:83:87:18:47 Protocol : 8
Version : 4 IP Header Length : 5 TTL : 1 Protocol : 17 Source Address : 172.16.163.57
Destination Address : 239.255.255.250
Source Port : 48140 Dest Port : 1900 Length : 133 Checksum : 52893
Data : M-SEARCH * HTTP/1.1
HOST: 239.255.255.250:1900
MAN: "ssdp:discover"
MX: 1
ST: urn:dial-multiscreen-org:service:dial:1

Destination MAC : 33:33:00:00:00:02 Source MAC : 10:2a:b3:87:2d:99 Protocol : 56710
Destination MAC : 01:00:5e:7f:ff:fa Source MAC : cc:2d:83:87:18:47 Protocol : 8
Version : 4 IP Header Length : 5 TTL : 1 Protocol : 17 Source Address : 172.16.163.57
Destination Address : 239.255.255.250
Source Port : 48140 Dest Port : 1900 Length : 133 Checksum : 52893
Data : M-SEARCH * HTTP/1.1
HOST: 239.255.255.250:1900
MAN: "ssdp:discover"
MX: 1
ST: urn:dial-multiscreen-org:service:dial:1

Destination MAC : ff:ff:ff:ff:ff:ff Source MAC : e8:2a:ea:7e:9e:6a Protocol : 8
Version : 4 IP Header Length : 5 TTL : 128 Protocol : 17 Source Address : 172.16.163.22
Destination Address : 255.255.255.255
Source Port : 17500 Dest Port : 17500 Length : 218 Checksum : 50717
Data : {"host_int": 23753075469625735788239345970876795749, "version": [2, 0],
"displayname": "", "port": 17500, "namespaces": [598014625, 122390562, 539153227,
709558733, 1252070978, 300006998, 625129660, 326547615]}

```

Gambar 4.9 Hasil report berupa teks

Data-data yang terdapat report tersebut berupa:

- a. Destination MAC
- b. Source Mac
- c. Versi IP
- d. Jenis Protokol
- e. Source Address
- f. Destination Address
- g. Source port
- h. Dentination port
- i. Sequence number
- j. Acknowledgement
- k. Panjang header
- l. Data

4.4 Pengujian Aplikasi

Pada pengujian sistem, digunakan jenis pengujian fungsionalitas sistem. Pada pengujian fungsionalitas sistem, dilakukan pengujian untuk memastikan bahwa setiap fungsi sistem dari aplikasi sudah berjalan dengan baik. Berikut adalah tabel pengujian fungsionalitas sistem, yaitu sebagai berikut:

Tabel 4.11 Pengujian Fungsionalis Program

No	Pengguna	Fungsional	Hasil Test	Keterangan
1.	Admin	Memperoleh paket data jenis TCP/IP	Berhasil	Mampu mendapatkan informasi dari paket data berjenis TCP/IP
2.	Admin	Memperoleh paket data jenis UDP	Berhasil	Mampu mendapatkan informasi dari paket data berjenis UDP
3.	Admin	Memperoleh paket data jenis ICMP	Berhasil	Mampu mendapatkan informasi dari paket data berjenis ICMP
4.	Admin	Memperoleh alamat port sumber	Berhasil	Mampu mendapatkan informasi port sumber dari paket data
5.	Admin	Memperoleh alamat port tujuan	Berhasil	Mampu mendapatkan informasi port tujuan dari paket data

6.	Admin	Memperoleh alamat IP sumber	Berhasil	Mampu mendapatkan informasi alamat IP sumber dari paket data
7.	Admin	Memperoleh alamat IP tujuan	Berhasil	Mampu mendapatkan informasi alamat IP tujuan dari paket data
8.	Admin	Memperoleh data panjang header	Berhasil	Mampu mendapatkan informasi panjang header dari paket data
9.	Admin	Memperoleh MAC Address Sumber	Berhasil	Mampu mendapatkan informasi MAC Address sumber dari paket data
10.	Admin	Memperoleh MAC Address tujuan	Berhasil	Mampu mendapatkan informasi MAC Address tujuan dari paket data

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil kegiatan yang telah dilakukan, dapat disimpulkan beberapa hal terkait dengan aplikasi packet sniffer, yaitu sebagai berikut :

- a. Aplikasi packet sniffer dapat berjalan dengan baik pada system operasi Ubuntu dengan bahasa pemrograman Python.
- b. Aplikasi packet sniffer mampu untuk menangkap beberapa tipe packet data seperti TCP, UDP, ICMP.

5.2 Saran

Adapun saran yang dapat disampaikan yaitu agar aplikasi yang dibuat dapat dikembangkan lebih lanjut sehingga dapat menghasilkan report yang lebih baik.

DAFTAR PUSTAKA

- Agus Eka Pratama, Putu. (2014). Handbook Jaringan Komputer. Bandung: Penerbit Informatika.
- BinaryTides. 2011. network packet sniffer. (online). (<http://www.binarytides.com/python-packet-sniffer-code-linux/>)
- Computer Science Udayana University. (2016). *Struktur Organisasi*. Diambil kembali dari Jurusan Ilmu Komputer FMIPA Universitas Udayana: <http://www.cs.unud.ac.id/id/Struktur-Organisasi>
- Forouzan, B. A. (2007). Data Communications and Networking, 4th Edition. McGraw Hill
- J. Postel (1980). RFC 768 UDP. USC/Information Sciences Institute
- Oracle, (2013). The TCP/IP Protocol Suite. (online). (https://docs.oracle.com/cd/E23823_01/html/816-4554/ipov-6.html, diakses tanggal 1 Desember 2016)
- RFC 793 TCP. (1981). DARPA Internet Program Protocol Specification. Information Sciences Institute University of Southern California.
- RFC 792 ICMP. (1981). DARPA Internet Program Protocol Specification. Information Sciences Institute University of Southern California.

LAMPIRAN A
Form Aktivitas Harian PKL

AKTIVITAS HARIAN PKL

Nama : I Putu Kuswara Adi Pradana
 NIM : 1308605017
 Lokasi PKL : Lab. Jaringan, ILKOM, Univ. Udayana
 Waktu Pelaksanaan : 05 September 2016 - 25 November 2016

No.	Nama Penanggung Jawab/Jabatan	Pelaksanaan PKL			Keterangan
		Tanggal	Lokasi	Aktivitas	
1	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	05-09-2016	Lab NCC ILKOM Unud	Bimbingan ke pembimbing PKL untuk tugas PKL	
2	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	06-09-2016	Lab NCC ILKOM Unud	Pembagian tugas PKL dan pencarian tutorial terkait tugas yang diberikan	
3	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	07-09-2016	Lab NCC ILKOM Unud	Mempelajari topologi jaringan Ilmu Komputer	
4	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	08-09-2016	Lab NCC ILKOM Unud	Mempelajari topologi jaringan Ilmu Komputer	
5	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	09-09-2016	Lab NCC ILKOM Unud	Mempelajari topologi jaringan Ilmu Komputer	
6	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	10-09-2016	Lab NCC ILKOM Unud		
7	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	11-09-2016			

8	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	12-09-2016			
9	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	13-09-2016	Lab NCC ILKOM Unud	Mempelajari tutorial instalasi server VPN	
10	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	14-09-2016	Lab NCC ILKOM Unud	Mempelajari tutorial instalasi server VPN	
11	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	15-09-2016	Lab NCC ILKOM Unud	Membuat server ujicoba VPN	
12	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	16-09-2016	Lab NCC ILKOM Unud	Membuat server ujicoba VPN	
13	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	17-09-2016			
14	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	18-09-2016			
15	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	19-09-2016	Lab NCC ILKOM Unud	Membuat server ujicoba VPN	
16	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	20-09-2016	Lab NCC ILKOM Unud	Dokumentasi server VPN	
17	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	21-09-2016	Lab SI ILKOM Unud	Mempelajari materi untuk instalasi jaringan dan thin client di lab sistem informasi	
18	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	22-09-2016	Lab SI ILKOM Unud	Mempelajari materi untuk instalasi jaringan dan thin client di lab sistem informasi	

19	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	23-09-2016	Lab SI ILKOM Unud	Membeli alat-alat yang dibutuhkan untuk pembangunan lab sistem informasi	
20	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	24-09-2016			
21	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	25-09-2016			
22	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	26-09-2016	Lab SI ILKOM Unud	Instalasi jaringan dan thin client di lab sistem informasi	
23	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	27-09-2016	Lab SI ILKOM Unud	Instalasi jaringan dan thin client di lab sistem informasi	
24	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	28-09-2016	Lab SI ILKOM Unud	Instalasi jaringan dan thin client di lab sistem informasi	
25	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	29-09-2016	Lab SI ILKOM Unud	Maintenance data elearning ilkom	
26	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	30-09-2016	Lab SI ILKOM Unud	Maintenance data elearning ilkom	
27	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	01-10-2016			
28	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	02-10-2016			
29	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	03-10-2016	Lab SI ILKOM Unud	Instalasi jaringan dan thin client di lab sistem informasi	

30	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	04-10-2016	Lab SI ILKOM Unud	Instalasi jaringan dan thin client di lab sistem informasi	
31	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	05-10-2016	Lab SI ILKOM Unud	Pengujian jaringan dan thin client di lab sistem informasi	
32	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	06-10-2016	Lab SI ILKOM Unud	Pengujian jaringan dan thin client di lab sistem informasi	
33	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	07-10-2016	Lab SI ILKOM Unud	Pengujian jaringan dan thin client di lab sistem informasi	
34	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	08-10-2016			
35	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	09-10-2016			
36	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	10-10-2016	Lab SI ILKOM Unud	Dokumentasi instalasi jaringan dan thin client di lab sistem informasi	
37	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	11-10-2016	Lantai 2 Gedung BF ILKOM Unud	Survei pemansangan jaringan di lantai 2 Gedung BF	
38	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	12-10-2016	Lantai 2 Gedung BF ILKOM Unud	Instalasi jaringan di lantai 2 Gedung BF	
39	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	13-10-2016	Lantai 2 Gedung BF ILKOM Unud	Instalasi jaringan di lantai 2 Gedung BF	
40	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	14-10-2016	Lantai 2 Gedung BF ILKOM Unud	Instalasi jaringan di lantai 2 Gedung BF	

41	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	15-10-2016			
42	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	16-10-2016			
43	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	17-10-2016	Lantai 2 Gedung BF ILKOM Unud	Instalasi jaringan di Perpustakaan	
44	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	18-10-2016	Lantai 2 Gedung BF ILKOM Unud	Instalasi jaringan di Perpustakaan	
45	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	19-10-2016	Lantai 2 Gedung BF ILKOM Unud	Pengujian jaringan di Gedung Bf	
46	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	20-10-2016	Lantai 2 Gedung BF ILKOM Unud	Dokumentasi jaringan di Gedung Bf	
47	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	21-10-2016	Gedung BC ILKOM Unud	Survei pemasangan jaringan WiFi di Gedung BC	
48	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	22-10-2016			
49	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	23-10-2016			
50	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	24-10-2016	Gedung BC ILKOM Unud	Instalasi jaringan WiFi di Gedung BC	
51	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	25-10-2016	Gedung BC ILKOM Unud	Instalasi jaringan WiFi di Gedung BC	
52	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	26-10-2016	Gedung BC ILKOM Unud	Dokumentasi jaringan di Gedung Bc	

53	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	27-10-2016	Lab NCC ILKOM Unud	Merapikan lab NCC	
54	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	28-10-2016	Lab NCC ILKOM Unud	Merapikan lab NCC	
55	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	29-10-2016			
56	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	30-10-2016			
57	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	31-10-2016	Lab NCC ILKOM Unud	Merapikan lab jaringan jaringan	
58	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	01-11-2016	Gedung BD ILKOM Unud	Survei pemasangan jaringan di lantai 2 Gedung BD	
59	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	02-11-2016	Gedung BD ILKOM Unud	Instalasi jaringan WiFi di Gedung Bd	
60	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	03-11-2016			
61	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	04-11-2016			
62	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	05-11-2016			
63	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	06-11-2016	Gedung BD ILKOM Unud	Instalasi jaringan WiFi di Gedung Bd	

64	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	07-11-2016	Gedung BD ILKOM Unud	Instalasi jaringan WiFi di Gedung Bd	
65	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	08-11-2016			
66	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	09-11-2016			
67	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	10-11-2016			
68	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	11-11-2016	Gedung BD ILKOM Unud	Instalasi jaringan WiFi di Gedung Bd	
69	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	12-11-2016	Gedung BD ILKOM Unud	Instalasi jaringan WiFi di Gedung Bd	
70	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	13-11-2016	Gedung BD ILKOM Unud	Instalasi jaringan WiFi di Gedung Bd	
71	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	14-11-2016	Gedung BD ILKOM Unud	Instalasi jaringan WiFi di Gedung Bd	
72	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	15-11-2016			
73	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	16-11-2016			
74	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	17-11-2016	Gedung BD ILKOM Unud	Pembuatan infrastuktur jaringan lan dan instalasi jaringan untuk lab di lantai 2 gedung bd	

75	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	18-11-2016	Gedung BD ILKOM Unud	Pembuatan infrastuktur jaringan lan dan instalasi jaringan untuk lab di lantai 2 gedung bd	
76	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	19-11-2016	Gedung BD ILKOM Unud	Pembuatan infrastuktur jaringan lan dan instalasi jaringan untuk lab di lantai 2 gedung bd	
77	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	20-11-2016	Gedung BD ILKOM Unud	Pembuatan infrastuktur jaringan lan dan instalasi jaringan untuk lab di lantai 2 gedung bd	
78	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	21-11-2016	Gedung BD ILKOM Unud	Pembuatan infrastuktur jaringan lan dan instalasi jaringan untuk lab di lantai 2 gedung bd	
79	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	22-11-2016			
80	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	23-11-2016			
81	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	24-11-2016	Gedung BD ILKOM Unud	Pengecekan jaringan di Gedung BD	
82	I Gede Oka Gatria Atitama, S.Kom., M.Kom.	25-11-2016	Gedung BD ILKOM Unud	Dokumentasi jaringan di Gedung BD	

..... ,

Pembimbing Lapangan,

I Gede Oka Gatria Atitama, S.Kom., M.Kom.
NIP. 1991022620160312001

LAMPIRAN B
Dokumentas Harian PKL



Pembuatan Lab Timur Jurusan Ilmu Komputer FMIPA UNUD

1



Pembuatan Lab Timur Jurusan Ilmu Komputer FMIPA UNUD

2



**Test Lab Timur Menggunakan Sistem Operasi Ubuntu Pada
NCOMPUTING**



**Test Lab Timur Menggunakan Sistem Operasi Windows Pada
NCOMPUTING**



**Merapikan Kabel Switch Lab Barat Jurusan Ilmu Komputer
FMIPA UNUD**



Membuat Infrastruktur Wifi Di Jurusan Ilmu Komputer



**Membuat 3 Lab Di Gedung BF Jurusan Ilmu Komputer FMIPA
UNUD**



**Membuat 3 Lab Di Gedung BF Jurusan Ilmu Komputer FMIPA
UNUD 2**