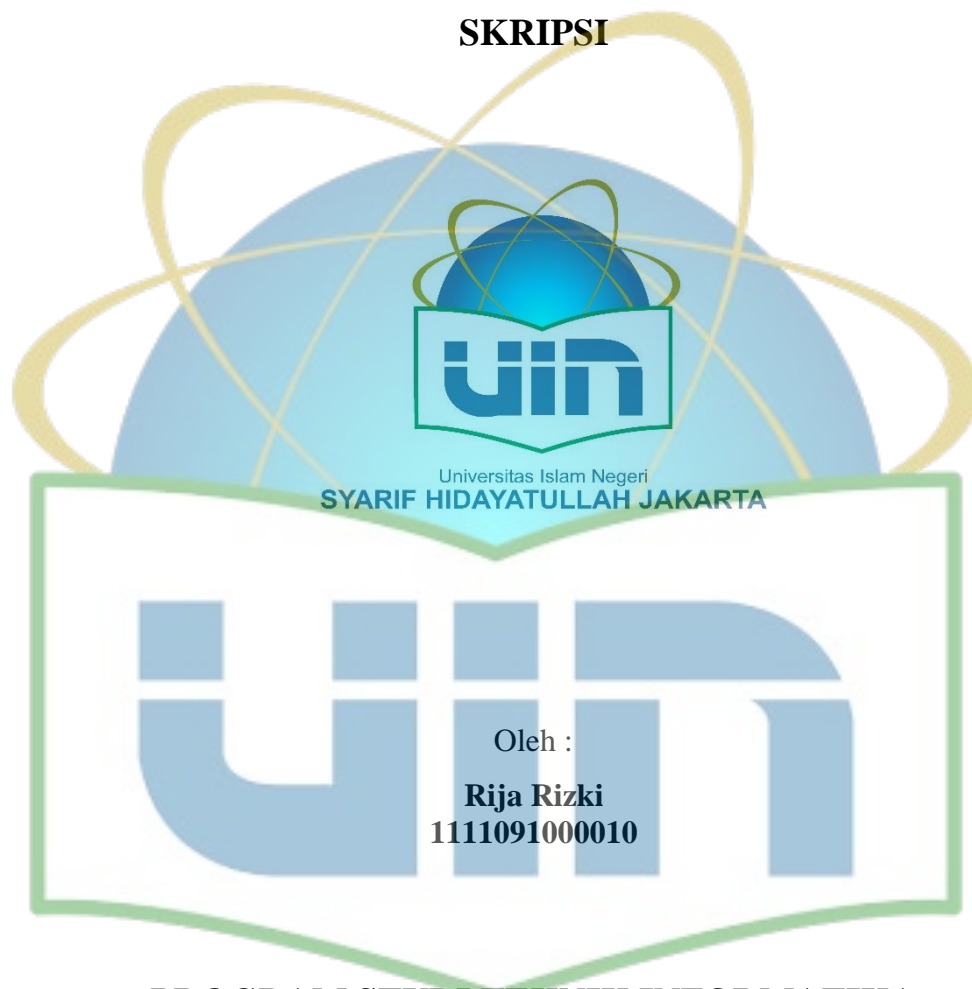


**PERANCANGAN SISTEM *MONITORING SERVER* DENGAN
MENGUNAKAN *BOT TELEGRAM* SEBAGAI MEDIA
NOTIFIKASI *ALERT*
(STUDI KASUS : PT. WAHANA PRESTASI LOGISTIK)**

SKRIPSI



**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI
SYARIF HIDAYATULLAH JAKARTA
2018M/1439H**

**PERANCANGAN SISTEM *MONITORING SERVER* DENGAN
MENGUNAKAN *BOT TELEGRAM* SEBAGAI MEDIA
NOTIFIKASI *ALERT***
(STUDI KASUS : PT. WAHANA PRESTASI LOGISTIK)

SKRIPSI

Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar
Sarjana Komputer



**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI
SYARIF HIDAYATULLAH JAKARTA
2018M/1439H**

LEMBAR PERSETUJUAN

PERANCANGAN SISTEM *MONITORING SERVER* DENGAN MENGUNAKAN *BOT TELEGRAM* SEBAGAI MEDIA NOTIFIKASI *ALERT*

(STUDI KASUS : PT. WAHANA PRESTASI LOGISTIK)

Skripsi

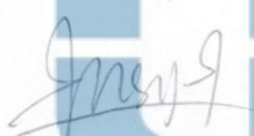
Sebagai Salah Satu Syarat untuk Memperoleh Gelar Sarjana Komputer Pada
Fakultas Sains dan Teknologi
Universitas Islam Negeri Syarif Hidayatullah Jakarta

Oleh :

Rija Rizki
1111091000010

Menyetujui,

Pembimbing I



Siti Umami Masruroh, M.Sc
NIP.198208232011012013

Pembimbing II


Hendra Bayu Suseno, M.Kom
NIP.198212112009011003

Mengetahui,

Ketua Program Studi Teknik Informatika,


Arini, MT
NIP.197601312009012001

HALAMAN PERNYATAAN ORISINALITAS

Dengan ini saya menyatakan bahwa :

1. Skripsi ini merupakan hasil karya asli saya yang diajukan untuk memenuhi salah satu persyaratan memperoleh gelar S1 di UIN Syarif Hidayatullah Jakarta.
2. Semua sumber yang saya gunakan dalam penulisan ini telah saya cantumkan sesuai dengan ketentuan yang berlaku di UIN Syarif Hidayatullah Jakarta.
3. Jika di kemudian hari terbukti bahwa karya ini bukan hasil karya asli saya atau merupakan hasil jiplakan dari karya orang lain, maka saya bersedia menerima sanksi yang berlaku di UIN Syarif Hidayatullah Jakarta.

Jakarta, Juli 2018



Rija Rizki

1111091000010

LEMBAR PENGESAHAN UJIAN

Skripsi yang berjudul “Perancangan Sistem Monitoring Server Dengan Menggunakan Bot Telegram Sebagai Media Notifikasi Alert (Studi Kasus : PT Wahana Prestasi Logistik)” telah diuji dan dinyatakan lulus dalam Sidang *Munaqosyah* Fakultas Sains dan Teknologi, Universitas Islam Negeri Syarif Hidayatullah Jakarta pada hari Kamis, 5 Juli 2018. Skripsi ini telah diterima sebagai salah satu syarat untuk memperoleh gelar Sarjana Strata Satu (S1) Program Studi Teknik Informatika.

Menyetujui,

Penguji I

Nashrul Hakiem, Ph.D
NIP. 197106082005011005

Penguji II

Nurhayati, Ph.D
NIP.196903161999032002

Pembimbing I

Siti Umami Masrurroh, M.Sc
NIP. 198208232011012013

Pembimbing II

Hendra Bayu Suseno, M.Kom
NIP.198212112009011003

Mengetahui,

Dekan Fakultas Sains dan Teknologi

Ketua Program Studi Teknik Informatika

Dr. Agus Salim, M.Si
NIP. 197208161999031003

Arini, MT
NIP. 197601312009012001

PERNYATAAN PERSETUJUAN PUBLIKASI SKRIPSI

Sebagai civitas akademik UIN Syarif Hidayatullah Jakarta, saya yang bertanda tangan dibawah ini:

Nama : Rija Rizki

NPM : 1111091000010

Program Studi : Teknik Informatika

Fakultas : Sains Dan Teknologi

Jenis Karya : Skripsi

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Islam Negeri Syarif Hidayatullah Jakarta Hak Bebas Royalti Noneksklusif (Non-exclusive Royalty Free Right) atas karya ilmiah saya yang berjudul:

**PERANCANGAN SISTEM *MONITORING SERVER* DENGAN
MENGUNAKAN *BOT TELEGRAM* SEBAGAI MEDIA NOTIFIKASI
*ALERT***

(STUDI KASUS : PT. WAHANA PRESTASI LOGISTIK)

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Islam Negeri Syarif Hidayatullah Jakarta berhak menyimpan, mengalih media/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Jakarta

Pada tanggal : 10 Juli 2018

Yang menyatakan

(Rija Rizki)

KATA PENGANTAR



Assalamu-alaikum Wr.Wb.

Puji syukur kehadiran Allah SWT atas petunjuk, rahmat, dan hidayah-Nya, penulis dapat menyelesaikan skripsi ini yang merupakan salah satu syarat untuk memperoleh gelar kesarjanaan Strata Satu (S1) bagi mahasiswa UIN Syarif Hidayatullah Jakarta, Program Studi Teknik Informatika, Fakultas Sains dan Teknologi.

Penulisan skripsi ini tidak akan tersusun dengan baik tanpa adanya bantuan dari pihak-pihak terkait. Oleh karena itu, pada kesempatan ini tidak lupa penulis mengucapkan banyak terimakasih kepada semua pihak yang telah membantu penulis dalam penyusunan skripsi ini.

Ucapan terimakasih yang sebesar-besarnya penulis sampaikan kepada :

1. Bpk. Dr.Agus Salim, M.Si selaku Dekan Fakultas Sains dan Teknologi.
2. Ibu Arini, MT selaku Ketua Program Studi Teknik Informatika.
3. Bpk. Feri Fahrianto, M.Sc selaku Sekretaris Program Studi Teknik Informatika.
4. Ibu Siti Umami Masruroh, M.Sc, selaku dosen pembimbing pertama, yang telah memberikan bantuan berupa bimbingan dan arahan dalam penulis menyelesaikan skripsi ini hingga selesai.
5. Bpk Hendra Bayu Suseno, M.Kom, selaku dosen pembimbing kedua, yang telah memberikan arahan serta kemudahan dalam penulis melakukan penelitian sehingga penulis dapat menyelesaikan skripsi ini dengan baik.

6. Orang tuaku, Bapak dan Ibu tercinta yang selalu memberikan semangat dan doa'nya sehingga penulis dapat menyelesaikan skripsi ini dengan baik.
7. Kepada manajemen PT. Wahana Prestasi Logistik umumnya dan khususnya kepada Bpk. Widi Raspito yang telah bersedia diwawancari dan membimbing penulis selama masa peneletian.
8. Seluruh dosen dan staff UIN Jakarta khususnya Fakultas Sains dan Teknologi yang telah memberikan ilmu dan pengalaman yang sangat berharga bagi penulis.
9. Dan kepada seluruh teman-teman Teknik Informatika A angkatan 2011.

Penulis menyadari bahwa skripsi ini masih jauh dari kesempurnaan, oleh karena itu kritik dan saran yang membangun sangat penulis harapkan demi kesempurnaan skripsi ini. Semoga skripsi ini dapat bermanfaat khususnya bagi penulis maupun bagi para pembaca.

Wassalamualaikum, Wr.Wb.

Jakarta, Juli 2018

Penulis

Rija Rizki

1111091000010

Penulis : Rija Rizki
Program Studi : Teknik Informatika
Judul : Perancangan Sistem *Monitoring Server* Dengan Menggunakan *Bot Telegram* Sebagai Notifikasi *Alert* (Studi Kasus PT Wahana Prestasi Logistik)

ABSTRAK

PT. Wahana Prestasi Logistik merupakan perusahaan yang bergerak dalam bidang jasa pengiriman barang, setiap harinya ada 40.000 – 45.000 barang yang dikirimkan ke *customer*. Data-data dari setiap barang tersebut disimpan dan diproses pada *server* yang dimiliki perusahaan. Oleh sebab itu pihak manajemen perusahaan sadar akan pentingnya kualitas kinerja *server* yang baik, demi kelancaran operasional perusahaan. Untuk menjaga agar *server* bekerja dengan baik, maka *server* harus terus dipantau, yang mana hal tersebut menjadi tanggung jawab seorang *server administrator*. Sistem berjalan yang digunakan untuk memonitoring *server* mengharuskan seorang *server administrator* terus *standby* didepan layar. Maka dari itu penulis mencoba menawarkan solusi untuk memonitoring *server* yang ada pada perusahaan dengan memanfaatkan *bot telegram* sebagai media notifikasi jika ada masalah pada *server*. Pada pengembangan sistem yang penulis tawarkan untuk memonitoring *server*, penulis menggunakan pendekatan metode *SPDLC* (*Security Policy Development Life Cycle*), dimana ada 5 tahapan dalam pengembangannya : *Analisis, Design, Implement, Enforcement, Enhancement*. Sistem monitoring server dengan menerapkan bahasa pemrograman *bash shell* yang penulis buat untuk memonitoring *server* pada PT. Wahana Prestasi Logistik, berjalan dengan baik dan dapat membantu *server administrator* memantau keadaan *server* ketika tidak berada di ruang monitoring server. Sehingga seorang *server administrator* tidak perlu selalu ada dan *standby* pada ruang monitoring *server*, karena adanya notifikasi pesan masuk pada aplikasi *telegram* yang ada pada *handheld/mobile device server administrator* secara *real time*, jika terjadi suatu masalah pada *server*.

Kata kunci : *telegram, bot telegram, server, server administrator, SPDLC, bash Shell.*

DAFTAR ISI

LEMBAR PERSETUJUAN.....	i
LEMBAR PERNYATAAN ORISINALITAS	ii
LEMBAR PENGESAHAN UJIAN	iii
PERNYATAAN PERSETUJUAN PUBLIKASI SKRIPSI	iv
KATA PENGANTAR	v
ABSTRAK.....	vii
DAFTAR ISI.....	viii
DAFTAR GAMBAR	xii
DAFTAR TABEL.....	xv
BAB I.....	1
PENDAHULUAN	1
1.1. Latar Belakang	1
1.2. Rumusan Masalah	2
1.3. Batasan Masalah.....	3
1.4. Tujuan Penelitian.....	3
1.5. Manfaat Penelitian.....	4
1.5.1. Bagi Penulis	4
1.5.2. Bagi Instansi.....	4
1.5.3. Bagi Universitas	5
1.6. Metode Penelitian.....	5
1.6.1. Metode Pengumpulan Data.....	5
1.6.2. Metode Pengembangan Sistem	6
1.7. Sistematika Penelitian	8
BAB II.....	10
LANDASAN TEORI.....	10
2.1. Sistem	10

2.2.	<i>Monitoring</i>	10
2.3.	<i>Server</i>	11
2.4.	<i>Web Server</i>	11
2.5.	<i>MySQL</i>	12
2.6.	<i>Telegram</i>	13
2.7.	<i>CentOS</i>	13
2.8.	<i>Ubuntu</i>	14
2.9.	<i>Bash Shell</i>	15
2.10.	<i>Vim</i>	15
2.11.	<i>SSH (Secure Shell)</i>	16
2.12.	<i>Stress-ng</i>	17
2.13.	<i>CPU</i>	17
2.14.	<i>RAM</i>	17
2.15.	<i>Disks</i>	18
2.15.1.	<i>SSD</i>	18
2.15.2.	<i>HDD</i>	19
2.16.	Metode Pengembangan Sistem.....	19
2.16.1.	<i>Analysis</i>	20
2.16.2.	<i>Design</i>	20
2.16.3.	<i>Implementation</i>	20
2.16.4.	<i>Enforcement</i>	20
2.16.5.	<i>Enhancement</i>	20
2.17.	Metode Pengumpulan Data.....	20
2.17.1.	Wawancara	21
2.17.2.	Studi Pustaka	22
2.17.3.	Observasi	22
2.17.4.	Studi Literatur Sejenis	23
BAB III	25

METODOLOGI PENELITIAN.....	25
3.1. Metode Pengumpulan Data	25
3.1.1. Wawancara.....	25
3.1.2. Observasi.....	25
3.1.3. Studi Pustaka.....	25
3.1.4. Tempat dan Waktu Penelitian	26
3.2. Metode Pengembangan Sistem	26
3.2.1. <i>Analysis</i>	26
3.2.2. <i>Design</i>	26
3.2.3. <i>Impelementation</i>	27
3.2.4. <i>Enforcement</i>	27
3.2.5. <i>Enhancement</i>	27
3.3. Kerangka Berpikir	28
BAB IV	29
ANALYSIS , DESIGN, DAN IMPLEMENTATION	29
4.1. <i>Analysis</i>	29
4.1.1. Identifikasi Masalah.....	29
4.1.2. Sistem Berjalan	30
4.1.3. Sistem Usulan	31
4.1.4. Kebutuhan Hardware Dan Software	31
4.2. <i>Design (Perancangan)</i>	33
4.2.1. Perancangan Sistem	33
4.2.2. <i>Flowchart Sistem</i>	34
4.3. <i>Implementation</i>	34
4.3.1. Pembuatan <i>Bot Telegram</i>	35
4.3.2. Konfigurasi Sistem <i>Monitoring Server</i>	40
4.3.3. Konfigurasi Notifikasi <i>Load CPU</i>	45
4.3.4. Konfigurasi Notifikasi <i>Memory Usage</i>	46

4.3.5.	Konfigurasi Notifikasi <i>Disks Usage</i>	47
4.3.6.	Konfigurasi Notifikasi <i>Service Mysql</i>	48
4.3.7.	Konfigurasi Notifikasi <i>Service HTTPD</i>	50
4.3.8.	Konfigurasi <i>Crontab</i>	52
BAB V.....		53
HASIL DAN PEMBAHASAN.....		53
5.1.	<i>Enforcement</i> (Pengujian).....	53
5.1.1.	Pengujian <i>Load CPU</i>	53
5.1.2.	Pengujian <i>Memory Usage</i>	54
5.1.3.	Pengujian <i>Disks Usage</i>	56
5.1.4.	Pengujian <i>Service HTTPD</i>	57
5.1.5.	Pengujian <i>Service Mysql</i>	59
5.1.6.	Pengujian <i>Server Down</i>	61
5.2.	<i>Enhancement</i> (Evaluasi).....	63
5.2.1.	Pemantauan <i>Load CPU</i>	63
5.2.2.	Pemantauan <i>Memory Usage</i>	64
5.2.3.	Pemantauan <i>Disks Usage</i>	65
5.2.4.	Pemantauan <i>Service HTTPD</i>	66
5.2.5.	Pemantauan <i>Service Mysql</i>	67
5.2.6.	Pemantauan <i>Server Down</i>	68
BAB VI		70
PENUTUP		70
6.1.	Kesimpulan.....	70
6.2.	Saran	70
DAFTAR PUSTAKA		72
LAMPIRAN		75

DAFTAR GAMBAR

Gambar 2.1 Tahapan-tahapan <i>SPDLC</i>	19
Gambar 3.1 Kerangka Berpikir	28
Gambar 4.1 Sistem Berjalan	30
Gambar 4.2 Sistem Usulan.....	31
Gambar 4.3 Perancangan Sistem.....	33
Gambar 4.4 <i>Flowchart</i> Sistem	34
Gambar 4.5 Tampilan <i>Home</i> Aplikasi <i>Telegram</i>	35
Gambar 4.6 Pencarian <i>BotFather</i>	36
Gambar 4.7 Memulai Pembuatan <i>Bot</i> Baru	36
Gambar 4.8 Pembuatan <i>Bot</i> Selesai	37
Gambar 4.9 Akses <i>Url</i> Untuk Menggunakan <i>Bot</i> Baru	38
Gambar 4.10 Memulai <i>Bot</i> Baru	38
Gambar 4.11 <i>Chat_id</i>	39
Gambar 4.12 Pengiriman Pesan Via <i>Browser</i>	40
Gambar 4.13 Pesan Masuk <i>Telegram</i>	40
Gambar 4.14 <i>SSH Dev Server</i>	41
Gambar 4.15 Pembuatan Direktori <i>Monserver</i>	42
Gambar 4.16 Pembuatan File <i>Monserver.sh</i>	42
Gambar 4.17 Merubah Hak Akses <i>Monserver.sh</i>	42
Gambar 4.18 Direktori Program	43
Gambar 4.19 Pembuatan File Konfigurasi.....	43
Gambar 4.20 Konfigurasi.....	44
Gambar 4.21 <i>Load CPU</i>	45
Gambar 4.22 Nilai <i>Load CPU</i>	46
Gambar 4.23 Pengiriman Pesan <i>Load CPU</i>	46
Gambar 4.24 <i>Memory Usage</i>	47

Gambar 4.25 Nilai <i>Memory Usage</i>	47
Gambar 4.26 Pengiriman Pesan <i>Memory Usage</i>	47
Gambar 4.27 <i>Disks Usage</i>	48
Gambar 4.28 Pengiriman Pesan <i>Disks Usage</i>	48
Gambar 4.29 <i>Service Mysql</i>	49
Gambar 4.30 Nilai Dari <i>Service Mysql</i>	49
Gambar 4.31 Pengiriman Pesan <i>Service Mysql</i>	50
Gambar 4.32 Nilai Dari <i>Service HTTPD</i>	50
Gambar 4.33 Pengiriman Pesan <i>Service HTTPD</i>	51
Gambar 4.34 Pengiriman Pesan <i>Server Down</i>	51
Gambar 4.35 Konfigurasi <i>Crontab</i>	52
Gambar 5.1 Pengujian <i>Load CPU</i>	53
Gambar 5.2 Pesan <i>Load CPU</i> Pada <i>Terminal</i>	53
Gambar 5.3 Pesan <i>Load CPU</i> Pada <i>Telegram</i>	54
Gambar 5.4 Pengujian <i>Memory Usage</i>	55
Gambar 5.5 Pesan <i>Memory Usage</i> Pada <i>Terminal</i>	55
Gambar 5.6 Pesan <i>Memory Usage</i> Pada <i>Telegram</i>	55
Gambar 5.7 Pengecekan <i>Disks Usage</i>	56
Gambar 5.8 Pesan <i>Disks Usage</i> Pada <i>Terminal</i>	56
Gambar 5.9 Pesan <i>Disks Usage</i> Pada <i>Telegram</i>	57
Gambar 5.10 Pengecekan <i>Service HTTPD Running</i>	57
Gambar 5.11 Pengecekan <i>Service HTTPD Dead</i>	58
Gambar 5.12 Pesan <i>Service HTTPD</i> Pada <i>Terminal</i>	58
Gambar 5.13 Pesan <i>Service HTTPD</i> Pada <i>Telegram</i>	59
Gambar 5.14 Pengecekan <i>Service Mysql Running</i>	59
Gambar 5.15 Pengecekan <i>Service Mysql Dead</i>	60
Gambar 5.16 Pesan <i>Service Mysql</i> Pada <i>Terminal</i>	60
Gambar 5.17 Pesan <i>Service Mysql</i> Pada <i>Telegram</i>	61

Gambar 5.18 Pesan <i>Server Down</i> Pada <i>Terminal</i>	62
Gambar 5.19 Pesan <i>Server Down</i> Pada <i>Telegram</i>	62
Gambar 5.20 Pemantauan <i>Load CPU</i> Sistem Berjalan.....	63
Gambar 5.21 Notifikasi <i>Load CPU</i> Pada <i>Telegram</i>	64
Gambar 5.22 Pemantauan <i>Memory Usage</i> Sistem Berjalan	64
Gambar 5.23 Notifikasi <i>Memory Usage</i> Pada <i>Telegram</i>	65
Gambar 5.24 Pemantauan <i>Disk Usage</i> Sistem Berjalan	65
Gambar 5.25 Notifikasi <i>Disk Usage</i> Pada <i>Telegram</i>	66
Gambar 5.26 Pemantauan <i>Service HTTPD</i> Sistem Berjalan	66
Gambar 5.27 Notifikasi <i>Service HTTPD</i> Pada <i>Telegram</i>	67
Gambar 5.28 Pemantauan <i>Service Mysql</i> Sistem Berjalan	67
Gambar 5.29 Notifikasi <i>Service Mysql</i> Pada <i>Telegram</i>	68
Gambar 5.30 Notifikasi <i>Server Down</i> Pada <i>Telegram</i>	68



DAFTAR TABEL

Table 2.1 Studi Literatur Sejenis.....	23
Table 4.1 Kebutuhan <i>Hardware</i> dan <i>Software</i>	32



BAB I

PENDAHULUAN

1.1. Latar Belakang

PT. Wahana Prestasi Logistik adalah perusahaan yang bergerak dalam bidang jasa pengiriman barang. Dimana setiap harinya hampir 40.000 sampai 45.000 barang yang diproses untuk dikirimkan ke *customer*. Semua data yang ada diproses pada *server* yang dimiliki PT. Wahana Prestasi Logistik, sehingga mereka menyadari betul, pentingnya kualitas kinerja *server* yang baik demi kelancaran operasional dan pekerjaan di perusahaan (Widi, wawancara, 17 Mei 2018).

Memiliki banyak *server* sebagai penunjang keoptimalan operasional dan pekerjaan dengan fungsi *server* yang berbeda-beda seperti *server* untuk *email*, aplikasi, *database* dan lainnya, menjadikan adanya kebutuhan untuk dapat mengelola *server-server* tersebut dengan baik. Oleh karena itu diperlukan sebuah fasilitas pendukung yaitu sistem *monitoring* agar *server administrator* dapat memonitor *server* ketika adanya masalah di *server*, meskipun tidak berada di depan komputer (Rasyid dkk, 2011).

Pada penelitian ini penulis memanfaatkan *bot telegram* sebagai media notifikasi ketika adanya gangguan atau masalah pada *server*. Diketahui bahwa *telegram* sebagai salah satu aplikasi *instant messaging*, dan sangat populer pada saat ini, mengklaim dapat menutupi beberapa kekurangan yang ada pada *whatsapp*. *Telegram* merupakan aplikasi *cloud based* dan alat enkripsi. *Telegram*

menyediakan enkripsi *end-to-end*, *self destruction messages*, dan infrastruktur *multi-data center* (Pinto, 2014).

Sebagai aplikasi pesan singkat yang *realtime*, *telegram* memberikan kemudahan akses bagi penggunanya, karena tersedia pada *platform mobile* maupun *desktop*. Pada *platform mobile telegram* dapat digunakan di *platform iphone*, *android* dan *windows phone*, sedangkan pada *platform desktop*, *telegram* dapat digunakan di *Windows*, *Linux*, *Mac OS* dan juga *Web browser*. *Telegram* mengklaim sebagai aplikasi pesan massal tercepat dan teraman yang berada di pasar. Selain itu *telegram* juga menyediakan wadah bagi pengembang yang ingin memanfaatkan *Open API* dan *Protocol* yang disediakan melalui pengembangan *telegram bot* yang didokumentasikan pada *web* resminya (Hamburger, 2014).

Ketika melakukan observasi penulis melihat bahwa *server administrator* harus terus *standby* untuk memantau aktifitas pada server. Dari masalah dan latar belakang yang ada maka penulis tertarik untuk melakukan penelitian mengenai masalah tersebut dengan judul ***“Perancangan Sistem Monitoring Server Dengan Menggunakan Bot Telegram Sebagai Notifikasi Alert (Studi Kasus PT. Wahana Prestasi Logistik)”***

1.2. Rumusan Masalah

Berdasarkan latar belakang diatas maka dapat disimpulkan 2 rumusan masalah diantaranya:

1. Bagaimana membangun dan merancang sistem *monitoring server* pada PT. Wahana Prestasi Logistik?

2. Bagaimana memanfaatkan *bot telegram* untuk mengirimkan notifikasi ke *server administrator* ketika adanya masalah pada *server* PT. Wahana Prestasi Logistik?

1.3. Batasan Masalah

1. Pengujian hanya pada lingkup *server* internal.
2. Tidak membahas tentang keamanan.
3. Pengujian gangguan *server* hanya pada *cpu*, *memory*, *disks*, *service mysql*, *service httpd*, dan *server down*.
4. Media notifikasi menggunakan *telegram*.
5. Membangun dan merancang sistem *monitoring server* menggunakan bahasa pemrograman *bash shell*.
6. Sistem operasi *server* menggunakan *CentOS 7* dan *Ubuntu 16.04 LTS*.
7. Metode pengembangan sistem menggunakan *SPDLC (Security Policy Development Life Cycle)*. *Analysis*, *Design*, *Implementation*, *Enforcement*, *Enhancement*.
8. Metode pengumpulan data menggunakan studi pustaka, observasi, dan wawancara.

1.4. Tujuan Penelitian

Adapun tujuan yang ingin dicapai oleh penulis dalam penelitian ini adalah sebagai berikut :

1. Merancang sistem *monitoring server* di PT Wahana Prestasi Logistik dengan menerapkan bahasa pemrograman *bash shell* untuk merancang sistem.
2. *Server administrator* akan menerima notifikasi jika adanya gangguan atau masalah pada *server* ketika *server administrator* tidak berada di ruang *monitoring server*.

1.5. Manfaat Penelitian

Setiap penelitian memiliki manfaatnya masing-masing, maka dari itu dengan adanya penelitian ini, penulis berharap memiliki beberapa manfaat diantaranya yaitu :

1.5.1. Bagi Penulis

1. Dapat menerapkan ilmu yang diperoleh selama kuliah.
2. Memperkaya literatur dan referensi tentang *server*.
3. Memahami bagaimana merancang dan membangun sistem *monitoring server* menggunakan *bash shell*.
4. Untuk memenuhi salah satu syarat kelulusan strata satu (S1),

Teknik Informatika, Fakultas Sains dan Teknologi, UIN Syarif Hidayatullah Jakarta.

1.5.2. Bagi Instansi

1. Dapat mengetahui hal yang terjadi pada *server* sehingga dapat langsung mengatasi masalah yang ada.

2. Membantu dan meringankan pekerjaan seorang *server administrator* instansi.
3. Meminimalisir terganggunya operasional dan pekerjaan instansi.

1.5.3. Bagi Universitas

1. Sebagai bahan referensi bagi penelitian-penelitian selanjutnya.
2. Mengetahui kemampuan mahasiswa dalam menguasai materi pelajaran yang diperoleh selama dibangku kuliah.
3. Sebagai bahan masukan dan evaluasi program Teknik Informatika di Universitas Islam Negeri Syarif Hidayatullah Jakarta.

1.6. Metode Penelitian

Dalam penulisan skripsi ini diperlukan data-data informasi yang lengkap sebagai bahan yang dapat mendukung kebenaran materi uraian dan pembahasan. Oleh karena itu dalam persiapannya penulis terlebih dahulu melakukan penelitian untuk mendapatkan data-data informasi atau bahan materi yang diperlukan. Metode yang akan digunakan untuk menyelesaikan skripsi ini adalah sebagai berikut :

1.6.1. Metode Pengumpulan Data

Metode pengumpulan data yang penulis gunakan yaitu :

1. Studi Pustaka

Mengumpulkan data dan informasi dengan cara membaca sumber sumber ilmiah dari buku, *ebook*, serta situs-situs internet sebagai referensi untuk mendapatkan informasi yang berkenaan

dengan topik yang dapat dijadikan acuan pembahasan dalam penelitian ini. Selain itu penulis juga melakukan studi literatur penelitian sejenis, dimana penulis mempelajari penelitian sejenis pada hasil penulisan karya ilmiah yang telah dilakukan sebelumnya sesuai dengan bahasan penelitian penulis.

2. Studi Lapangan

Pada metode pengumpulan data yang berdasarkan pada studi lapangan terdapat 2 metode yaitu :

a. Observasi

Metode yang digunakan penulis adalah metode observasi, yaitu dengan melakukan pengamatan terhadap aktivitas yang sedang terjadi di *server* PT. Wahana Prestasi Logistik

b. Wawancara

Penulis melakukan wawancara dengan pihak divisi IT PT. Wahana Prestasi Logistik yaitu Bapak Widi Raspito, untuk mendapatkan informasi tentang *server* yang ada pada PT. Wahana Prestasi Logistik.

1.6.2. Metode Pengembangan Sistem

Dalam penelitian skripsi ini, penulis menggunakan pendekatan terhadap model *SPDLC* (*Security Policy Development Life Cycle*) yang memiliki 5 tahapan yaitu:

1. *Analysis*

Merupakan analisis masalah dengan melakukan studi literatur serta mengidentifikasi masalah dan sistem seperti apa yang sudah ada dan yang akan diperlukan untuk mengatasi masalah tersebut serta analisis kebutuhan yang diperlukan.

2. *Design*

Merupakan tahapan dilakukannya perancangan sistem yang akan dibangun. Mulai dari *flowcart diagram* hingga alur kerja sistem yang dirancang.

3. *Implementation*

Merupakan tahapan dimana kebutuhan perangkat, penulisan kode program, serta penerapan rancangan arsitektur. Pada tahap ini, program dibuat sesuai rancangan pada tahapan *design* serta mengikuti alur diagram yang telah dibuat.

4. *Enforcement*

Merupakan percobaan dalam menjalankan program yang telah dirancang. Program diujicobakan agar setiap langkah dapat diamati saat melakukan pengujian. Data yang didapat pada tahapan ini menentukan apakah sistem dapat menjawab permasalahan yang telah dirumuskan.

5. *Enhancement*

Melakukan pengoptimalan dalam setiap komponen perancangan. Mulai dari *hardware*, *software* dan kebutuhan perancangan yang lain seperti pengawasan terhadap pengaruh jaringan serta penambahan tiap komponen yang diperlukan.

1.7. **Sistematika Penelitian**

Dalam penyusunan skripsi ini , pembahasan yang penulis sajikan terbagi dalam enam bab, yang secara singkat akan diuraikan sebagai berikut:

BAB I PENDAHULUAN

Bab ini membahas tentang latar belakang, perumusan masalah, pembatasan masalah, tujuan dan manfaat penelitian, metodologi penelitian dan sistematika penulisan.

BAB II LANDASAN TEORI

Bab ini membahas secara singkat teori dan konsep yang diperlukan dalam mendukung penelitian skripsi.

BAB III METODOLOGI PENELITIAN

Pada bab ini akan dijelaskan metode-metode yang digunakan penulis dalam melakukan penelitian skripsi.

BAB IV ANALISIS, PERANCANGAN , DAN IMPLEMENTASI

Pada bab ini akan dijelaskan implementasi perancangan sistem *monitoring server* menggunakan *bash shell* dan *telegram* sebagai media notifikasi pemberitahuan.

BAB V HASIL DAN PEMBAHASAN

Dalam bab ini diuraikan hasil dan pembahasan serta percobaan rancangan sistem *monitoring server* dengan notifikasi melalui *telegram*.

BAB VI PENUTUP

Bab ini berisi tentang kesimpulan serta saran dari apa yang telah diterangkan dan diuraikan pada bab bab sebelumnya mengenai hal yang perlu diperbaiki.



BAB II

LANDASAN TEORI

2.1. Sistem

Mulyadi (2008) menyatakan bahwa: "Suatu sistem pada dasarnya adalah sekelompok unsur yang erat hubungannya satu dengan yang lainnya, yang berfungsi bersama-sama untuk mencapai tujuan tertentu."

Sistem menurut Krismiaji (2010) merupakan "rangkaian komponen yang dikoordinasikan untuk mencapai serangkaian tujuan, yang memiliki karakteristik meliputi: komponen, atau sesuatu yang dapat dilihat, didengar atau dirasakan, proses, kegiatan untuk mengkoordinasikan komponen yang terlibat dalam sebuah sistem; tujuan, sasaran akhir yang ingin dicapai dari kegiatan koordinasi komponen tersebut."

Dari pengertian diatas dapat diambil suatu kesimpulan bahwa suatu sistem merupakan elemen yang saling berkaitan dan saling mempengaruhi dalam melakukan kegiatan bersama untuk mencapai suatu tujuan tertentu.

2.2. Monitoring

Monitoring dalam bahasa Indonesia dikenal dengan istilah pemantauan. *Monitoring* merupakan sebuah kegiatan untuk menjamin akan tercapainya semua tujuan organisasi dan manajemen (Handoko, 1995).

Monitoring juga didefinisikan sebagai langkah untuk mengkaji apakah kegiatan yang dilaksanakan telah sesuai dengan rencana, mengidentifikasi masalah

yang timbul agar langsung dapat diatasi, melakukan penilaian apakah pola kerja dan manajemen yang digunakan sudah tepat untuk mencapai tujuan, mengetahui kaitan antara kegiatan dengan tujuan untuk memperoleh ukuran kemajuan (Sutabri, 2012).

Dengan kata lain, *monitoring* merupakan salah satu proses didalam kegiatan organisasi yang sangat penting yang dapat menentukan terlaksana atau tidaknya sebuah tujuan organisasi. Tujuan dilakukannya *monitoring* adalah untuk memastikan agar tugas pokok organisasi dapat berjalan sesuai dengan rencana yang telah ditentukan (Aviana, 2012).

2.3. *Server*

Server adalah sebuah sistem komputer yang menyediakan jenis layanan (*service*) tertentu dalam sebuah jaringan komputer. *Server* didukung dengan prosesor yang bersifat *scalable* dan *RAM* yang besar, juga dilengkapi dengan sistem operasi khusus, yang disebut sebagai sistem operasi jaringan (*network operating system*). *Server* juga menjalankan perangkat lunak administratif yang mengontrol akses terhadap jaringan dan sumber daya yang terdapat di dalamnya, seperti halnya berkas atau alat pencetak (*printer*), dan memberikan akses kepada *workstation* anggota jaringan (www.it-jurnal.com).

2.4. *Web Server*

Web server adalah sebuah *software* yang memberikan layanan berbasis data dengan menggunakan protokol *HTTP* atau *HTTPS* dari *client* menggunakan aplikasi *web browser* untuk *request* data dan *server* akan mengirim data dalam bentuk

halaman *web* dan pada umumnya berbentuk dokumen *HTML* (Alam & Firmansyah, 2017).

Salah satu program dari *web server* adalah *apache*. *Apache* merupakan *web server* yang paling banyak dipergunakan di *internet*. Program ini pertama kali didesain untuk sistem operasi lingkungan *UNIX*, untuk saat ini telah tersedia *apache* yang didesain untuk sistem operasi lainnya. *Apache* mempunyai program pendukung yang cukup banyak. Hal ini memberikan layanan yang cukup lengkap bagi penggunaanya (Alam & Firmansyah, 2017).

Protokol yang digunakan untuk melayani fasilitas *web/www* ini menggunakan *HTTP*. *Apache* memiliki fitur-fitur canggih seperti pesan kesalahan yang dapat dikonfigurasi, autentikasi berbasis basis data dan lain sebagainya. *Apache* juga didukung oleh sejumlah antarmuka pengguna berbasis grafik (*GUI*) yang memungkinkan penanganan *server* menjadi mudah. *Apache* merupakan *software open source* dikembangkan oleh komunitas *open source* yang terdiri dari pengembang - pengembang dibawah naungan *Apache Software Foundation* (I Made, 2012).

2.5. MySQL

MySQL adalah sebuah *database manajemen system (DBMS)* populer yang memiliki fungsi sebagai *relational database manajemen system (RDBMS)*. Selain itu *MySQL software* merupakan suatu aplikasi yang sifatnya *open source* serta *server* basis data *MySQL* memiliki kinerja sangat cepat, reliable, dan mudah untuk

digunakan serta bekerja dengan arsitektur *client server* atau *embedded system* (www.mysql.com).

2.6. Telegram

Telegram adalah sebuah sistem perpesanan yang lintas *platform* dan berpusat pada keamanan kerahasiaan pribadi penggunanya, sedangkan *bot* adalah program komputer yang melakukan pekerjaan tertentu secara otomatis. *Bot* adalah sebuah mesin, dibuat memudahkan kehidupan keseharian tanpa harus terpaku di depan komputer. Jika ingin membuat *bot telegram*, diperlukan komunikasi utama dengan *server telegram* dilakukan melalui protokol *MTPROTO*, sebuah protokol *biner* buatan *telegram* sendiri. *Bot* yang paling terkenal adalah *telegram-bot* buatan Yugo Perez. *Bot-telegram cli* bekerja layaknya akun pribadi dan manfaat *bot* ini dijamin juga oleh *telegram* yang kemudian meluncurkan *bot API* (*Application Programming Interface*) agar orang banyak dapat membangun *bot* menggunakan bahasa pemrograman yang mereka kuasai tanpa harus berhubungan dengan *telegram-cli* atau *MTPROTO* (Ellis, 2014).

2.7. CentOS

CentOS (*Community ENTERprise Operating System*) merupakan *Distro Linux* yang cocok dipergunakan dalam skala *Enterprise* selain itu juga gratis. *CentOS* di buat dari *source code Red Hat Enterprise (RHEL)* dan dirilis dibawah *General Public License (GPL)* yang dikembangkan oleh sebuah komunitas yang disebut *CentOS Project* (www.centos.org).

Keuntungan linux *CentOS* adalah *open source*, kompatibilitas, *user friendly*. *CentOS* tersedia secara gratis, dukungan teknis utamanya disediakan terhadap para pengguna melalui milis, forum berbasis *web*, ataupun *chat*. Proyek *CentOS* tidak berafiliasi dengan *Red Hat*. Untuk penggalangan dana, *CentOS* berbasis donasi dari para pengguna serta sponsor dari perusahaan-perusahaan yang menggunakannya (Hidra dkk, 2014).

2.8. *Ubuntu*

Ubuntu berasal dari bahasa kuno Afrika, yang berarti "rasa perikemanusiaan terhadap sesama manusia". *Ubuntu* juga bisa berarti "aku adalah aku karena keberadaan kita semua". Tujuan dari distribusi Linux *Ubuntu* adalah membawa semangat yang terkandung di dalam *Ubuntu* ke dalam dunia perangkat lunak (ubuntu-id.org).

Ubuntu merupakan salah satu distro *Linux* yang berbasis *Debian*. *Ubuntu* disponsori oleh *Canonical*. Proyek *Ubuntu* mencoba bekerja sama dengan *Debian* dalam menanggapi persoalan yang tetap membuat banyak orang menggunakan *Debian*. *Ubuntu* menyediakan sistem yang berdasarkan *Debian* dengan frekuensi waktu rilis yang teratur, ketersediaan dukungan untuk pengguna perusahaan, dan tampilan desktop yang lebih dipertimbangkan. *Ubuntu* memfasilitasi penggunaanya dengan cara penyebaran yang digunakan oleh *Debian* dengan memberikan perbaikan keamanan, mengeluarkan perbaikan *bug* kritis, tampilan desktop yang konsisten, dan rilis yang berisi aplikasi terbaru dari dunia *open source* dalam enam bulan terakhir (idubuntu.org).

Ubuntu untuk tampilan *GUI* memakai *desktop Gnome*. *Desktop Gnome* ini merupakan salah satu tampilan *desktop* di *linux* selain *KDE* dan *XFCE*. *Cd* instalasi *Ubuntu* hanya 1 keping *cd*, 1 keping *cd* berisi paket instalasi *ubuntu*, perkantoran, permainan yang sangat terbatas. *Ubuntu* menyediakan paket-paket instalasi tambahan atau lebih sering disebut sebagai *repository* pada *server ubuntu* di *internet* atau pada 4 keping *dvd* repositori yang bisa di *download* (Much, 2006).

2.9. *Bash Shell*

Bourne Again Shell yang lebih sering kita sebut *Bash*, merupakan salah satu program dalam *GNU Project* yang menjadi *shell* yang paling banyak digunakan. *Bash* merupakan pengembangan dari *bourne shell*, Namanya juga diambil dari penciptanya yaitu *Stephen Bourne*. Kemudian *bash* dikembangkan oleh *Brian Fox*.

Saat ini, *bash* menjadi *script*' primer pada sistem *Linux* dan telah disertakan pula di *Mac OS X Tiger*. *Bash* juga memiliki bahasa pemrograman yang baik serta interaktivitas yang mudah di pahami. Selain *Bash*, *shell* yang ada ialah *C-shell* dan *Korn Shell*. Perbedaan ketiganya terlihat pada format penyimpanannya, yaitu *Bash* (.sh), *C-shell* (.csh), dan *Korn Shell* (.ksh) (www.codepolitan.com).

2.10. *Vim*

Vim ("*Vi Improved*") adalah sebuah "*vi clone*", yaitu sebuah program yang mirip dengan editor teks "*vi*". *Vim* bekerja dalam mode teks pada semua *terminal*, tetapi juga memiliki antar muka grafis, contohnya menu dan dukungan terhadap *mouse*. *Vim* bersifat *Open Source* dan tersedia dalam banyak *platform* dan memiliki dukungan-dukungan tambahan yang lebih banyak dibandingkan dengan *Vi*.

Vim lebih mudah digunakan untuk pemula dibandingkan dengan *Vi* karena *online help* yang luas, perintah-perintah "*undo*" dan "*redo*" dukungan terhadap *mouse* dan ikon dan menu yang dapat dikonfigurasi (*GUI*). *Vim* tersedia dalam sistem-sistem: *AmigaOS*, *Atari MiNT*, *BeOS*, *DOS*, *MacOS*, *NextStep*, *OS/2*, *OSF*, *RiscOS*, *SGI*, *UNIX*, *VMS*, *Win16* + *Win32* (*Windows95/98/00/NT*) - dan terutama *FreeBSD* dan *Linux* (www.vim.org).

2.11. SSH (*Secure Shell*)

Secure Shell (ssh) adalah suatu protokol yang memfasilitasi sistem komunikasi yang aman diantara dua sistem yang menggunakan arsitektur *client/server*, serta memungkinkan seorang *user* untuk *login* ke *server* secara *remote*. Berbeda dengan *telnet* dan *ftp* yang menggunakan *plain text*, *SSH* mengenkripsi data selama proses komunikasi sehingga menyulitkan penyusup/*intruder* yang mencoba mendapatkan *password* yang tidak dienkripsi. Fungsi utama aplikasi ini adalah untuk mengakses mesin secara *remote*. Bentuk akses *remote* yang bisa diperoleh adalah akses pada *mode* teks maupun *mode* grafis/*X* apabila konfigurasinya mengijinkan (Syarif, 2007).

Enkripsi yang digunakan oleh *SSH* menyediakan kerahasiaan dan integritas data melalui jaringan yang tidak aman seperti Internet. *SSH* menggunakan kriptografi kunci publik untuk mengautentikasi komputer remote dan membiarkan komputer remote untuk mengautentikasi pengguna, jika perlu. *SSH* biasanya digunakan untuk login ke mesin remote dan mengeksekusi berbagai perintah. *SSH* dapat digunakan untuk mentransfer file melalui *SFTP* atau *SCP*. *SSH* menggunakan

client-server model. Standar TCP port 22 telah ditetapkan sebagai jalur untuk server SSH. Sebuah klien SSH biasanya digunakan untuk membangun koneksi ke SSH daemon supaya dapat dikendalikan via remote (Batara dkk, 2013).

2.12. *Stress-ng*

Stress-ng adalah aplikasi yang digunakan untuk melakukan *stress test* pada sistem komputer seperti pengujian *cpu*, *ram*, *disk* dan lain sebagainya. Dimana aplikasi ini dirancang untuk menguji berbagai fisik subsistem pada sebuah komputer serta berbagai antarmuka kernel sistem operasi. *Stress-ng* memiliki banyak fitur untuk menguji secara spesifik, seperti untuk menguji *floating point*, *integer*, *bit manipulation* dan *control flow* pada *CPU* (manpages.ubuntu.com).

2.13. *CPU*

CPU (*Central Processing Unit*) atau sering dikenal dengan sebutan *Processor* adalah sebuah komponen berupa *chip* atau *IC* berbentuk persegi empat yang merupakan komponen utama pada sistem komputer sebagai pengendali proses kinerja komputer, dengan dibantu oleh komponen lainnya. Pada dasarnya fungsi *CPU* adalah menjalankan program-program yang disimpan dalam memori utama dengan cara mengambil instruksi-instruksi, menguji instruksi tersebut dan mengeksekusinya satu persatu sesuai alur perintah (Purwadi dkk, 2015).

2.14. *RAM*

RAM (*Random Access Memory*) adalah tempat penyimpanan sementara pada komputer yang isinya dapat diakses dalam waktu yang tetap, tidak memperdulikan letak data tersebut dalam memori atau acak. *RAM* sendiri bersifat

volatile artinya membutuhkan aliran listrik, berbeda dengan media penyimpanan lainnya seperti *flashdisk*, *hardisk* atau *cd/dvd* yang bersifat *non-volatile*. Fungsi dari sebuah *RAM* adalah untuk mempercepat pemrosesan data pada komputer, semakin besar *RAM* yang dimiliki, maka akan semakin cepat sebuah komputer memrosesnya (www.it-jurnal.com).

2.15. Disks

2.15.1. SSD

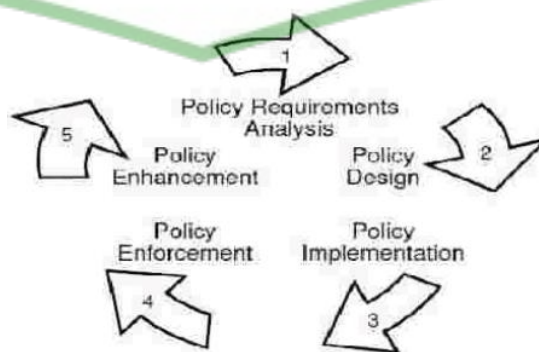
SSD singkatan dari *Solid State Drive* atau *Solid State Disk*, adalah perangkat penyimpan data yang menggunakan serangkaian *IC* sebagai memori yang digunakan untuk menyimpan data atau informasi. *SSD* menanamkan chip memori berbasis silikon sebagai media penyimpanan untuk menulis dan membaca data persisten. *SSD* juga dikenal sebagai *flash drive* atau kartu *flash*, dimasukkan ke dalam slot di komputer *server* - disebut sebagai penyimpanan *flash server-side* - atau sebagai bagian dari sistem penyimpanan berbagai perusahaan *flash*. *SSD* bisa dianggap sebagai versi canggih dari *USB Flash drive* dengan kapasitas yang jauh lebih besar dan berfungsi sebagai pengganti *Hardisk* yang selama ini digunakan pada perangkat komputer. Perbedaan *Hardisk (HDD)* bergerak secara mekanik untuk mengambil dan menyimpan data atau Informasi, sedangkan *SSD* bergerak secara elektrik untuk menyimpan dan mengambil data atau informasi (www.ubaya.ac.id).

2.15.2. HDD

Hard Disk Drive adalah salah satu jenis *storage* yang berupa piringan disk yang berputar sangat cepat, dilapisi dengan bahan magnetik, dan dengan kepala magnetik yang dipergunakan untuk membaca dan menulis data di permukaannya. *HDD* bekerja dengan peralatan mekanik. Untuk mengakses data, maka kepala magnetik akan diposisikan terlebih dahulu lalu akan memutar piringan. Untuk memposisikan kepala magnetik ini dibutuhkan waktu. Oleh karena itu, *file system* yang didesain untuk *HDD* dirancang untuk secepat mungkin untuk mencapai performa yang bagus (Wikimedia, 2012).

2.16. Metode Pengembangan Sistem

Security Policy Development Life Cycle (SPDLC) adalah suatu pendekatan proses dalam komunikasi data yang menggambarkan siklus yang tiada awal dan akhirnya dalam membangun sebuah jaringan komputer mencakup lima tahap, yaitu *Analysis, Design, Implementation, Enforcement* dan *Enhancement* (Wahsheh & Jim, 2008).



Gambar 2.1 Tahapan-tahapan *SPDLC*

2.16.1. *Analysis*

Analysis adalah sebuah proses yang dilakukan untuk pemecahan sebuah permasalahan.

2.16.2. *Design*

Design adalah proses penggambaran pemecahan masalah dengan solusi yang ditawarkan.

2.16.3. *Implementation*

Implementation adalah proses untuk mewujudkan sebuah sistem yang baru ke dalam sistem yang sebenarnya.

2.16.4. *Enforcement*

Enforcement adalah proses pengujian sistem dan penyelesaian dari kasus tersebut.

2.16.5. *Enhancement*

Enhancement adalah peningkatan atau perbaikan pada sistem untuk kebijakan selanjutnya.

2.17. Metode Pengumpulan Data

Teknik pengumpulan data adalah cara yang dapat digunakan oleh peneliti untuk mengumpulkan data (Abdurahman dkk, 2011). Pada bagian ini, akan ada tiga teknik pengumpulan data yang akan dibahas, yaitu antara lain : teknik wawancara, studi pustaka, dan observasi.

2.17.1. Wawancara

Wawancara merupakan salah satu teknik pengumpulan data yang dilakukan dengan cara mengadakan tanya jawab, baik secara langsung maupun tidak langsung secara bertatap muka dengan sumber data (responden). Wawancara langsung diadakan dengan orang yang menjadi satuan pengamatan dan dilakukan tanpa perantara. Sementara wawancara tidak langsung dilakukan terhadap seseorang yang dimintai keterangan tentang orang lain. Pengumpulan data melalui teknik wawancara biasanya digunakan untuk mengungkapkan masalah sikap dan persepsi seorang secara langsung dengan sumber data. Oleh karena itu, wawancara dapat dijadikan suatu alat pengumpulan data yang efektif, terutama karena:

1. Wawancara dapat dilaksanakan kepada setiap individu tanpa dibatasi oleh faktor usia maupun kemampuan membaca.
2. Data yang diperoleh dapat langsung diketahui objektivitasnya, karenadilaksanakan secara hubungan tatap muka atau *face to face relation*. Meskipun wawancara mempunyai manfaat, namun terdapat pula beberapa kelemahan diantaranya : (a) Oleh karena wawancara dilakukan secara perseorangan, maka pelaksanaannya menuntut banyak waktu, tenaga, dan biaya terutama bila ukuran sampel cukup besar. (b) Faktor bahasa, baik pewawancara maupun responden sangat mempengaruhi hasil data yang diperoleh (Abdurahman dkk, 2011).

2.17.2. Studi Pustaka

Tinjauan Pustaka atau *literature review* adalah bahan yang tertulis berupa buku, jurnal yang membahas tentang topik yang hendak diteliti. Tinjauan pustaka membantu peneliti untuk melihat ide-ide, pendapat, dan kritik tentang topik tersebut yang sebelumnya dibangun dan dianalisis oleh para ilmuwan sebelumnya. Tinjauan pustaka bertujuan untuk melihat dan menganalisa nilai tambah penelitian ini dibandingkan dengan penelitian-penelitian sebelumnya. Penggunaan metode penelitian baik kualitatif maupun kuantitatif akan membahas tinjauan pustaka pada awal penelitian dengan tujuan untuk peneguhan atas pentingnya masalah atau topik penelitian yang akan dibahas (Raco, 2010).

2.17.3. Observasi

Teknik observasi merupakan salah satu teknik pengumpulan data dimana peneliti mengadakan pengamatan dan pencatatan secara sistematis terhadap objek yang diteliti, baik dalam situasi buatan yang secara khusus diadakan (laboratorium) maupun dalam situasi alamiah atau sebenarnya (lapangan). Pengumpulan data melalui teknik observasi biasanya digunakan sebagai alat untuk mengukur tingkah laku individu ataupun proses terjadinya suatu kejadian yang dapat diamati, baik dalam situasi buatan yang secara khusus diadakan maupun dalam situasi alamiah atau sebenarnya. Alat pengumpulan data dalam teknik observasi adalah berupa catatan informal, daftar cek, skala, penilaian, dan pencatatan dengan alat. Catatan informal

merupakan pencatatan data yang biasa dilakukan dalam observasi yang tidak berstruktur (Abdurahman dkk, 2011).

2.17.4. Studi Literatur Sejenis

Pada bagian ini akan dijabarkan beberapa penelitian-penelitian sejenis yang sudah ada sebelumnya, penelitian tersebut dapat berupa skripsi maupun jurnal dengan tema sejenis yang dapat dijadikan acuan oleh peneliti dalam proses penulisan skripsi maupun pengembangan sistem. Adapun penelitian sejenis yang dijadikan referensi oleh peneliti adalah sebagai berikut :

Table 2.1 Studi Literatur Sejenis

Judul	Penulis	Kelebihan	Kekurangan
Implementasi Sistem Monitoring Server Menggunakan Nagios	Juli Yanto Seminar Nasional Telekomunikasi dan Informatika (SELISIK 2016)	Auto start service yang dimonitor jika down.	Notifikasi menggunakan email sehingga tidak real time.
Sistem Monitoring Server Dan Perangkat Jaringan Pada Enterprise Resource Planning Fasilkom Unsri Menggunakan	Ahmad Heryanto, Adi Hermansyah, dan M. Nizar Jurnal SISTEMASI,	Aplikasi monitoring mampu membuat report terhadap kondisi	Notifikasi menggunakan sms gateway yang membutuhkan biaya

Protokol ICMP Dan SNMP	Volume 6, Nomor 3, September 2017 : 1 – 10	perangkat jaringan	
Nagios Untuk Monitoring Server Dengan Pengiriman Notifikasi Gangguan Server Menggunakan Email Dan Sms Gateway (Studi Kasus : Pt. Gamatechno Indonesia – Yogyakarta)	Nurul Fatmawati Asri, Amir Hamzah, Muhammad Sholeh Jurnal JARKOM Vol. 1 No. 2 Januari 2014	Mempunyai 2 cara dalam mengirimkan notifikasi	Notifikasi menggunakan email dan sms gateway yang membutuhkan biaya

BAB III

METODOLOGI PENELITIAN

3.1. Metode Pengumpulan Data

Pada penelitian ini, penulis menggunakan beberapa metode yang bertujuan untuk pengumpulan data sebagai pendukung. Beberapa metode yang penulis gunakan untuk pengumpulan data adalah sebagai berikut :

3.1.1. Wawancara

Penulis melakukan wawancara dengan divisi IT PT. Wahana Prestasi Logistik yaitu dengan Bapak Widi Raspito selaku manager IT, untuk mengumpulkan data primer yang berkaitan dengan topik penelitian. Hasil dari wawancara penulis lampirkan pada lampiran.

3.1.2. Observasi

Penulis melakukan pengamatan langsung ke lapangan yang menjadi lokasi penelitian penulis, yaitu PT. Wahana Prestasi Logistik. Pengamatan yang dilakukan meliputi pengamatan infrastruktur *server* yang ada dan cara memonitoring *server*.

3.1.3. Studi Pustaka

Untuk melengkapi kebutuhan informasi yang diperlukan dalam penulisan skripsi ini, penulis mendapatkan informasi dari beberapa referensi yang diperoleh dari buku-buku yang terdapat didalam perpustakaan dan

beberapa artikel atau buku elektronik yang diperoleh dari media *internet*, penulis juga melakukan studi literatur penelitian sejenis.

3.1.4. Tempat dan Waktu Penelitian

Dalam penulisan skripsi ini, penulis melakukan penelitian di PT. Wahana Prestasi Logistik yang beralamat di Jl. Rempoa Raya No.88, Rempoa, Ciputat, 15412 Tangerang Selatan, Banten. Penelitian dilakukan dalam jangka waktu 2 bulan.

3.2. Metode Pengembangan Sistem

Pada penelitian ini, penulis menggunakan metode pengembangan *SPDLC* (*Security Policy Development Life Cycle*). Berikut adalah tahapan-tahapan penelitian menggunakan metode pengembangan *SPDLC*.

3.2.1. Analysis

Merupakan analisis masalah dengan melakukan studi literatur mengenai *bot telegram* untuk notifikasi *monitoring server* serta mengidentifikasi masalah dan sistem seperti apa yang diperlukan untuk mengatasi masalah tersebut.

3.2.2. Design

Merupakan tahapan dilakukannya perancangan sistem yang akan dibangun. Mulai dari *flowcart diagram* hingga alur kerja sistem yang dirancang.

3.2.3. *Impelementation*

Merupakan tahapan dimana kebutuhan perangkat, penulisan kode program *bot* serta penerapan rancangan arsitektur *bot telegram*. Pada tahap ini, program dibuat sesuai rancangan pada tahapan *design* serta mengikuti alur diagram yang telah dibuat.

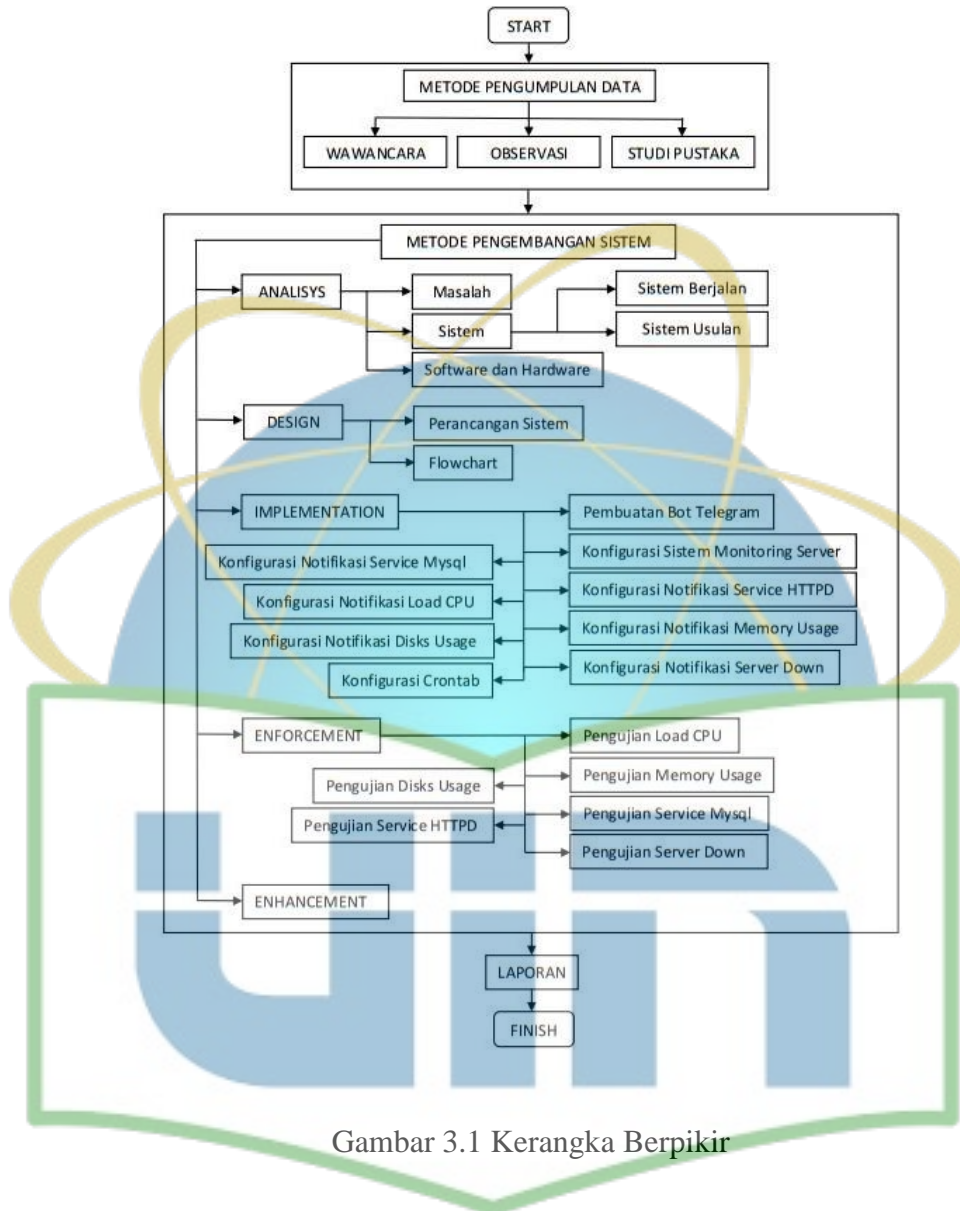
3.2.4. *Enforcement*

Merupakan percobaan dalam menjalankan program *bot telegram* yang telah dirancang. Program *bot* diujicobakan ke dalam *server* agar setiap langkah dapat diamati saat melakukan pengujian. Data yang didapat pada tahapan ini menentukan apakah sistem dapat menjawab permasalahan yang telah dirumuskan.

3.2.5. *Enhancement*

Melakukan pengoptimalan dalam setiap komponen perancangan. Mulai dari *hardware*, *software* dan kebutuhan perancangan *bot* yang lain seperti pengawasan terhadap pengaruh jaringan serta penambahan tiap komponen yang diperlukan.

3.3. Kerangka Berpikir



Gambar 3.1 Kerangka Berpikir

BAB IV

ANALYSIS , DESIGN, DAN IMPLEMENTATION

Pada bab ini penulis akan menjelaskan semua proses perancangan dan pembuatan sistem *monitoring server* pada PT. Wahana Prestasi Logistik yang beralamat di Jl. Rempoa Raya No.88, Rempoa, Ciputat, 15412, Tangerang Selatan, Banten. Dalam hal ini akan lebih difokuskan pada penjelasan mengenai perancangan sistem *monitoring server* dimana *server* menggunakan sistem operasi *CentOS 7* hingga siap digunakan sebagai sistem deteksi jika ada masalah pada *server* melalui media notifikasi *telegram* sebagai pesan pemberitahuan kepada *server administrator* di PT. Wahana Prestasi Logistik.

4.1. Analysis

4.1.1. Identifikasi Masalah

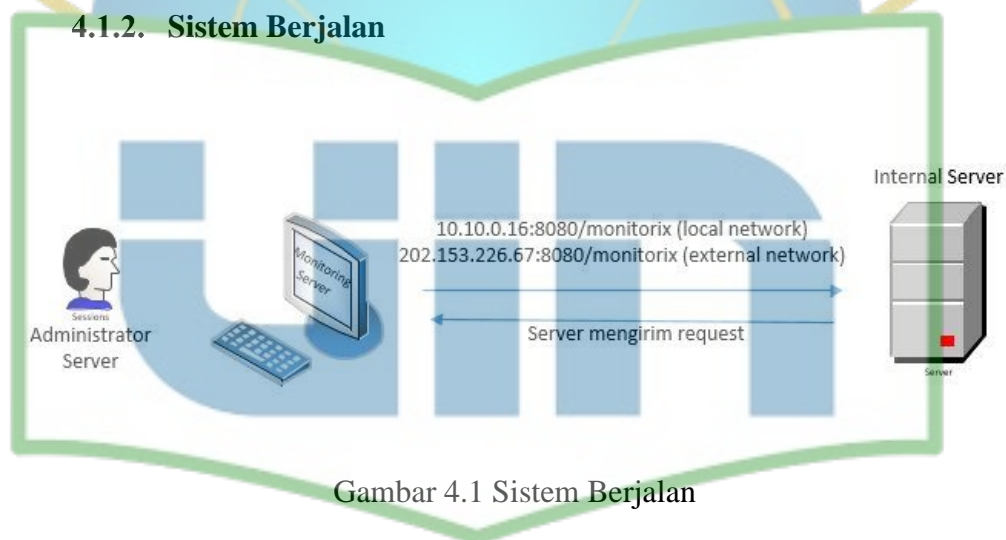
PT. Wahana Prestasi Logistik adalah perusahaan multinasional yang bergerak di bidang jasa pengiriman barang. Dimana setiap harinya hampir 40.000 sampai 45.000 barang yang diproses untuk dikirimkan ke *customer*. Semua data yang ada diproses pada *server* yang dimiliki PT. Wahana Prestasi Logistik, sehingga *server* harus selalu dalam kondisi yang baik dan prima.

Pada kasus seorang *server administrator* yang harus setiap saat memantau kondisi *server* agar berjalan normal tanpa ada gangguan dari

dalam maupun dari luar, pada kondisi ini *server* sangat bergantung pada kesiagaan *server administrator* dalam menjaga kondisi *server*.

Pada hasil wawancara dan observasi yang dilakukan, untuk memonitoring *server* yang ada pada PT. Wahana Prestasi Logistik, sudah tersedianya sistem yang dijalankan, tetapi *server administrator* harus terus berada di depan layar untuk memonitoring dan tidak adanya pemberitahuan secara *real time* kepada *server administrator*, jika *server administrator* tidak ada di ruang *monitoring server*. Hal ini pula yang mendorong penulis ingin membuat sistem *monitoring server* dengan media pemberitahuan notifikasi secara *real time* yaitu *telegram*.

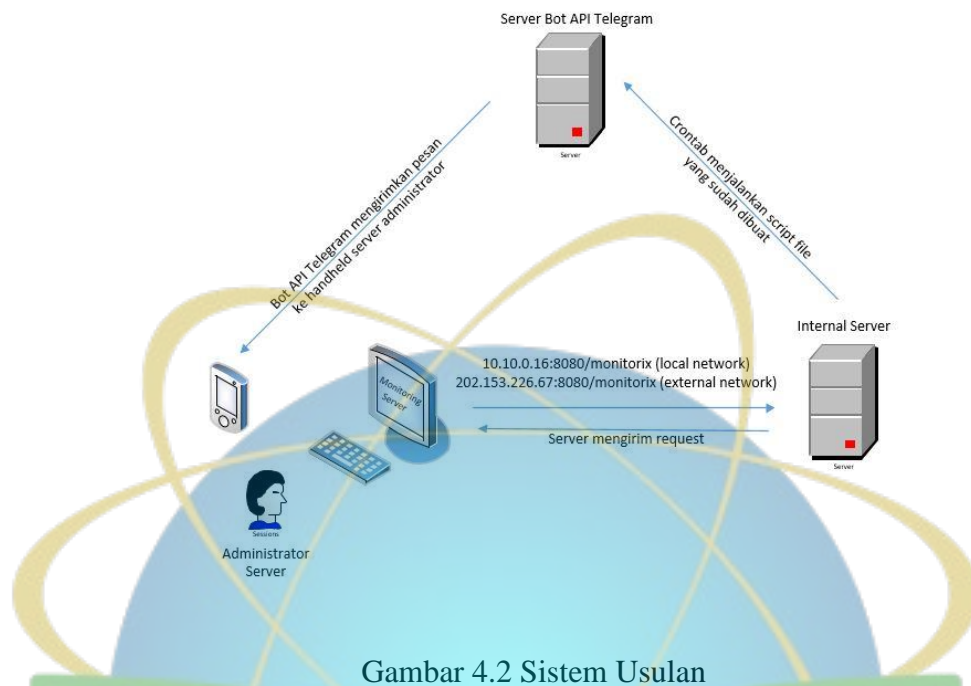
4.1.2. Sistem Berjalan



Gambar 4.1 Sistem Berjalan

Dari gambar sistem yang berjalan dapat disimpulkan bahwa *server administrator* harus *standby* di depan monitor untuk memantau kondisi *server*. Sistem monitoring akan bekerja memantau semua kondisi yang terjadi di *server*. Jika terjadi suatu masalah pada *server*, maka *server administrator* akan mengetahuinya dengan cara melihat sistem monitoring.

4.1.3. Sistem Usulan



Gambar 4.2 Sistem Usulan

Dari gambar sistem usulan dapat dilihat bahwa *server administrator* tidak harus *standby* untuk memonitoring *server* karena *server administrator* akan mendapatkan notifikasi secara *real time* jika ada masalah pada *server* melalui *telegram*.

4.1.4. Kebutuhan Hardware Dan Software

Bedasarkan hasil analisis yang diperoleh untuk membangun sistem *monitoring server* melalui *telegram* sebagai media notifikasinya, maka dibutuhkan beberapa komponen baik dalam bentuk *hardware* maupun *software* sebagai berikut.

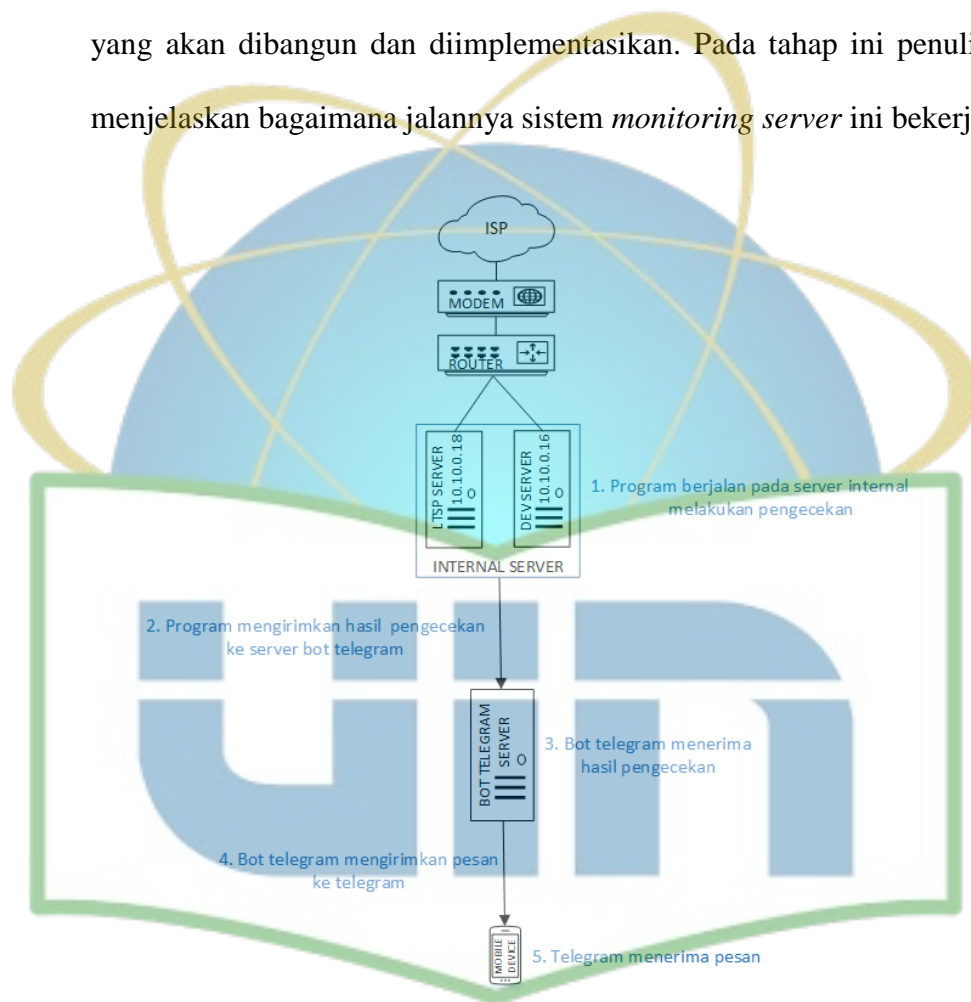
Table 4.1 Kebutuhan *Hardware* dan *Software*

NO	PERANGKAT	HARDWARE	SOFTWARE
1	SERVER 1	Prosesor intel core i5-2320 3.0 GHz, Ram 8GB, Disk 500GB dan 1TB, Lan card intel 82579v gigabit network	OS CentOS 7, Mysql, Httpd, Crontab, Command line, Vim
2	SERVER 2	Prosesor Intel(R) Core(TM) i5-3330 CPU @ 3.00GHz, Ram 16GB, Disk 250GB dan 4TB, Land card RTL8111/8168/8411 PCI Express Gigabit Ethernet Controller	OS Ubuntu 16.04 LTS, Command line
3	MOBILE DEVICE	Prosesor SD 660, Ram 6GB, Internal 64GB, Network GSM/HSDPA/LTE, Wlan Wi-Fi 802.11 a/b/g/n/ac, Batre 3500mAh	OS Android 7 Nougat, Telegram, Juice SSH

4.2. Design (Perancangan)

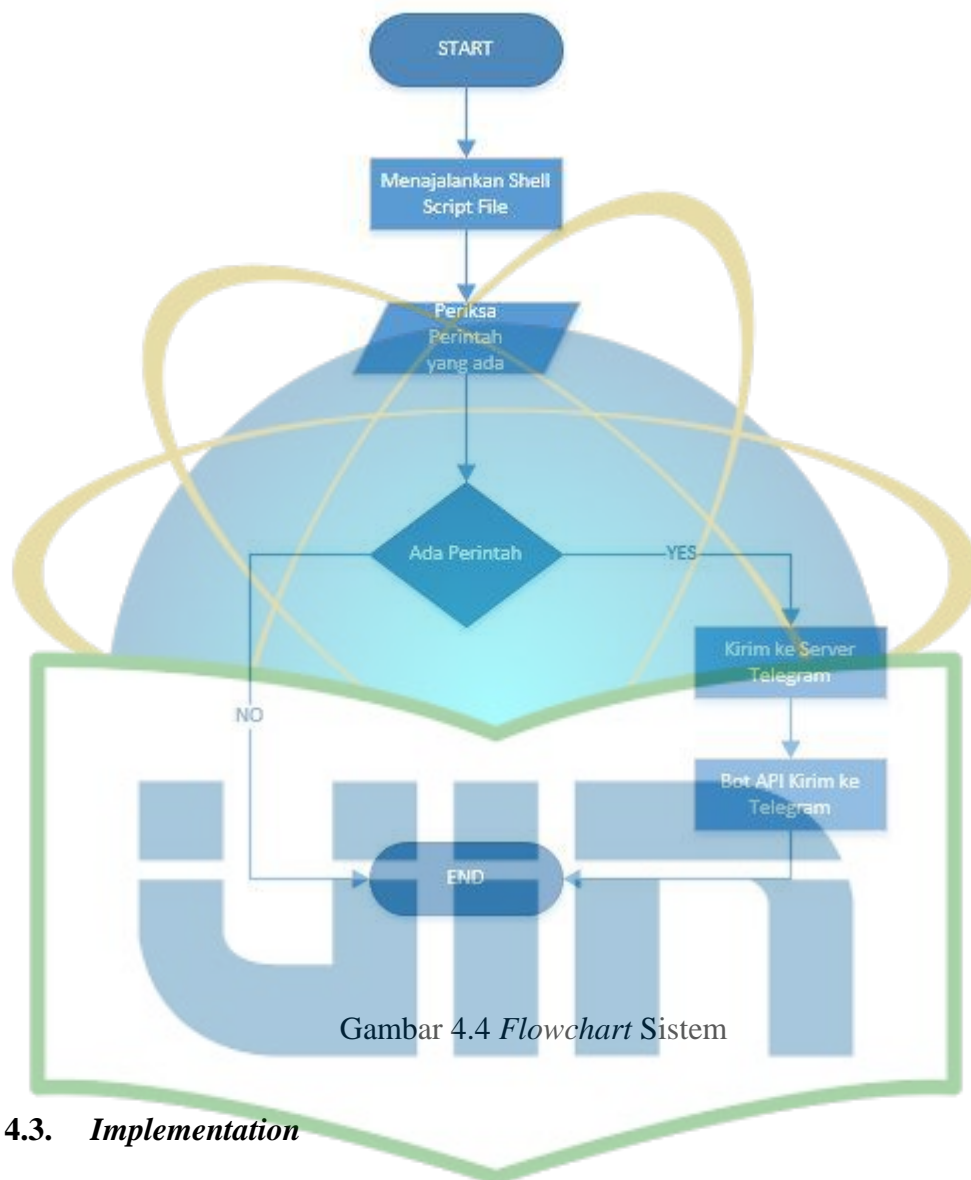
4.2.1. Perancangan Sistem

Setelah melengkapi semua kebutuhan baik analisis, *hardware* maupun *software*, selanjutnya adalah membuat perancangan sistem baru, yang akan dibangun dan diimplementasikan. Pada tahap ini penulis akan menjelaskan bagaimana jalannya sistem *monitoring server* ini bekerja.



Gambar 4.3 Perancangan Sistem

4.2.2. Flowchart Sistem



Gambar 4.4 Flowchart Sistem

4.3. Implementation

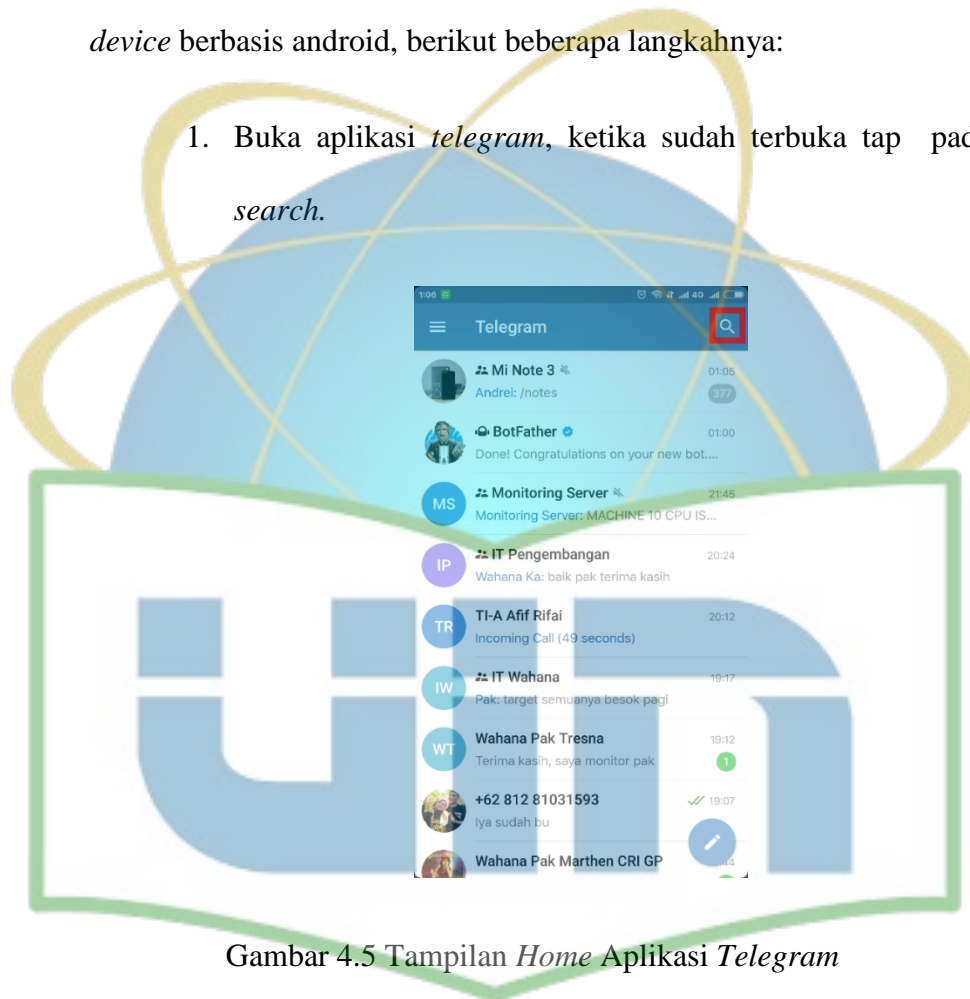
Pada tahap ini penulis akan melakukan penjelasan mengenai penerapan rancangan sistem *monitoring server* pada server PT. Wahana Prestasi Logistik. Mulai dari pembuatan *bot telegram* hingga konfigurasi pada sistem, lalu menghubungkan dengan media *telegram* untuk pemberitahuan ke *server*

administrator. Sehingga *server administrator* menerima pesan sesuai dengan percobaan.

4.3.1. Pembuatan *Bot Telegram*

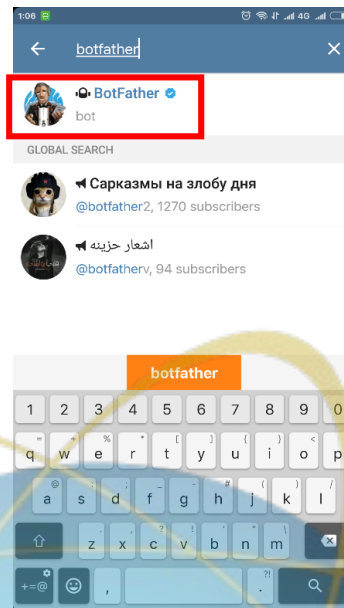
Pada tahapan pembuatan *bot telegram* penulis menggunakan *mobile device* berbasis android, berikut beberapa langkahnya:

1. Buka aplikasi *telegram*, ketika sudah terbuka tap pada *icon search*.

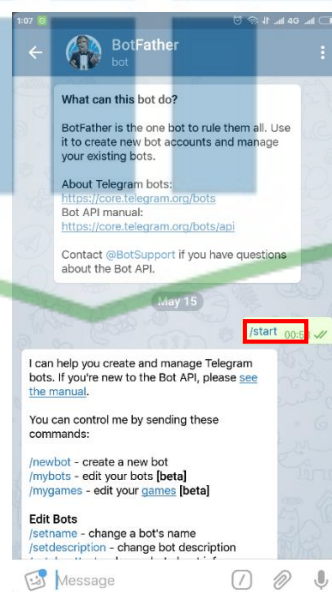


Gambar 4.5 Tampilan *Home* Aplikasi *Telegram*

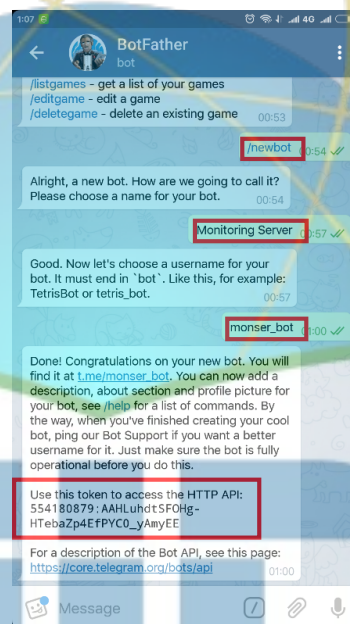
2. Setelah itu ketikkan *botfather*. *Botfather* adalah *bot* utama untuk membuat sebuah *bot* tetapi masih belum bisa dijalankan, maksudnya di *botfather* hanya bisa membuat *bot*-nya saja belum untuk membuat *bot* itu berjalan.

Gambar 4.6 Pencarian *BotFather*

3. Untuk memulai pembuatan *bot* ketik `/start`, dimana nantinya *botfather* akan memberikan info tentang langkah-langkah pembuatan *bot*.

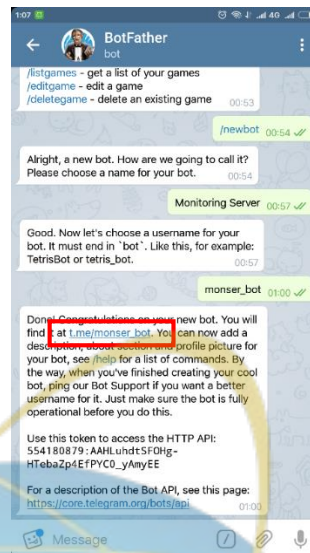
Gambar 4.7 Memulai Pembuatan *Bot* Baru

4. Setelah *botfather* memberikan info tentang bagaimana pembuatan *bot*, ketikan */newbot* tujuannya adalah untuk membuat *bot* baru, dalam hal ini penulis menggunakan nama *Monitoring Server* dan *username monser_bot*, untuk *username* harus diakhiri *_bot*. Pada tahap ini akan mendapatkan *token* untuk mengakses *HTTP API*.



Gambar 4.8 Pembuatan *Bot* Selesai

5. *Bot* yang sudah dibuat masih belum bisa digunakan agar bisa digunakan maka harus mengakses *url* yang diberikan oleh *botfather*.



Gambar 4.9 Akses *Url* Untuk Menggunakan *Bot* Baru

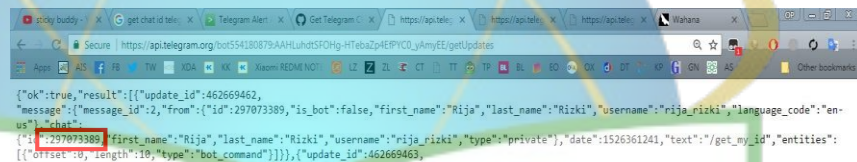
6. Setelah mengakses *url* yang diberikan oleh *botfather*. Tahap selanjutnya adalah memulai *chat* dengan *bot* yang telah dibuat tadi, dengan cara tap tulisan *start* dan *bot* sudah bisa digunakan.



Gambar 4.10 Memulai *Bot* Baru

7. Untuk tahap selanjutnya yaitu mendapatkan *chat_id bot* yang sudah dibuat agar pesan yang terkirim sesuai dengan *bot* yang dibuat, gunakan *web browser*, penulis menggunakan *chrome* untuk mengakses :

<https://api.telegram.org/bot<token>/getUpdates> ganti “<token>” dengan *token* yang didapat dari *botfather* yaitu 554180879:AAHLuhdtSFOHg-HTebaZp4EfPYC0_yAmyEE. Setelah mengakses *url* tersebut maka akan didapatkan *chat_id* yaitu “297073389”.



Gambar 4.11 *Chat_id*

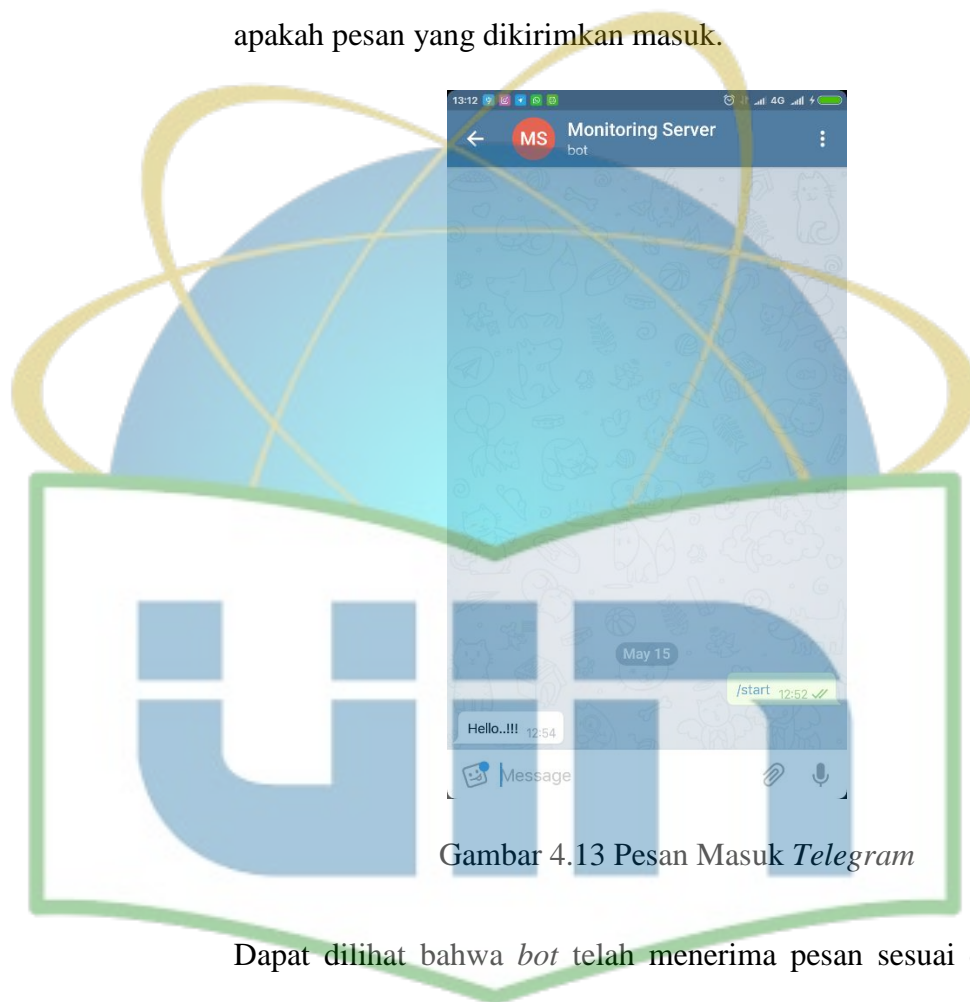
8. Tahap terakhir untuk pembuatan *bot telegram* yaitu dengan mencoba *chat_id* yang didapatkan untuk mengirimkan pesan ke *bot* dengan mengakses *url* :

https://api.telegram.org/bot554180879:AAHLuhdtSFOHg-HTebaZp4EfPYC0_yAmyEE/sendMessage?chat_id=297073389&text=Hello..!!! , adapun pesan yang akan dikirimkan yaitu Hello..!!!”.



Gambar 4.12 Pengiriman Pesan Via *Browser*

Setelah mengakses *url* tersebut lakukan pengecekan pada *bot* apakah pesan yang dikirimkan masuk.



Gambar 4.13 Pesan Masuk *Telegram*

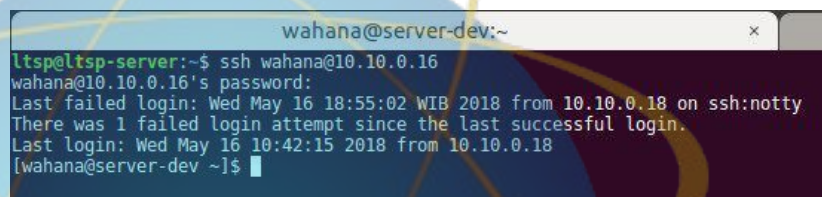
Dapat dilihat bahwa *bot* telah menerima pesan sesuai dengan yang dikirimkan, maka *bot* sudah bisa digunakan.

4.3.2. Konfigurasi Sistem *Monitoring Server*

Pada konfigurasi sistem *monitoring server* semua dilakukan pada file *monserver.conf* yang berada di direktori */home/wahana/monserver/config*.

Berikut akan dijelaskan tahapan-tahapan konfigurasi sistem monitoring server :

1. Tahap pertama yang harus dilakukan yaitu mengakses server via ssh, dimana penulis menggunakan *server* ke 2 yang memiliki *GUI* dengan *OS Ubuntu 16.04 LTS*. Buka *terminal* atau *command line* pada *server 2 (ltsp-server)*, kemudian lakukan *ssh* pada *server 1 (server-dev)*, lihat gambar.



```
wahana@server-dev:~
ltsp@ltsp-server:~$ ssh wahana@10.10.0.16
wahana@10.10.0.16's password:
Last failed login: Wed May 16 18:55:02 WIB 2018 from 10.10.0.18 on ssh:notty
There was 1 failed login attempt since the last successful login.
Last login: Wed May 16 10:42:15 2018 from 10.10.0.18
[wahana@server-dev ~]$
```

Gambar 4.14 SSH Dev Server

2. Masuk sebagai *root* dengan mengetikan *sudo su* dan masukan password untuk *user root*. Lakukan akses ke direktori wahana dengan mengetikan *cd /home/wahana*, kemudian buat 1 direktori dengan nama *monserver* dengan cara mengetikan *mkdir monserver* lalu tekan tombol *enter*, lihat apakah direktori sudah terbuat dengan mengetik *ls* lalu *enter*. Pada gambar dibawah terlihat bahwa direktori *monserver* sudah terbuat. Pada direktori ini nantinya semua *file* yang diperlukan untuk membangun *minitoring server* diletakkan.

```
[wahana@server-dev ~]$ sudo su
[sudo] password for wahana:
[root@server-dev wahana]# cd /home/wahana
[root@server-dev wahana]# mkdir monserver
[root@server-dev wahana]# ls
android-sdk_r24.2-linux.tgz  monserver  theeeye
[root@server-dev wahana]#
```

Gambar 4.15 Pembuatan Direktori *Monserver*

3. Setelah direktori *monserver* terbuat, akses direktori tersebut dengan mengetikkan *cd monserver* lalu *enter*, jika sudah dalam direktori *monserver* selanjutnya buat direktori dengan nama *conf* dimana pada direktori ini akan diletakkan file konfigurasi. Selanjutnya buat 1 file dengan nama *monserver.sh* dengan cara mengetikkan *vim monserver.sh* lalu *enter*. Lihat apakah direktori *conf* dan file *monserver.sh* sudah terbuat dengan cara ketik *ls* lalu *enter*.

```
[root@server-dev wahana]# cd monserver/
[root@server-dev monserver]# mkdir conf
[root@server-dev monserver]# vim monserver.sh
[root@server-dev monserver]# ls
conf  monserver.sh
[root@server-dev monserver]#
```

Gambar 4.16 Pembuatan File *Monserver.sh*

4. Setelah direktori *conf* dan file *monserver.sh* terbuat, ubah *permission file monserver.sh* agar bisa dieksekusi nantinya, yaitu dengan cara *chmod +x monserver.sh*.

```
[root@server-dev monserver]# ls
conf  monserver.sh
[root@server-dev monserver]# chmod +x monserver.sh
[root@server-dev monserver]#
```

Gambar 4.17 Merubah Hak Akses *Monserver.sh*

5. Setelah *permission* file *monserver.sh* diubah, edit file tersebut dan tambahkan aturan untuk mengetahui direktori program dan file konfigurasi program.

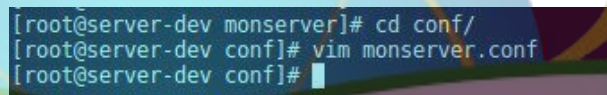


```

root@server-dev:/home/wahana/monserver
#!/bin/bash
cd /home/wahana/monserver/
#reading config file
source config/monserver.conf
  
```

Gambar 4.18 Direktori Program

6. Masuk ke direktori *conf* dan buat *file* dengan nama *monserver.cnf* caranya sama dengan membuat *file* *monserver.sh* yaitu dengan mengetik *vim monserver.conf* lalu enter.



```

[root@server-dev monserver]# cd conf/
[root@server-dev conf]# vim monserver.conf
[root@server-dev conf]#
  
```

Gambar 4.19 Pembuatan File Konfigurasi

7. Langkah selanjutnya yaitu membuat aturan-aturan pada *file monserver.conf* yang berada pada direktori *conf*. Berikut akan dijelaskan maksud dan kegunaan yang ada pada *file config* sistem *monitoring server*:

```

root@server-dev:/home/wahana/monserver
machine="DEV SERVER"
status="on"
sql="on"
httpd="on"
disk="on"
alive="on"
memory="on"
cpu="on"
token="554180879:AAHLuhdtSF0Hg-HTebaZp4EfPYC0_yAmyEE"
chat_id="297073389"
url="https://api.telegram.org/bot$token/sendMessage"
ip_servers=10.10.0.16,10.10.0.18
ip_server1=10.10.0.16
ip_server2=10.10.0.18
name_server1="DEV SERVER"
name_server2="LTSP SERVER"
message_sql="SQL IS DOWN!!!"
message_httpd="HTTPD IS DOWN!!!"
message_disk="DISK IS FULL"
message_alive="IS DOWN!!!"
message_cpu="CPU IS HIGH!!!"
message_memory="MEMORY IS HIGH!!!"
threshold_disk=95
threshold_cpu=80
threshold_memory=80
cpu_count=4

```

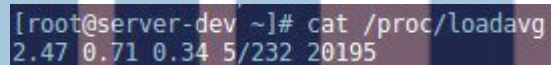
Gambar 4.20 Konfigurasi

- a. Untuk konfigurasi awal yaitu menentukan nama *server*, penulis menggunakan *machine* sebagai alias nama dan untuk nama *server* yaitu “DEV SERVER”.
- b. Jika status “on” maka program akan berjalan dan jika status “off” maka program tidak akan berjalan, ini juga berlaku untuk *sql*, *httpd*, *disk*, *alive*, *memory*, dan *cpu*.
- c. *Token*, *chat_id*, dan *url* didapatkan pada waktu pembuatan dan pengecekan *bot telegram*.
- d. *Ip_servers* yaitu *ip* yang digunakan pada masing-masing *server*. Dalam penelitian ini ada dua *server* yang digunakan dimana *ip_server1* yaitu 10.10.0.16 dengan *name_server1* yaitu “DEV SERVER” dan *ip_server2* yaitu 10.10.0.18 dengan *name_server2* yaitu “LTSP SERVER”. Hal ini diperuntukan mengetahui jika ada *server* yang mati.

- e. *Messages* digunakan untuk menerangkan masalah apa yang dialami oleh *server* yang nantinya akan dikirimkan ke *bot telegram*.
- f. *Threshold* atau yang disebut ambang batas, hal ini untuk menentukan berapa persen dari penggunaan baik *cpu*, *memory* ataupun *disks* yang diperbolehkan jika diatas yang sudah ditentukan berarti ada masalah pada *server*. *Cpu_count*, untuk masalah ini harus disesuaikan *core* yang tersedia pada *cpu server* pada penelitian ini server memiliki 4 *core* pada *cpu*-nya.

4.3.3. Konfigurasi Notifikasi Load CPU

Pada konfigurasi notifikasi *load cpu*, *counter* digunakan untuk mendapatkan berapa besar *load cpu*. Untuk mendapatkan besarnya *load cpu* yaitu dengan menjalankan perintah “`cat /proc/loadavg`”.



```
[root@server-dev ~]# cat /proc/loadavg
2.47 0.71 0.34 5/232 20195
```

Gambar 4.21 Load CPU

Dapat dilihat dari hasil ada beberapa nilai yang didapatkan, dan nilai yang digunakan nantinya untuk mendapatkan besarnya *load cpu* adalah nilai pada *column* 1 tanpa angka dibelakang koma. Untuk mendapatkan nilai tersebut maka dijalankan perintah tambahan dari perintah sebelumnya `cat /proc/loadavg | awk -F. '{print $1}'`.


```
[root@server-dev ~]# cat /proc/loadavg | awk -F. '{print $1}'
2
```

Gambar 4.22 Nilai *Load CPU*

Setelah menjalankan perintah tambahan maka didapatkan nilai untuk konfigurasi notifikasi *load cpu*. Kemudian untuk mengetahui berapa persen *cpu* yang terpakai penulis menggunakan *percent* sebagai parameternya. Dimana hasil dari *counter* dikalikan 100 dan dibagi jumlah *core cpu*. Jika hasil *percent* lebih tinggi dari ambang batas *cpu* yang sudah ditentukan pada file *monserver.conf* pada direktori */home/wahana/monserver/conf*, maka program akan mengirimkan pesan dengan menjalankan perintah *curl -s -X POST \$url -d chat_id=\$chat_id -d text="\$machine \$message_cpu = \$percent%"*

```
#check CPU
if [ "$cpu" == "on" ]; then
  counter=$(cat /proc/loadavg | awk -F. '{print $1}')
  let percent=counter*100/cpu_count
  if [[ $percent -gt $threshold_cpu ]]; then
    echo | curl -s -X POST $url -d chat_id=$chat_id -d text="$machine $message_cpu = $percent%"
  fi
fi
```

Gambar 4.23 Pengiriman Pesan *Load CPU*

4.3.4. Konfigurasi Notifikasi *Memory Usage*

Hal pertama untuk konfigurasi notifikasi memory adalah menjalankan perintah *free -m* pada terminal atau command line, ini berguna untuk mengecek jumlah total memory, berapa yang terpakai dan berapa yang free.


```
[root@server-dev ~]# free -m
```

	total	used	free	shared	buff/cache	available
Mem:	7723	2346	232	50	5143	5012
Swap:	7935	49	7886			

Gambar 4.24 *Memory Usage*

Setelah mendapatkan detail dari jumlah total memory, berapa yang terpakai, dan berapa yang free, selanjutnya menghitung berapa persen dari pemakaian memory dengan menjalankan perintah tambahan `free -m | awk 'NR==2{print $3*100/$2}' | awk -F. '{print $1}'`. Dimana nantinya hasil dari perintah tersebut di pakai oleh *counter*.

```
[root@server-dev ~]# free -m | awk 'NR==2{print $3*100/$2}' | awk -F. '{print $1}'
31
```

Gambar 4.25 *Nilai Memory Usage*

Jika hasil yang diperoleh lebih tinggi dari ambang batas *memory* yang sudah ditentukan, maka program akan mengirimkan pesan dengan menjalankan perintah `curl -s -X POST $url -d chat_id=$chat_id -d text="$machine $message_memory = $counter%"`

```
#check MEMORY
if [ "$memory" == "on" ]; then
    counter=$(free -m | awk 'NR==2{print $3*100/$2}' | awk -F. '{print $1}')
    if [ $(($counter -gt $sthreshold_memory)) ]; then
        echo | curl -s -X POST $url -d chat_id=$chat_id -d text="$machine $message_memory = $counter%"
    fi
fi
```

Gambar 4.26 *Pengiriman Pesan Memory Usage*

4.3.5. Konfigurasi Notifikasi *Disks Usage*

Jalankan perintah `df -H | grep -vE '^Filesystem|tmpfs|cdrom' | awk '{ print $5 " " $1 " " $6 }'`

pada terminal atau command line, untuk mendapatkan berapa persen penggunaan disk, filesystem, dan kemana disks dimount. Hasil tersebut didapatkan dari fungsi awk, dimana yang diambil yaitu column ke 5 (*used%*), column ke 1 (*filesystem*), dan column 6 (*mounted on*).

```
[root@server-dev ~]# df -H | grep -vE '^Filesystem|tmpfs|cdrom' | awk '{ print $5 " " $1 " " $6 }'
```

9%	/dev/mapper/centos-root	/
93%	/dev/sdb1	/data
19%	/dev/sda1	/boot

```
[root@server-dev ~]#
```

Gambar 4.27 *Disks Usage*

Parameter yang digunakan untuk konfigurasi ini yaitu *usep* (penggunaan dalam %), *partition*, dan *dir*. Jika *usep* melebihi dari ambang batas yang sudah ditetapkan maka program akan menjalankan perintah `curl -s -X POST $url -d chat_id=$chat_id -d text="$machine $message_disk [$partition $dir] = $usep%"` untuk mengirimkan notifikasi kepada server administrator.

```
#!/bin/bash
#check DISK
if [ "$disk" == "on" ]; then
df -H | grep -vE '^Filesystem|tmpfs|cdrom' | awk '{ print $5 " " $1 " " $6 }' | while read output;
do
usep=$(echo $output | awk '{ print $1 }' | cut -d'%' -f1)
partition=$(echo $output | awk '{ print $2 }' | cut -d'/' -f1)
dir=$(echo $output | awk '{ print $3 }')
if [ $(($usep -gt $threshold)) ]; then
echo | curl -s -X POST $url -d chat_id=$chat_id -d text="$machine $message_disk [$partition $dir] = $usep%"
fi
done
fi
```

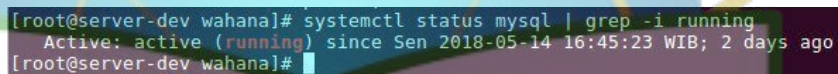
Gambar 4.28 Pengiriman Pesan *Disks Usage*

4.3.6. Konfigurasi Notifikasi *Service Mysql*

Pada konfigurasi notifikasi, apakah *service mysql* berjalan atau tidak dengan menggunakan *counter*, guna untuk mendapatkan nilai 1, jika nilai

lebih kecil dari 1, dengan artian *service mysql* tidak berjalan dan program akan mengirimkan *message* ke *bot telegram*.

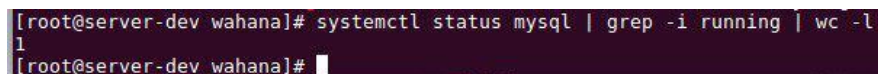
Untuk mengetahui *service mysql* berjalan pada *ubuntu* atau *linux* yaitu dengan menjalankan perintah pada *terminal* atau *command line* “*systemctl status mysql*”, sebelum menjalankan perintah tersebut harus masuk sebagai *root*. Pada konfigurasi hanya perlu mengetahui bahwa *service mysql* berjalan, jadi perintah yang dijalankan yaitu “*systemctl status mysql | grep -i running*” dimana *grep* mempunyai fungsi sebagai pencari *string* di dalam suatu *file*, dapat dilihat pada gambar bahwa *service* berjalan.



```
[root@server-dev wahana]# systemctl status mysql | grep -i running
Active: active (running) since Sen 2018-05-14 16:45:23 WIB; 2 days ago
[root@server-dev wahana]#
```

Gambar 4.29 Service Mysql

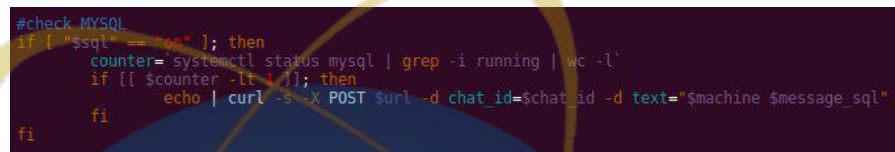
Selanjutnya untuk mendapatkan nilai 1 maka perintah yang dijalankan pada *terminal* atau *command line* yaitu “*systemctl status mysql | grep -i running | wc -l*” dimana *wc* (*word count*) berfungsi untuk menghitung kata yang ada. Diketahui bahwa kata “*running*” hanya ada 1 kata pada *output* perintah yang dijalankan.



```
[root@server-dev wahana]# systemctl status mysql | grep -i running | wc -l
1
[root@server-dev wahana]#
```

Gambar 4.30 Nilai Dari Service Mysql

Jika nilai tidak sama dengan 1 maka sistem akan mengirimkan pesan bahwa service mysql tidak berjalan, dengan menjalankan perintah `curl -s -X POST $url -d chat_id=$chat_id -d text="$machine $message_sql"`. Curl sendiri adalah program dan *library* untuk mengirim dan mengambil data melalui *URL*.

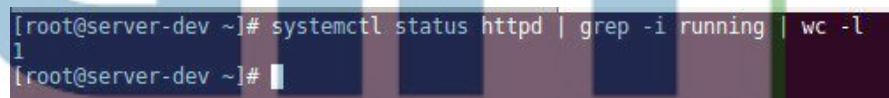


```
#check MYSQL
if [ "$sql" == "0" ]; then
  counter=$(systemctl status mysql | grep -i running | wc -l)
  if [[ $counter -lt 1 ]]; then
    echo | curl -s -X POST $url -d chat_id=$chat_id -d text="$machine $message_sql"
  fi
fi
```

Gambar 4.31 Pengiriman Pesan *Service Mysql*

4.3.7. Konfigurasi Notifikasi *Service HTTPD*

Sama halnya dengan konfigurasi untuk notifikasi *service mysql*. *Counter* digunakan untuk mendapatkan nilai 1 dari perintah yang dijalankan. Perbedaan pada konfigurasi yaitu dalam pengecekan *service*, pada konfigurasi ini yang dicek adalah *service httpd*.



```
[root@server-dev ~]# systemctl status httpd | grep -i running | wc -l
1
[root@server-dev ~]#
```

Gambar 4.32 Nilai Dari *Service HTTPD*

Jika nilai tidak sama dengan 1 maka sistem akan mengirimkan pesan bahwa *service httpd* tidak berjalan, dengan menjalankan perintah `curl -s -X POST $url -d chat_id=$chat_id -d text="$machine $message_httpd"`.

```
#check HTTPD
if [ "$httpd" == "on" ]; then
    counter=$(systemctl status httpd | grep -i running | wc -l)
    if [[ $counter -lt 1 ]]; then
        echo | curl -s -X POST $url -d chat_id=$chat_id -d text="$machine $message_httpd"
    fi
fi
```

Gambar 4.33 Pengiriman Pesan *Service HTTPD*

Pada file konfigurasi *monserver.conf* dijelaskan bahwa ada 2 *server* yang digunakan, dimana masing-masing *server* memiliki ip 10.10.0.16 dan 10.10.0.18, ip tersebut digunakan untuk mengetahui jika *server* dalam keadaan mati. Adapun caranya yaitu melakukan *ping* ke *ip server* yang ada pada file konfigurasi, dengan sintak `ping -c1 ip | grep -i error | wc -l`, dimana sintak tersebut digunakan *counter* untuk menghasilkan nilai 0, jadi jika nilai *counter* lebih besar dari 0 yang artinya salah satu ip server tidak bisa di *ping* dan ada *output error* maka sistem akan menjalankan perintah `curl -s -X POST $url -d chat_id=$chat_id -d text="$name_server1 $message_alive"` dan mengirimkan pesan ke *bot telegram* yang akan diteruskan ke *server admin*.

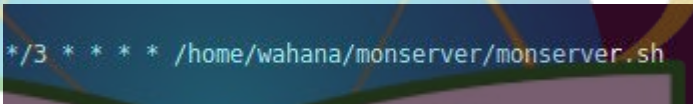
```
#check SERVER ON/OFF
if [ "$alive" == "on" ]; then
    for i in $(echo $ip_servers | tr ',' '\n'); do
        counter=$(ping -c1 $i | grep -i error | wc -l)
        if [[ $counter -gt 0 ]]; then
            case $i in
                $ip_server1)
                    echo | curl -s -X POST $url -d chat_id=$chat_id -d text="$name_server1 $message_alive"
                    ;;
                $ip_server2)
                    echo | curl -s -X POST $url -d chat_id=$chat_id -d text="$name_server2 $message_alive"
                    ;;
                *)
                    ;;
            esac
        fi
    done
fi
```

Gambar 4.34 Pengiriman Pesan *Server Down*

4.3.8. Konfigurasi *Crontab*

Crontab digunakan untuk mengatur penjadwalan menjalankan perintah atau *script* yang ada pada file *monserver.sh* yang sudah di konfigurasi untuk mengirimkan pesan ke *server administrator* jika ada masalah pada *server*.

Pada penelitian *crontab* diseting agar berjalan 3 menit sekali, adapun cara setingnya yaitu buka *terminal* atau *command line*, masuk sebagai *root* kemudian ketikkan `crontab -e` kemudian ketikkan `*/3 * * * *` `/home/wahana/monserver/monserver.sh` dan simpan.



```
*/3 * * * * /home/wahana/monserver/monserver.sh
```

Gambar 4.35 Konfigurasi *Crontab*

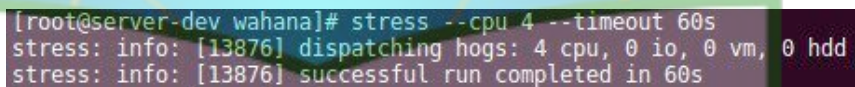
BAB V

HASIL DAN PEMBAHASAN

5.1. *Enforcement* (Pengujian)

5.1.1. Pengujian *Load CPU*

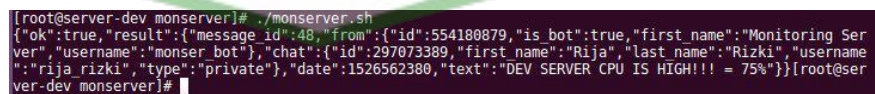
Untuk melakukan pengujian *load cpu*, penulis akan menggunakan fitur aplikasi *stress test*, jadi aplikasi stress test harus sudah terinstall pada *server*. Kemudian buka *terminal* atau *command line* ketik pada terminal `stress --cpu 4 --timeout 60s`, pengujian dilakukan selama 1 menit. Pada pengujian load cpu ambang batas diset menjadi 50.



```
[root@server-dev wahana]# stress --cpu 4 --timeout 60s
stress: info: [13876] dispatching hogs: 4 cpu, 0 io, 0 vm, 0 hdd
stress: info: [13876] successful run completed in 60s
```

Gambar 5.1 Pengujian *Load CPU*

Ketika stress test dijalankan buka satu lagi *terminal* atau *command line* masuk sebagai *root* lalu akses direktori dimana file *monserver.sh* berada yaitu `/home/wahana/monserver` dan jalankan file *monserver.sh*.



```
[root@server-dev monserver]# ./monserver.sh
{"ok":true,"result":{"message_id":48,"from":{"id":554180879,"is_bot":true,"first_name":"Monitoring Server","username":"monserver_bot"},"chat":{"id":297073389,"first_name":"Rija","last_name":"Rizki","username":"rija_rizki","type":"private"},"date":1526562380,"text":"DEV SERVER CPU IS HIGH!!! = 75%"}[root@server-dev monserver]#
```

Gambar 5.2 Pesan *Load CPU* Pada *Terminal*

Ada informasi yang diberikan oleh terminal ketika menjalankan file *monserver.sh*, “DEV SERVER CPU IS HIGH!!! = 75%”, selanjutnya

melakukan pengecekan pada aplikasi telegram apakah pesan yang dikirim sama dengan yang ada di terminal.



Gambar 5.3 Pesan *Load CPU* Pada *Telegram*

Dari hasil pengecekan pada aplikasi telegram diketahui bahwa pesan yang masuk sama dengan pesan yang ada pada terminal, yang artinya pengujian load cpu telah berhasil.

5.1.2. Pengujian *Memory Usage*

Sama halnya dengan pengujian *load cpu*, ambang batas *memory* juga di set menjadi 50 dan diuji menggunakan aplikasi *stress test*. Langkahnya juga hampir sama pada waktu pengujian *load cpu*, yaitu buka *terminal* atau *command line* lalu ketikkan `stress --vm 8 --vm-bytes 1G --timeout 60s`, pengujian dilakukan selama 1 menit.

```
[root@server-dev wahana]# stress --vm 8 --vm-bytes 1G --timeout 60s
stress: info: [16201] dispatching hogs: 0 cpu, 0 io, 8 vm, 0 hdd
stress: info: [16201] successful run completed in 60s
```

Gambar 5.4 Pengujian *Memory Usage*

Ketika pengujian *stress test* berjalan, buka *terminal* atau *command* line lainnya, lalu akses direktori dimana file *monserver.sh* berada dan jalankan file tersebut, sebelumnya masuk sebagai *root*.

```
[root@server-dev monserver]# ./monserver.sh
{"ok":true,"result":{"message_id":61,"from":{"id":554180879,"is_bot":true,"first_name":"Monitoring Server","username":"monserver_bot"},"chat":{"id":297073389,"first_name":"Riza","last_name":"Rizki","username":"riza_rizki","type":"private"},"date":1526563320,"text":"DEV SERVER MEMORY IS HIGH!!! = 67%"}}
```

Gambar 5.5 Pesan *Memory Usage* Pada *Terminal*

Ada informasi yang diberikan oleh terminal ketika menjalankan file *monserver.sh*, “DEV SERVER MEMORY HIGH!!! = 67 %”, selanjutnya melakukan pengecekan pada aplikasi *telegram* apakah pesan yang dikirim sama dengan yang ada di *terminal*.

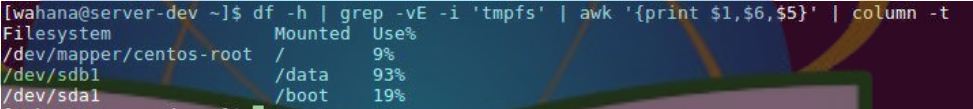


Gambar 5.6 Pesan *Memory Usage* Pada *Telegram*

Dari hasil pengecekan pada aplikasi *telegram* diketahui bahwa pesan yang masuk sama dengan pesan yang ada pada *terminal*, yang artinya pengujian *memory usage* telah berhasil.

5.1.3. Pengujian *Disks Usage*


Pada pengujian *disks usage* langkah pertama yang dilakukan adalah pengecekan pemakaian *disks* dengan menjalankan perintah pada terminal `df -h | grep -vE -i 'tmpfs' | awk '{print $1,$6,$5}' | column -t`. Adapun yang akan diuji yaitu `/dev/sdb1` yang mana pemakaiannya sudah mencapai 93%



```
[wahana@server-dev ~]$ df -h | grep -vE -i 'tmpfs' | awk '{print $1,$6,$5}' | column -t
Filesystem      Mounted      Use%
/dev/mapper/centos-root /             9%
/dev/sdb1       /data        93%
/dev/sda1       /boot        19%
```

Gambar 5.7 Pengecekan *Disks Usage*

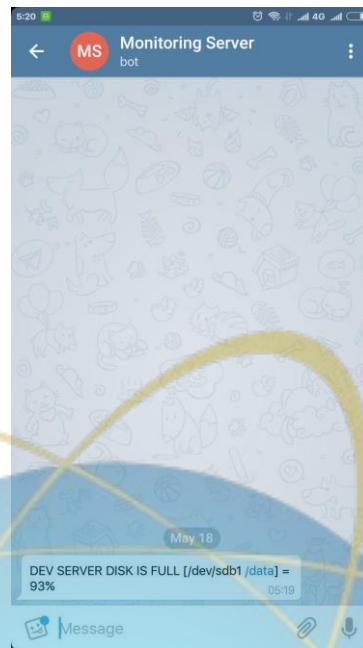
Selanjutnya edit konfigurasi pada file *monserver.conf* dan seting untuk ambang batas disks menjadi 90. Tahap berikutnya jalankan file *monserver.sh*



```
[root@server-dev monserver]# ./monserver.sh
{"ok":true,"result":{"message_id":85,"from":{"id":554180879,"is_bot":true,"first_name":"Monitoring Server",
"username":"monser_bot"},"chat":{"id":297073389,"first_name":"Rija","last_name":"Rizki","username":"rija_rizki",
"type":"private"},"date":1526598751,"text":"DEV SERVER DISK IS FULL [/dev/sdb1 /data] = 93%","entities
```

Gambar 5.8 Pesan *Disks Usage* Pada *Terminal*

Setelah menjalankan file *monserver.sh* didapatkan info pada *terminal* bahwa ada pesan yang dikirimkan ke *telegram* "DEV SERVER DISK IS FULL [/dev/sdb1 /data] = 93%". Selanjutnya pengecekan pada aplikasi *telegram*.



Gambar 5.9 Pesan *Disks Usage* Pada *Telegram*

Dari hasil pengecekan pada aplikasi *telegram* diketahui bahwa pesan yang masuk sama dengan pesan yang ada pada *terminal*, yang artinya pengujian *disks usage* telah berhasil.

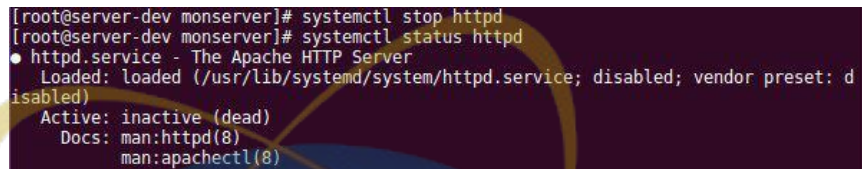
5.1.4. Pengujian *Service HTTPD*

Tahap awal untuk pengujian ini yaitu melakukan pengecekan service *HTTPD* dengan cara menjalankan perintah `systemctl status httpd`

```
[root@server-dev monserver]# systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: d
 isabled)
   Active: active (running) since Sab 2018-05-12 10:38:00 WIB; 6 days ago
     Docs: man:httpd(8)
           man:apachectl(8)
  Process: 3488 ExecReload=/usr/sbin/httpd $OPTIONS -k graceful (code=exited, status
 =0/SUCCESS)
```

Gambar 5.10 Pengecekan *Service HTTPD Running*

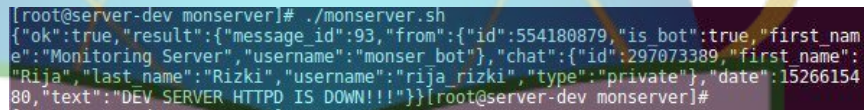
Jika status *service httpd running* maka jalankan perintah `systemctl stop httpd` untuk mematikan *servicenya*, kemudian lakukan pengecekan kembali dengan perintah `systemctl status httpd` untuk memastikan bahwa *service* sudah berhenti.



```
[root@server-dev monserver]# systemctl stop httpd
[root@server-dev monserver]# systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: d
  Active: inactive (dead)
     Docs: man:httpd(8)
          man:apachectl(8)
```

Gambar 5.11 Pengecekan *Service HTTPD Dead*

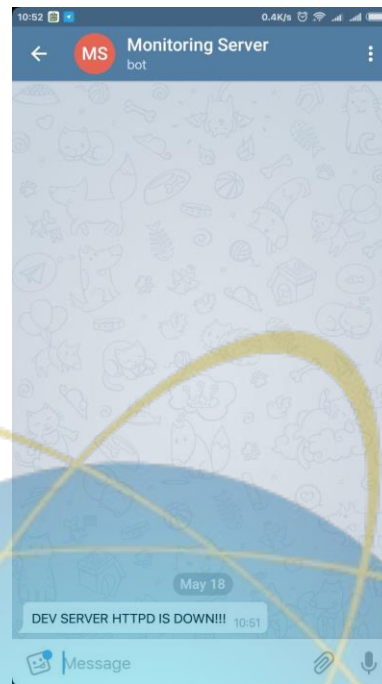
Selanjutnya menjalankan file *monserver.sh*



```
[root@server-dev monserver]# ./monserver.sh
{"ok":true,"result":{"message_id":93,"from":{"id":554180879,"is_bot":true,"first_nam
e":"Monitoring Server","username":"monser_bot"},"chat":{"id":297073389,"first name":
"Rija","last_name":"Rizki","username":"rija_rizki","type":"private"},"date":15266154
80,"text":"DEV SERVER HTTPD IS DOWN!!!"}}[root@server-dev monserver]#
```

Gambar 5.12 Pesan *Service HTTPD* Pada *Terminal*

Setelah menjalankan file *monserver.sh* didapatkan info pada *terminal* bahwa ada pesan yang dikirimkan ke telegram "DEV SERVER HTTPD IS DOWN!!!". Selanjutnya pengecekan pada aplikasi *telegram*.



Gambar 5.13 Pesan *Service HTTPD* Pada *Telegram*

Dari hasil pengecekan pada aplikasi *telegram* diketahui bahwa pesan yang masuk sama dengan pesan yang ada pada *terminal*, yang artinya pengujian *service httpd* telah berhasil.

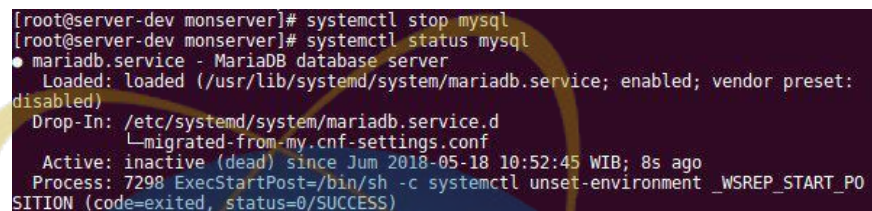
5.1.5. Pengujian *Service Mysql*

Tahap awal untuk pengujian ini yaitu melakukan pengecekan *service mysql* dengan cara menjalankan perintah `systemctl status mysql`.

```
[root@server-dev monserver]# systemctl status mysql
● mariadb.service - MariaDB database server
  Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; vendor preset: disabled)
  Drop-In: /etc/systemd/system/mariadb.service.d
           └─migrated-from-my.cnf-settings.conf
  Active: active (running) since Sen 2018-05-14 16:45:23 WIB; 3 days ago
  Process: 7298 ExecStartPost=/bin/sh -c systemctl unset-environment _WSREP_START_POSITION (code=exited, status=0/SUCCESS)
```

Gambar 5.14 Pengecekan *Service Mysql Running*

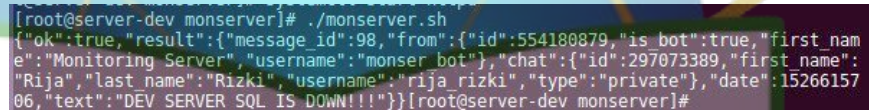
Jika status *service mysql running*, maka jalankan perintah `systemctl stop mysql` untuk mematikan *servicenya*, kemudian lakukan pengecekan kembali dengan perintah `systemctl status mysql` untuk memastikan bahwa *service* sudah berhenti.



```
[root@server-dev monserver]# systemctl stop mysql
[root@server-dev monserver]# systemctl status mysql
● mariadb.service - MariaDB database server
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; vendor preset: disabled)
   Drop-In: /etc/systemd/system/mariadb.service.d
            └─migrated-from-my.cnf-settings.conf
   Active: inactive (dead) since Jum 2018-05-18 10:52:45 WIB; 8s ago
   Process: 7298 ExecStartPost=/bin/sh -c systemctl unset-environment _WSREP_START_PO
   SITION (code=exited, status=0/SUCCESS)
```

Gambar 5.15 Pengecekan *Service Mysql Dead*

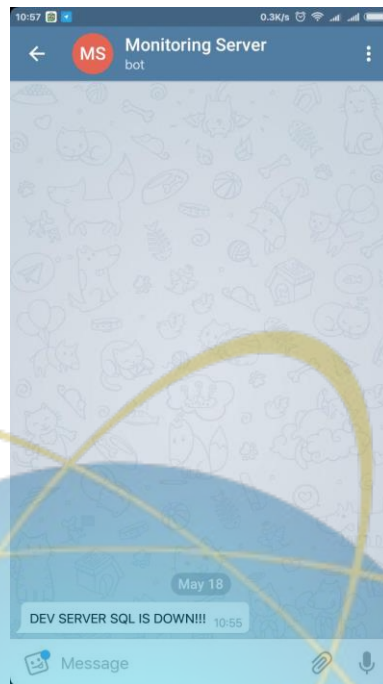
Selanjutnya menjalankan file *monserver.sh*



```
[root@server-dev monserver]# ./monserver.sh
{"ok":true,"result":{"message_id":98,"from":{"id":554180879,"is_bot":true,"first_name":"Monitoring Server","username":"monser_bot"},"chat":{"id":297073389,"first_name":"Rija","last_name":"Rizki","username":"rija_rizki","type":"private"},"date":1526615706,"text":"DEV SERVER SQL IS DOWN!!!"}}
```

Gambar 5.16 Pesan *Service Mysql* Pada *Terminal*

Setelah menjalankan file *monserver.sh* didapatkan info pada *terminal* bahwa ada pesan yang dikirimkan ke *telegram* "DEV SERVER MYSQL IS DOWN!!!". Selanjutnya pengecekan pada aplikasi *telegram*.

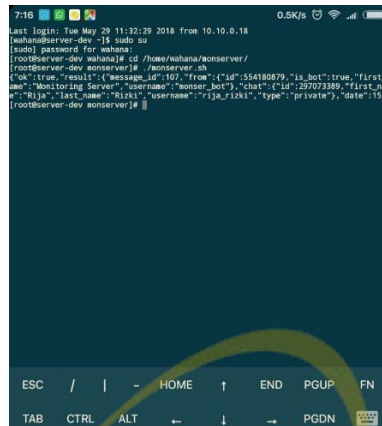


Gambar 5.17 Pesan *Service Mysql* Pada *Telegram*

Dari hasil pengecekan pada aplikasi *telegram* diketahui bahwa pesan yang masuk sama dengan pesan yang ada pada *terminal*, yang artinya pengujian *service mysql* telah berhasil.

5.1.6. Pengujian *Server Down*

Tahapan awal untuk pengujian *server down* yaitu dengan mematikan salah satu *server*, *LTSP Server* yang akan dimatikan. Setelah *server* mati selanjutnya mengakses *dev server* via *ssh* dengan aplikasi *juice ssh*.



```

7:16 0.5K/s
Last login: Tue May 29 11:32:29 2018 from 10.10.0.18
[wahanaServer-dev ~]$ sudo su
[sudo] password for wahana:
[root@server-dev wahana]# cd /home/wahana/monserver/
[root@server-dev monserver]# ./monserver.sh
{"ok":true,"result":{"message_id":107,"from":{"id":554180879,"is_bot":true,"first_name":"Monitoring Server","username":"monserver_bot","chat":{"id":237073389,"first_name":"Rizka","last_name":"Rizki","username":"rizka_rizki","type":"private"},"date":1527700000}}
[root@server-dev monserver]#

```

Gambar 5.18 Pesan *Server Down* Pada *Terminal*

Selanjutnya akses direktori dimana file *monserver.sh* berada `cd /home/wahana/monserver` lalu jalankan file *monserver.sh*. Setelah menjalankan file *monserver.sh*, dapat dilihat bahwa ada info jika *LTSP Server down*, begitu juga dengan aplikasi instagram yang telah mendapatkan pesan dari *bot telegram* bahwa *LTSP Server down*.



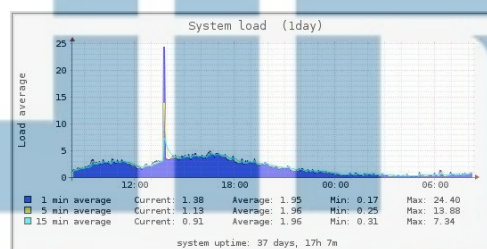
Gambar 5.19 Pesan *Server Down* Pada *Telegram*

5.2. *Enhancement* (Evaluasi)

Fase selanjutnya pada metode pengembangan sistem *SPDLC* adalah *Enhancement*. Fase ini meliputi aktivitas perbaikan dan evaluasi terhadap sistem yang telah dibangun. Dari hasil pengujian sistem monitoring *server* dengan media notifikasi *telegram* pada *server* yang dimiliki PT Wahana Prestasi Logistik berjalan dengan baik dan langsung mengirimkan notifikasi ke *telegram* jika terjadi masalah atau gangguan pada *server*, hal ini sesuai dengan rancangan yang penulis buat. Dengan adanya sistem ini memungkinkan bisa membantu *server administrator* di PT Wahana Prestasi Logistik bisa bertindak lebih cepat untuk menangani masalah yang ada.

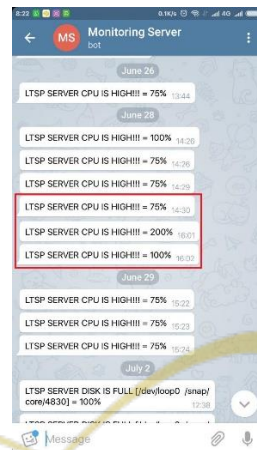
Berikut akan dipaparkan output atau hasil dari sistem yang berjalan dengan sistem yang penulis usulkan :

5.2.1. Pemantauan *Load CPU*



Gambar 5.20 Pemantauan *Load CPU* Sistem Berjalan

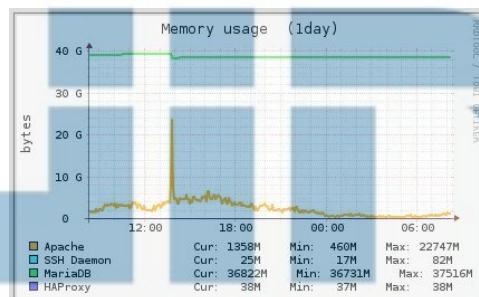
Pada gambar 5.20 dapat dilihat *output* dari sistem berjalan menampilkan *output*-an berupa grafik, dimana ada grafik yang meningkat secara signifikan yang menandakan bahwa *load cpu* mengalami kenaikan, *server administrator* mengetahui hal tersebut jika hanya berada di ruang *monitoring server* dan melihat pada layar monitoring.



Gambar 5.21 Notifikasi *Load CPU* Pada *Telegram*

Pada gambar 5.21 menunjukkan notifikasi yang dikirim oleh bot telegram ke aplikasi telegram setelah terjadinya kenaikan *load cpu* pada server. *Server administrator* dengan cepat mengetahui jika *load cpu* sedang naik setelah menerima notifikasi tanpa harus berada di ruang monitoring server.

5.2.2. Pemantauan *Memory Usage*



Gambar 5.22 Pemantauan *Memory Usage* Sistem Berjalan

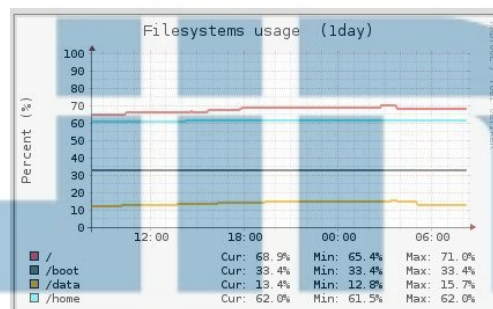
Pada gambar 5.22 dapat dilihat *output* dari sistem berjalan menampilkan output-an berupa grafik, dimana ada grafik yang meningkat secara signifikan yang menandakan bahwa memory usage mengalami kenaikan, pada outputan sistem usulan, bisa dilihat juga *memory* terpakai oleh *service* apa saja, seperti *apache*, *ssh*, *mariadb*, dan *haproxy*.



Gambar 5.23 Notifikasi *Memory Usage* Pada *Telegram*

Pada gambar 5.23 menunjukkan notifikasi yang dikirim oleh *bot telegram* ke aplikasi *telegram* setelah terjadinya kenaikan *memory usage* pada *server*. *Server administrator* dengan cepat mengetahui jika *memory* sedang naik setelah menerima notifikasi tanpa harus berada di ruang *monitoring server*.

5.2.3. Pemantauan *Disks Usage*



Gambar 5.24 Pemantauan *Disk Usage* Sistem Berjalan

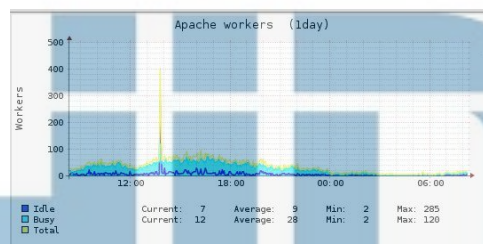
Pada gambar 5.24 dapat dilihat output dari sistem berjalan menampilkan outputan berupa grafik, dimana ada beberapa grafik yang ditampilkan yaitu direktori */(root)*, */boot*, */data*, */home*. Ketika *server administrator* ingin mengetahui berapa persen penggunaan disk maka *server administrator* harus melihat layar monitoring.



Gambar 5.25 Notifikasi *Disks Usage* Pada *Telegram*

Pada gambar 5.25 menunjukkan notifikasi yang dikirim oleh *bot telegram* ke aplikasi *telegram* jika salah satu direktori pada server sudah melewati ambang batas yang sudah ditentukan pada file konfigurasi, dan tugas *server administartor* memindahkan file yang ada di direktori tersebut ke direktori *backup*-an jika tersedia.

5.2.4. Pemantauan *Service HTTPD*



Gambar 5.26 Pemantauan *Service HTTPD* Sistem Berjalan

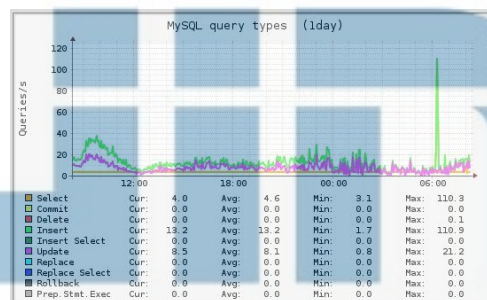
Pada gambar 5.26 pemantauan *service httpd* hanya berupa *apache worker*. Ketika *service httpd* tidak berjalan maka *server administrator* tidak mengetahui hal tersebut. *Server administrator* baru mengetahui jika aplikasi *web* tidak berjalan.



Gambar 5.27 Notifikasi *Service HTTPD* Pada *Telegram*

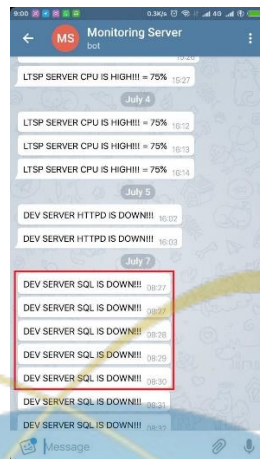
Pada gambar 5.27 menunjukkan notifikasi yang dikirim oleh *bot telegram* ke aplikasi *telegram* jika *service httpd* pada server tidak aktif yang mengakibatkan aplikasi *web* tidak dapat diakses. Hal ini menjadi salah satu kelebihan dari sistem yang dibuat untuk memonitoring server.

5.2.5. Pemantauan *Service Mysql*



Gambar 5.28 Pemantauan *Service Mysql* Sistem Berjalan

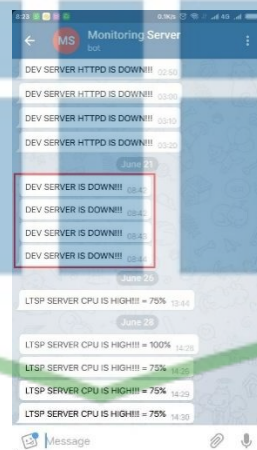
Pada gambar 5.28 pemantauan *service mysql* hanya berupa berupa proses-proses yang ada pada *mysql* seperti *select*, *commit*, *delet*, *insert*, *update* dsb. Ketika *service mysql* tidak berjalan maka *server administrator* tidak mengetahui hal tersebut. *Server administrator* baru mengetahui jika user tidak dapat *login* pada aplikasi *web* atau proses pada sistem yang berhubungan dengan *database*.



Gambar 5.29 Notifikasi *Service Mysql* Pada *Telegram*

Pada gambar 5.29 menunjukkan notifikasi yang dikirim oleh *bot telegram* ke aplikasi *telegram* jika *service mysql* pada *server* tidak aktif, sedangkan pada sistem berjalan jika hal ini terjadi maka *server administrator* tidak mengetahuinya.

5.2.6. Pemantauan *Server Down*



Gambar 5.30 Notifikasi *Server Down* Pada *Telegram*

Pada gambar 5.30 menunjukkan notifikasi yang dikirim oleh *bot telegram* ke aplikasi *telegram* bahwa *server down* atau mati, sedangkan pada sistem berjalan untuk pemantauan *server down* tidak adanya fitur tersebut sehingga *server administrator* diharuskan

melakukan pengecekan fisik *server* secara langsung apakah server mati atau *service* sistem berjalan pada posisi *off*.



BAB VI

PENUTUP

6.1. Kesimpulan

Berdasarkan hasil analisis, perancangan, dan implementasi yang telah penulis lakukan maka didapatkan kesimpulan sebagai berikut :

1. Perancangan sistem *monitoring server* dengan menerapkan bahasa pemrograman *bash shell* melalui *telegram* sebagai media notifikasi ketika adanya masalah atau gangguan pada *server* PT. Wahana Prestasi Logistik berhasil dengan baik dan mampu mendeteksi adanya masalah di *server*.
2. Notifikasi *telegram* berjalan sesuai dengan rancangan dalam memberikan pesan kepada *server administrator* PT. Wahana Prestasi Logistik jika adanya masalah pada *server*. Hal ini membantu *server administrator* mengetahui masalah yang terjadi pada *server* ketika tidak sedang berada di ruang *monitoring server*.

6.2. Saran

Sebagai pengembangan dari penelitian yang telah penulis lakukan, diberikan beberapa saran sebagai berikut:

1. Membuat fungsi *request* ke *bot* untuk memberikan informasi tentang keadaan *server*, jadi tidak menunggu *bot* mengirimkan pesan ketika adanya problem yang terjadi pada *server*.

2. Pendeteksian masalah bisa lebih banyak dan dikembangkan tidak hanya sebatas masalah yang penulis lakukan.



DAFTAR PUSTAKA

- Abdurahman, M., Muhidin, S, A., & Somantri,A. (2011). *Dasar-Dasar Metode Statistika Untuk Penelitian*. Bandung:Pustaka Setia.
- Amnur, H., Defni., Prayama, D., & Agustin, F. (2014). *Perancangan dan Implementasi Network Monitoring Sistem Menggunakan Nagios dengan Email dan SMS Alert*. Poli Rekayasa Volume 10, Nomor 1, Oktober 2014 ISSN : 1858-3709.
- Asri, N, F., Hamzah, A., & Sholeh, M. (2014). *Nagios Untuk Monitoring Server Dengan Pengiriman Notifikasi Gangguan Server Menggunakan Email Dan Sms Gateway (Studi Kasus : Pt.Gamatechnondonesia –Yogyakarta)*. Jurnal JARKOM Vol. 1 No.2 Januari 2014.
- Aviana, P, S. (2012). *Penerapan Pengendalian Internal Dalam Sistem Informasi Akuntansi Berbasis Komputer*. Jurnal Ilmiah Mahasiswa Akuntansi Vol.1 No.4, 65-70.
- CentOS, "What is CentOS Linux?" <https://wiki.centos.org/?id=3> (diakses 20 Mei 2018).
- Codepolitan, (Nov 12, 2016) "Membuat Program Bash Pertama" <https://www.codepolitan.com/belajar-bash-mencoba-bash-untuk-pertama-kali-57bbca3c28e54-17341> (diakses 21 Mei 2018).
- Hamburger, E. "Why Telegram has become the hottest messaging app in the world," <http://www.theverge.com/2014/2/25/5445864/telegramMessenger-hottest-app-in-the-world,%0A%0A> (diakses 19 Mei 2018).
- Handoko, T, H. (1995). *Manajemen Personalia dan Sumber Daya Manusia*. Yogyakarta: BPFE.
- Heryanto, A., Hermansyah, A., & Nizar, M. (2017). *Sistem Monitoring Server Dan Perangkat Jaringan Pada Enterprise Resource Planning Fasilkom Unsri Menggunakan Protokol Icmp Dan Snmp*. Jurnal SISTEMASI, Volume 6, Nomor 3, September 2017 : 1 – 10. E-ISSN:2540-9717, ISSN:2302-8149.
- I Made. & Cokorda. (2012). *Perancangan Implementasi Konsep Routing Dan Virtual Private Network Antara Webserver Moodle Dan Webserver Drupal*. Jurnal Elektronik Ilmu Komputer, Jeliku Vol 1 No. 2, Universitas Udayana, November 2012.

- IT-JURNAL.COM, “Apa Yang di Maksud Dengan Server ?”
<https://www.it-jurnal.com/apa-yang-di-maksud-dengan-server/>
 (diakses 19 Mei 2018).
- IT-JURNAL.COM, “Pengertian RAM (Random Acces Memory)”
<https://www.it-jurnal.com/pengertian-ram-random-acces-memory/>
 (diakses 19 Mei 2018).
- Krismiaji. (2010). *Sistem Informasi Akuntansi* : Yogyakarta : UPP-STIM YKPN.
- Mulyadi. (2008). *Sistem Akuntansi*. Jakarta: Salemba Empat.
- Muslim, M, A. (2006). *Pengembangan Distro Ubuntu untuk Aplikasi Game Centre*. Jurnal Teknologi Informasi Dinamik Volume XI, No.1, Januari 2006 : 16-22 ISSN : 0854-9524.
- MySQL. (2012, 15 Oktober). *Why MySQL?* Available:
<http://www.mysql.com/why-mysql/> (diakses 20 Mei 2018).
- Purwadi., Erwansyah, K., & Ikhsan, M. *Arsitektur Komputer tentang Mekanisme Kerja Prosesor dalam Menjalankan Intruksi dan Interupsi pada Sistem Kerja Komputer*. Jurnal Ilmiah Saintikom ISSN : 1978-6603.
- Pinto, R, L. (2014). *Secure Instant Messaging, Master Thesis, Department of Computer Science and Engineering*. Frankfurt University.
- Raco, J, R. (2010). *Metode Penelitian Kualitatif Jenis, Karakteristik dan Keunggulannya*. Jakarta:Grasindo.
- Rahmatulloh, A., & Firmansyah, MSN. (2017). *Implementasi Load Balancing Web Server menggunakan Haproxy dan Sinkronisasi File pada Sistem Informasi Akademik Universitas Siliwangi*. Jurnal Teknologi Dan Sistem Informasi - Vol. 03 No. 02(2017) 241-248.
- Rasyid, B, A., Solikin & Sularsa, A. (2011). *Realisasi Monitoring Server Menggunakan Nagios Dengan Memanfaatkan Event Handler, Email Dan SMS Gateway*. Academia Politeknik Telkom, Edisi September 2011, Lembaga Penelitian Politeknik Telkom, Bandung.
- Sakti, B., Aziz, A., & Doewes, A. (2013). *Uji Kelayakan Implementasi SSH sebagai Pengaman FTP Server dengan Penetration Testing*. Jurnal Itsmart Vol 2. No 1. Juni 2013 ISSN : 2301-7201.
- Sutabri, T. (2012). *Konsep Sistem Informasi*. Yogyakarta : C.V. Andi Offset.

Syarif, H., K. (2007). *Perancangan & Implementasi Pengembangan Server Linux Sebagai Sarana Bantu Perkuliahan*. Jurnal Computech & Bisnis, Vol. 1, No. 2, Desember 2007, 74-83.

UBAYA, "Media Penyimpanan Data Solid State Drive (SSD)"
http://www.ubaya.ac.id/2014/content/articles_detail/219/Media-Penyimpanan-Data-Solid-State-Drive--SSD-.html
 (diakses 21 Mei 2018).

Ubuntu id, "Linux untuk umat manusia" <http://ubuntu-id.org/>
 (diakses 19 Mei 2018).

Ubuntu, "stress-ng"
<http://manpages.ubuntu.com/manpages/artful/man1/stress-ng.1.html>
 (diakses 21 Mei 2018).

Utomo,D., Sholeh, M., & Avorizano, A. (2017). *Membangun Sistem Mobile Monitoring Keamanan Web Aplikasi Menggunakan Suricata dan Bot Telegram Channel*. Seminar Nasional Teknoka. Vol. 2, 2017ISSN No. 2502-8782.

VIM, "Apakah Vim itu?" <https://www.vim.org/6k/features.in.txt>
 (diakses 21 Mei 2018).

Wahsheh, L, A. & Foss, J, A. (2008). *Security Policy Development: Towards a Life-Cycle and Logic-Based Verification Model*.USA.

Wikimedia Foundation, Inc. (2012, 6). *Hard Disk Drive* Available:
http://en.wikipedia.org/wiki/Hard_disk_drive
 (diakses 21 Mei 2018).

Yanto, J. (2016). *Implementasi Sistem Monitoring Server Menggunakan Nagios*. Seminar Nasional Telekomunikasi dan Informatika (SELISIK 2016). ISSN : 2503-2844.

LAMPIRAN

1. SK PEMBIMBING



**KEMENTERIAN AGAMA
UNIVERSITAS ISLAM NEGERI (UIN)
SYARIF HIDAYATULLAH JAKARTA
FAKULTAS SAINS DAN TEKNOLOGI**

Jl. Ir. H. Juanda No. 95 Ciputat 15412 Indonesia
Telp.: (62-21) 7493606, 7493547, 7401925 Fax.: (62-21) 7493315

Email : fst@uinjkt.ac.id
Website : fst.uinjkt.ac.id

Nomor : B- 3049 / F9 / KM.01 / 05 / 2018
Lampiran : -
Perihal : Bimbingan Skripsi

Jakarta, 05 Juni 2018

Kepada Yth.

1. Siti Ummi Masruroh, M.Sc
2. Hendra Bayu Suseno, M.Kom

Assalamu'alaikum Wr.Wb.

Dengan ini diharapkan kesediaan Saudara untuk menjadi pembimbing I/II/ (Materi/Teknis)* penulisan skripsi mahasiswa:

Nama : Riya Rizki
NIM : 1111091000010
Program Studi : Teknik Informatika
Judul Skripsi : **“Perancangan Sistem Monitoring Server Dengan Menggunakan Bot Telegram Sebagai Media Notifikasi Alert**

Judul tersebut telah disetujui oleh Program Studi bersangkutan pada tanggal 4 Januari 2018 dengan outline, abstraksi dan daftar pustaka terlampir. Bimbingan skripsi ini diharapkan selesai dalam waktu 6 (enam) bulan setelah ditandatangani surat penunjukan pembimbing skripsi.

Apabila terjadi perubahan terkait dengan skripsi tersebut selama proses pembimbingan, harap segera melaporkan kepada Program Studi bersangkutan.

Demikian atas kesediaan Saudara, kami ucapkan terima kasih.

Wassalamu'alaikum Wr.Wb.

Wadek Bidang Akademik

Dr. Ir. Elpawati, MP

NIP: 19641204 199203 2 001

Tembusan:
Dekan (sebagai laporan)

2. SK PENELITIAN



PT. Wahana Prestasi Logistik
Jl. Rempoa Raya No. 88 Rempoa Ciputat 15412, Tangerang- Indonesia
Telp. (021) 7341688 (Hunting) Fax. (021) 73886974

No : 00011/IT/VI/2018
Lamp : -
Perihal : Pemberitahuan

Yang bertanda tangan dibawah ini:

Nama : Widi Raspito
NIK : 11180550
Jabatan : Manager IT

Menyatakan bahwa

Nama : Rija Rizki
NIM : 1111091000010
Instansi : Fakultas Sains Dan Teknologi UIN Syarif Hidayatullah
Jakarta

Telah mengadakan penelitian di PT Wahana Prestasi Logistik sejak tanggal 9 April 2018 s.d 2 Juni 2018 dengan judul:

“Perancangan Sistem Monitoring Server Dengan Menggunakan Bot Telegram Sebagai Media Notifikasi Alert”

Demikian surat keterangan ini kami sampaikan agar dapat dipergunakan sebagaimana mestinya.

Manager IT

Widi Raspito
NIK : 11180550

3. WAWANCARA

HASIL WAWANCARA

Responden : Widi Raspito
 Jabatan : Manager IT PT Wahana Prestasi Logistik
 Tanggal : 17 Mei 2018
 Lokasi : PT Wahana Prestasi Logistik, Rempoa, Ciputat Timur

1. P : Bagaimana sistem yang berjalan saat ini untuk memantau atau memonitor server yang dimiliki PT Wahana Prestasi Logistik ?

J : Saat ini untuk memantau server yang dimiliki PT Wahana Prestasi Logistik menggunakan aplikasi *open source* yaitu *monitorix*.

2. P : Apakah seorang *server administrator* akan mendapatkan notifikasi jika tidak sedang berada di tempat apabila terjadi masalah pada *server*?

J : Tidak, *server administrator* tidak mendapatkan notifikasi jika ada masalah pada *server*, sehingga *server administrator* harus terus *standby* di ruang monitoring.

3. P : Bagaimana seorang *server administrator* mengetahui jika terjadi masalah pada *server* saat tidak berada di tempat?

J : Saat ini jika terjadi masalah pada *server* maka *server administrator* akan dihubungi dan diinfokan oleh orang yang berada di kantor, dengan indikator jika mengakses aplikasi atau sistem lama atau tersendat.

4. P : Seberapa pentingkah peran *server* untuk perusahaan ini?

J : Sangat begitu penting karena setiap harinya ada hampir 40.000 - 45.000 barang milik *customer* atau pelanggan yang kesemua datanya disimpan dan diproses pada *server*.

5. P : Menurut anda bagaimana jika diterapkannya sistem *monitoring server* yang akan mengirimkan notifikasi ke *server administrator* jika ada masalah?

J : Hal tersebut sangat diharapkan, dan akan membantu *server administrator* dalam memonitoring *server*. Sehingga *server administrator* dapat dengan cepat mengatasi masalah yang ada.

6. P : Bagaimanakah menurut anda jika digunakannya aplikasi *telegram* sebagai media notifikasi jika ada masalah pada *server*?

J : Menarik menurut saya, karena saat ini media komunikasi sering menggunakan *instant messaging* dibandingkan dengan menggunakan *sms* atau *email*. *Telegram* juga salah satu aplikasi *instant messaging* yang sangat populer saat ini.



Responden


Widi Raspito