

**IMPLEMENTASI LOG EVENT MANAGEMENT WEB
APPLICATION FIREWALL MENGGUNAKAN
ELASTICSEARCH LOGSTASH
DAN KIBANA**

SKRIPSI

**Sebagai Syarat Dalam Menyelesaikan Program Studi Strata Satu (S-1)
Program Studi Teknik Informatika**



Disusun Oleh:

**NAMA : BUKHARI HASAN
NPM : 2017470020
JURUSAN : TEKNIK INFORMATIKA**

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH JAKARTA
2021**

UNIVERSITAS MUHAMMADIYAH JAKARTA
FAKULTAS TEKNIK – PRODI TEKNIK INFORMATIKA

LEMBAR PERSETUJUAN

**IMPLEMENTASI LOG EVENT MANAGEMENT WEB
APPLICATION FIREWALL MENGGUNAKAN
ELASTICSEARCH LOGSTASH
DAN KIBANA**

NAMA : BUKHARI HASAN
NPM : 2017470020
PROGRAM STUDI : TEKNIK INFORMATIKA

Skripsi ini telah disetujui pada tanggal, 28 Juli 2021

Oleh

Pembimbing



(Yana Adharani, M.Kom)

Mengetahui,

Ketua Program Studi Teknik Informatika

(Popy Meilina, M.Kom)

UNIVERSITAS MUHAMMADIYAH JAKARTA
FAKULTAS TEKNIK – PRODI TEKNIK INFORMATIKA

TANDA BUKTI PERSETUJUAN PEMBIMBING SKRIPSI

Pada Semester Genap

Tahun Akademik 2020/2021

Yang bertanda tangan di bawah ini Pembimbing skripsi menyatakan bahwa :

NAMA : BUKHARI HASAN
NPM : 2017470020

Judul Skripsi

**IMPLEMENTASI LOG EVENT MANAGEMENT WEB
APPLICATION FIREWALL MENGGUNAKAN
ELASTICSEARCH LOGSHTASH
DAN KIBANA**

Dimulai bulan, tahun : Maret, 2021

Selesai bulan, tahun : Juli, 2021

Untuk ikut serta Ujian Sidang Strata Satu (S1) yang diselenggarakan oleh Program Studi Teknik Informatika Universitas Muhammadiyah Jakarta.

Jakarta, 28 Juli 2021

Pembimbing



(Yana Adharani, M.Kom)

UNIVERSITAS MUHAMMADIYAH JAKARTA
FAKULTAS TEKNIK – PRODI TEKNIK INFORMATIKA

LEMBAR PENGESAHAN

**IMPLEMENTASI LOG EVENT MANAGEMENT WEB
APPLICATION FIREWALL MENGGUNAKAN
ELASTICSEARCH LOGSTASH
DAN KIBANA**

NAMA : BUKHARI HASAN
NPM : 2017470020
PROGRAM STUDI : TEKNIK INFORMATIKA

Skripsi ini telah diuji pada tanggal, 4 Agustus 2021

Oleh Penguji,

1. Popy Meilina, M.Kom :



2. Jumail, M.Sc :



3. Yana Adharani, M.Kom :



UNIVERSITAS MUHAMMADIYAH JAKARTA
FAKULTAS TEKNIK – PRODI TEKNIK INFORMATIKA

LEMBAR PERNYATAAN

Bersama ini saya menyatakan bahwa isi yang terkandung dalam skripsi ini, dengan judul :

**IMPLEMENTASI LOG EVENT MANAGEMENT WEB APPLICATION
FIREWALL MENGGUNAKAN ELASTICSEARCH LOGSHTASH DAN
KIBANA**

Adalah murni merupakan hasil penelitian dan pemikiran saya sendiri.

Demikian pernyataan ini saya buat dan siap menerima konsekuensi apapun di masa yang akan datang apabila ternyata skripsi ini merupakan salinan ataupun contoh karya-karya yang telah dibuat / diterbitkan sebelum tanggal skripsi ini.

Jakarta, 28 Juli 2021

Penulis



(Bukhari Hasan)

UNIVERSITAS MUHAMMADIYAH JAKARTA
FAKULTAS TEKNIK – PRODI TEKNIK INFORMATIKA

**SURAT PERNYATAAN PERSETUJUAN PUBLIKASI KARYA
ILMIAH UNTUK KEPENTINGAN AKADEMIK**

Yang bertanda tangan di bawah ini, saya :

Nama : Bukhari Hasan
NPM : 2017470020
Program Studi : Teknik Informatika
Jenjang : Strata Satu (S1)
Jenis Karya : Skripsi

Demi pengembangan ilmu pengetahuan, dengan ini menyetujui untuk memberikan ijin kepada pihak Program Studi Teknik Informatika FT-UMJ **Hak Bebas Royalti Non-Eksklusif (*Non-Exclusive Royalty-Free Right*)** atas karya ilmiah saya yang berjudul : **IMPLEMENTASI LOG EVENT MANAGEMENT WEB APPLICATION FIREWALL MENGGUNAKAN ELASTICSEARCH LOGSTASH DAN KIBANA.**


Dengan **Hak Bebas Royalti Non-Eksklusif** ini pihak FT-UMJ berhak menyimpan, mengalih media atau bentuk-kan, mengelolanya dalam pangkalan data (*database*). Mendistribusikannya dan menampilkan atau mempublikasikannya di internet atau media lain untuk kepentingan akademis tanpa perlu meminta ijin dari Saya selama tetap mencantumkan nama kami sebagai penulis/pencipta karya ilmiah tersebut.

Saya bersedia untuk menanggung secara pribadi, tanpa melibatkan pihak FT-UMJ, segala bentuk tuntutan hukum yang timbul atas pelanggaran Hak Cipta dalam karya ilmiah saya ini.

Demikian pernyataan ini saya buat dengan sebenarnya.

Jakarta, 28 Juli 2021

Yang menyatakan,



(Bukhari Hasan)




UNIVERSITAS MUHAMMADIYAH JAKARTA
FAKULTAS TEKNIK – JURUSAN TEKNIK INFORMATIKA






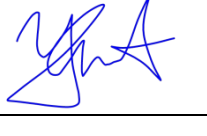
DAFTAR PRESENSI BIMBINGAN SKRIPSI

**IMPLEMENTASI LOG EVENT MANAGEMENT WEB
APPLICATION FIREWALL MENGGUNAKAN
ELASTICSEARCH LOGSTASH
DAN KIBANA**

NAMA : BUKHARI HASAN
NPM : 2017470020
JURUSAN : TEKNIK INFORMATIKA

Dosen Pembimbing : Yana Adharani,M.Kom

No	Tanggal	Catatan Dosen Pembimbing	Paraf
1	18-03-2021	BAB I : - Revisi Latar Belakang - Revisi Identifikasi Masalah - Revisi Metodologi Penelitian - Revisi Sistematika Penulisan	
2	27-03-2021	BAB I: - Revisi Sistematika Penulisan BAB II: - Tambahkan teori tentang topologi jaringan - Hapus teori tentang PHP dan MySQL	
3	14-04-2021	BAB I: - Revisi Sistematika Penulisan BAB II: - Lengkapi teori tentang topologi jaringan - Tambahkan teori tentang implementasi keamanan pada web server BAB III: - Tambahkan desain visualisasi log - Tambahkan pembuatan rules logstash - Tambahkan scenario ujicoba	

No	Tanggal	Catatan Dosen Pembimbing	Paraf
4	25-04-2021	BAB II: - Tambahkan kekurangan dari web server tanpa firewall BAB III: - Lengkapi rules logstash - Revisi desain visualisasi log	
5	27-04-2021	ACC BAB I, BAB II dan BAB III untuk Seminar Skripsi	
6	15-06-2021	BAB III : - Ganti kata Sistem Berjalan dan Sistem Usulan - Pindahkan proses penguraian Log ke BAB IV BAB IV : - Lanjutkan penulisan pada BAB IV	
7	12-07-2021	BAB III : - Berikan narasi awal pada BAB III BAB IV : - Tambahkan gambar popup notifikasi slack dan gambar log modsecurity. BAB V : - Sesuaikan kesimpulan dengan batasan dan rumusan masalah	
8	24-07-2021	- Pengecekan bab V	
9	28-07-2021	- Abstrak dan ACC akhir	

Dosen Pembimbing



(Yana Adharani, M.Kom)

ABSTRACT

Web-based application is one type of application that is widely used for various purposes such as internet banking, e-commerce to e-government. However, the widespread use of web-based applications is also directly proportional to the cyber attacks they face. In this study, the implementation of the Log Event Management Web Application Firewall aims to detect, log and block cyber attacks that are carried out as well as store, visualize logs and present notifications from these attacks. The implementation is carried out using 2 servers with Ubuntu Server 20.04 operating system and 1 system administrator computer with Ubuntu Desktop 20.04 operating system. The first server is used for web server with ModSecurity as Web Application Firewall, Apache HTTP Server as Web Server and Filebeat as log shipper. The second server is used as an ELK Stack server with Elasticsearch, Logstash and Kibana to store, parse and visualize logs from the web server and elasticsearch to send notifications. While on the computer system administrator used Slack to receive attack notifications. Testing is done by performing 3 types of cyber attacks on web applications, namely SQL Injection, Local File Inclusion and Cross Site Scripting. The results of these tests indicate that SQL Injection, Local File Inclusion and Cross Site Scripting attacks were successfully detected, logged and blocked by ModSecurity. In addition, the submitted logs were successfully stored, parsed and visualized on the ELK Stack server as well as attack notifications via the Slack channel in near real time.

Keyword : Log Event Management, Web Application Firewall, SQL Injection, Local File Inclusion, Cross Site Scripting.

ABSTRAK

Aplikasi berbasis web merupakan salah satu jenis aplikasi yang banyak digunakan untuk berbagai keperluan seperti internet *banking*, *e-commerce* hingga *e-government*. Namun maraknya penggunaan aplikasi berbasis web ini juga berbanding lurus dengan serangan siber yang dihadapi. Pada penelitian ini dilakukan implementasi *Log Event Management Web Application Firewall* yang bertujuan untuk mendeteksi, mencatat log dan memblokir serangan siber yang dilakukan serta menyimpan, memvisualisasikan log dan menghadirkan notifikasi dari serangan tersebut. Implementasi dilakukan dengan menggunakan 2 server dengan sistem operasi Ubuntu Server 20.04 serta 1 komputer *system administrator* dengan sistem operasi Ubuntu Desktop 20.04. Server pertama digunakan untuk web server dengan *ModSecurity* sebagai *Web Application Firewall*, *Apache HTTP Server* sebagai Web Server dan *Filebeat* sebagai *log shipper*. Pada server kedua digunakan sebagai *ELK Stack* server dengan *Elasticsearch*, *Logstash* dan *Kibana* untuk menyimpan, mengurai dan memvisualisasikan log dari web server serta *elastalert* untuk mengirimkan notifikasi. Sedangkan pada komputer *system administrator* digunakan *Slack* untuk menerima notifikasi serangan. Pengujian dilakukan dengan melakukan 3 jenis serangan siber pada web *application* yaitu *SQL Injection*, *Local File Inclusion* dan *Cross Site Scripting*. Hasil dari pengujian tersebut menunjukkan bahwa serangan *SQL Injection*, *Local File Inclusion* dan *Cross Site Scripting* berhasil dideteksi, dicatat pada log dan diblokir oleh *ModSecurity*. Selain itu log yang dikirimkan pun berhasil disimpan, diurai dan divisualisasikan pada *ELK Stack* server serta adanya notifikasi serangan melalui *channel Slack* dalam waktu yang mendekati *real time*.

Kata kunci : *Log Event Management*, *Web Application Firewall*, *SQL Injection*, *Local File Inclusion*, *Cross Site Scripting*.

KATA PENGANTAR

Segala puji bagi Allah SWT Tuhan semesta alam atas rahmat dan hidayah-Nya, sehingga penulis dapat menyelesaikan skripsi dengan berjudul “Implementasi *Log Event Management Web Application Firewall* Menggunakan *Elasticsearch, Logstash* dan *Kibana*”. Untuk selanjutnya penulis mengucapkan banyak terima kasih kepada pihak-pihak yang telah membantu dalam penyelesaian skripsi ini, yaitu :

1. Irfan Purnawan, S.T, M.Chem.Eng, Selaku Dekan Fakultas Teknik Universitas Muhammadiyah Jakarta.
2. Popy Meilina, M.Kom, Selaku Ketua Program Studi Teknik Informatika Fakultas Teknik Universitas Muhammadiyah Jakarta.
3. Yana Adharani, M.Kom, Selaku Dosen Pembimbing Skripsi yang telah memberikan arahan, nasehat serta motivasi sehingga penulis dapat menyelesaikan skripsi ini.
4. Ayahanda dan Ibunda penulis yang selalu memberikan doa dan dukungan kepada penulis.
5. Seluruh pengajar dan staf Program Studi Teknik Informatika Fakultas Teknik Universitas Muhammadiyah Jakarta.
6. Muzammil Insani Alfaruqi, S.Kom yang selalu membantu penulis ketika mengalami kesulitan.
7. Seluruh mahasiswa Teknik Informatika FT-UMJ angkatan 2017 yang senantiasa menjadi teman diskusi untuk bertukar pikiran.
8. Keluarga Besar HMIF BEM FT-UMJ yang senantiasa memberikan semangat dan bantuan.
9. Semua pihak yang tidak dapat disebutkan satu per satu yang telah membantu penulis selama pengerjaan Skripsi.

Akhir kata penulis menyadari bahwa pada skripsi ini masih terdapat kekurangan dan kesalahan. Untuk itu penulis berharap kepada semua pihak dapat memaklumi serta memberikan kritik dan saran agar lebih baik lagi ke depannya.

Penulis juga berharap skripsi ini dapat memberikan manfaat bagi siapa pun yang membacanya.

Jakarta, 28 Juli 2021

Penulis,

A handwritten signature in blue ink, consisting of a large, stylized letter 'B' followed by a vertical line and a small flourish.

(Bukhari Hasan)

DAFTAR ISI

LEMBAR PERSETUJUAN.....	i
TANDA BUKTI PERSETUJUAN PEMBIMBING SKRIPSI.....	ii
LEMBAR PENGESAHAN	iii
LEMBAR PERNYATAAN	iv
SURAT PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIK.....	v
DAFTAR PRESENSI BIMBINGAN SKRIPSI	vi
ABSTRACT.....	viii
ABSTRAK	ix
KATA PENGANTAR	x
DAFTAR ISI.....	xii
DAFTAR TABEL.....	xvi
DAFTAR GAMBAR	xvii
BAB I PENDAHULUAN	1
1.1. Latar Belakang.....	1
1.2. Identifikasi Masalah	2
1.3. Rumusan Masalah	3
1.4. Batasan Masalah.....	4
1.5. Tujuan Penelitian.....	4
1.6. Metodologi Penelitian	5
1.7. Sistematika Penulisan.....	6
BAB II TINJAUAN PUSTAKA.....	8
2.1. Log.....	8
2.2. Serangan Siber.....	8
2.2.1. <i>SQL Injection</i>	9
2.2.2. <i>Cross Site Scripting (XSS)</i>	10
2.2.3. Local File Inclusion (LFI).....	10
2.3. ELK <i>Stack</i>	11
2.3.1. Elasticsearch.....	11
2.3.2. Logstash	12
2.3.3. Kibana	13

2.4.	Filebeat	13
2.5.	Jaringan Komputer	14
2.5.1.	Local Area Network (LAN)	14
2.5.2.	Metropolitan Area Network (MAN)	15
2.5.3.	Wide Area Network (WAN)	16
2.6.	Topologi Jaringan	16
2.6.1.	Topologi Bus	17
2.6.2.	Topologi Ring	17
2.6.3.	Topologi Mesh	18
2.6.4.	Topologi Tree	19
2.6.5.	Topologi Star	20
2.7.	Media Transmisi	21
2.7.1.	Kabel <i>Unshielded Twisted Pair</i> (UTP)	21
2.7.2.	Kabel <i>Shielded Twisted Pair</i> (STP)	22
2.8.	Router	22
2.9.	Switch	23
2.10.	Network Interface Card (NIC)	23
2.11.	IP (Internet Protocol) Address	24
2.12.	Web Server	25
2.12.1.	Apache HTTP Server	25
2.12.2.	Nginx	25
2.13.	ModSecurity	26
2.14.	Fail2ban	26
2.15.	Ubuntu	27
2.16.	Slack	27
2.17.	Contoh Implementasi Keamanan pada Web Server	27
2.18.	Metodologi Pengembangan	29
BAB III METODOLOGI PENELITIAN		31
3.1.	Web Server Tanpa Log Event <i>Management Web Application Firewall</i>	32
3.1.1.	Topologi Jaringan Komputer	33
3.1.2.	Spesifikasi Perangkat Keras	34
3.1.3.	Spesifikasi Perangkat Lunak	36
3.1.4.	Flow Map Pengecekan Log	37

3.2.	Analisis Permasalahan Web Server tanpa Log Event <i>Management Web Application Firewall</i>	38
3.3.	Perancangan Log Event Management Web Application Firewall	39
3.3.1.	Perancangan Topologi Jaringan	39
3.3.2.	Spesifikasi Perangkat Keras Tambahan	41
3.3.3.	Spesifikasi Perangkat Lunak Tambahan	42
3.3.4.	<i>Flow Map Log Event Management</i>	44
3.3.5.	Penggunaan IP Address.....	46
3.3.6.	Pembuatan Rules Logstash	47
3.3.7.	Desain Visualisasi Log.....	49
3.3.8.	Skenario Uji Coba.....	50
BAB IV HASIL DAN PEMBAHASAN		52
4.1.	Instalasi Sistem Operasi Ubuntu Server.....	52
4.2.	Instalasi Sistem Operasi Ubuntu Desktop.....	64
4.3.	Konfigurasi IP <i>Static</i>	71
4.4.	Konfigurasi Web Server.....	73
4.4.1.	Instalasi Perangkat Lunak pada Web Server.....	73
4.4.2.	Konfigurasi Apache Web Server dan MySQL.....	75
4.4.3.	Konfigurasi Web Application Firewall.....	76
4.4.4.	Konfigurasi Filebeat.....	79
4.4.5.	Konfigurasi Fail2ban.....	79
4.5.	Konfigurasi Komputer System Administrator	81
4.6.	Konfigurasi ELK Stack Server.....	83
4.6.1.	Instalasi Perangkat Lunak pada ELK <i>Stack</i> Server.....	83
4.6.2.	Konfigurasi Elasticsearch.....	85
4.6.3.	Pembuatan Rules Logstash	86
4.6.4.	Konfigurasi Nginx.....	95
4.6.5.	Konfigurasi Kibana	96
4.6.6.	Konfigurasi Elastalert.....	105
4.7.	Pengujian Log Event <i>Management Web Application Firewall</i>	107
4.7.1.	Uji Coba Serangan tanpa Log Event <i>Management Web Application Firewall</i>	108
4.7.2.	Uji Coba Serangan dengan Log Event <i>Management Web Application Firewall</i>	111

4.8. Pembahasan Hasil	119
BAB V KESIMPULAN DAN SARAN.....	121
5.1. Kesimpulan.....	121
5.2. Saran.....	122
DAFTAR PUSTAKA	123

DAFTAR TABEL

Tabel 3. 1 Spesifikasi Perangkat Keras Komputer Web Server.....	34
Tabel 3. 2 Spesifikasi Perangkat Keras Komputer System Administrator	34
Tabel 3. 3 Spesifikasi Media Penghubung dan Transmisi	35
Tabel 3. 4 Perangkat Lunak Komputer Web Server	36
Tabel 3. 5 Perangkat Lunak Komputer System Administrator	36
Tabel 3. 6 Spesifikasi Perangkat Keras ELK Stack Server.....	41
Tabel 3. 7 Spesifikasi Router	42
Tabel 3. 8 Perangkat Lunak Komputer ELK Stack Server	42
Tabel 3. 9 Perangkat Lunak Komputer Web Server	43
Tabel 3. 10 Perangkat Lunak Komputer System Administrator	44
Tabel 3. 11 Penggunaan IP Address	46
Tabel 3. 12 Skenario Uji Coba.....	51
Tabel 4. 1 Uji Coba Serangan	107
Tabel 4. 2 Hasil Pengujian Log Event Management Web Application Firewall	119

DAFTAR GAMBAR

Gambar 2. 1 Jaringan Local Area Network (K. T. Prasetyo, n.d.).....	15
Gambar 2. 2 Jaringan Metropolitan Area Network (Isky, 2018)	15
Gambar 2. 3 Jaringan Wide Area Network (Isky, 2018)	16
Gambar 2. 4 Contoh Topologi Bus (Haerudin et al., 2017).....	17
Gambar 2. 5 Contoh Topologi Ring (Haerudin et al., 2017)	18
Gambar 2. 6 Contoh Topologi Mesh (Haerudin et al., 2017)	18
Gambar 2. 7 Contoh Topologi Tree	19
Gambar 2. 8 Contoh Topologi Star (Simanullang, Napitupulu, Jamaluddin, & Purba, 2018)	20
Gambar 2. 9 Kabel Unshielded Twisted Pair (UTP) (Sora, 2015).....	21
Gambar 2. 10 Kabel Shielded Twisted Pair (Mail, 2020).....	22
Gambar 2. 11 Router (Kurniawan, 2020).....	23
Gambar 2. 12 Contoh Switch (Peniarsih, 2020)	23
Gambar 2. 13 Network Interface Card (Peniarsih, 2020)	24
Gambar 2. 14 Contoh Topologi Jaringan Tanpa Firewall (Sahren, 2021).....	28
Gambar 2. 15 Topologi Jaringan Dengan Firewall (Sahren, 2021)	28
Gambar 2. 16 Topologi Jaringan Penerapan ModSecurity (Riska & Alamsyah, 2021)	29
Gambar 2. 17 Metode Pengembangan (Sholihah et al., 2020).....	30
Gambar 3. 1 Metodologi Penerapan Log Event Management Web Application Firewall	31
Gambar 3. 2 Topologi Jaringan Komputer	33
Gambar 3. 3 Flow Map Pengecekan Log.....	37
Gambar 3. 4 Contoh Log pada Command Line Interface (CLI).....	38
Gambar 3. 5 Topologi Jaringan Log Event Management Web Application Firewall	40
Gambar 3. 6 Flow Map Log Event Management.....	45
Gambar 3. 7 Mock-up Desain Dashboard Log Modsecurity	50
Gambar 4. 1 Pilih Bahasa Ubuntu Server	53
Gambar 4. 2 Update Ubuntu Server	54
Gambar 4. 3 Konfigurasi Keyboard Ubuntu Server.....	55
Gambar 4. 4 Konfigurasi Jaringan Ubuntu Server.....	56
Gambar 4. 5 Konfigurasi Proxy Ubuntu Server.....	57
Gambar 4. 6 Konfigurasi Ubuntu Archive Mirror Ubuntu Server.....	58
Gambar 4. 7 Konfigurasi Penyimpanan Ubuntu Server.....	59
Gambar 4. 8 File System Summary Ubuntu Server	60
Gambar 4. 9 Profile Setup Ubuntu Server	61
Gambar 4. 10 SSH Setup Ubuntu Server	62
Gambar 4. 11 Featured Server Snaps Ubuntu Server	63
Gambar 4. 12 Reboot Ubuntu Server	64
Gambar 4. 13 Tampilan Awal Instalasi Ubuntu Desktop	65

Gambar 4. 14 Konfigurasi Keyboard Ubuntu Desktop.....	66
Gambar 4. 15 Jenis Instalasi Ubuntu Desktop	67
Gambar 4. 16 Konfigurasi Ruang Penyimpanan Ubuntu Desktop	68
Gambar 4. 17 Penentuan Lokasi Ubuntu Desktop	69
Gambar 4. 18 Profile Setup Ubuntu Desktop.....	70
Gambar 4. 19 Kotak Dialog Restart Ubuntu Desktop.....	71
Gambar 4. 20 Nama Network Interface	72
Gambar 4. 21 Konfigurasi IP Static	73
Gambar 4. 22 Tampilan Web Application	76
Gambar 4. 23 Konfigurasi ModSecurity pada Sites Available Apache.....	78
Gambar 4. 24 Konfigurasi Filebeat.....	79
Gambar 4. 25 Konfigurasi Jail File pada Fail2ban	80
Gambar 4. 26 Konfigurasi Filter Apache Modsecurity pada Fail2ban	81
Gambar 4. 27 Tampilan Awal Slack.....	82
Gambar 4. 28 Penambahan Incoming Webhooks pada Slack.....	82
Gambar 4. 29 Konfigurasi Incoming Webhooks pada Slack.....	83
Gambar 4. 30 Konfigurasi Elasticsearch.....	85
Gambar 4. 31 Grok Filter Waktu, Jenis Log, IP Attacker dan Pesan ModSecurity	87
Gambar 4. 32 Grok Filter File Rules ModSecurity.....	87
Gambar 4. 33 Grok Filter Nama Serangan.....	88
Gambar 4. 34 Grok Filter Pesan Keterangan Serangan	88
Gambar 4. 35 Grok Filter Keterangan Serangan.....	89
Gambar 4. 36 Grok Filter Data Pola Serangan	89
Gambar 4. 37 Grok Filter Pola Serangan	90
Gambar 4. 38 Grok Filter Data Halaman yang Diserang.....	90
Gambar 4. 39 Grok Filter Halaman yang Diserang	91
Gambar 4. 40 Grok Filter IP Address Web Server	91
Gambar 4. 41 Konfigurasi Nginx.....	96
Gambar 4. 42 Konfigurasi Kibana	97
Gambar 4. 43 Autentikasi Kibana Dashboard.....	98
Gambar 4. 44 Tampilan Kibana Dashboard.....	98
Gambar 4. 45 Pembuatan Index Pattern pada Kibana.....	99
Gambar 4. 46 Pembuatan Nama Index Pattern pada Kibana	99
Gambar 4. 47 Tampilan Pilih Sub Menu Dashboard pada Kibana	100
Gambar 4. 48 Tampilan Pembuatan Visualisasi pada Kibana	101
Gambar 4. 49 Tampilan Visualisasi Pie Chart pada Kibana	101
Gambar 4. 50 Filter Visualisasi Pie Chart pada Kibana	102
Gambar 4. 51 Tampilan Visualisasi Bar Chart pada Kibana	103
Gambar 4. 52 Tampilan Visualisasi Tabel pada Kibana.....	103
Gambar 4. 53 Filter Visualisasi Tabel pada Kibana	104
Gambar 4. 54 Tampilan Dashboard Kibana.....	105
Gambar 4. 55 Konfigurasi Elastalert.....	106
Gambar 4. 56 Konfigurasi Rules Elastalert.....	107

Gambar 4. 57 Uji Coba Serangan Local File Inclusion tanpa Log Event Management Web Application Firewall	109
Gambar 4. 58 Uji Coba Serangan Cross Site Scripting tanpa Log Event Management Web Application Firewall	110
Gambar 4. 59 Uji Coba Serangan SQL Injecton tanpa Log Event Management Web Application Firewall	110
Gambar 4. 60 Log Error Apache.....	111
Gambar 4. 61 Uji Coba Serangan Local File Inclusion dengan Log Event Management Web Application Firewall	112
Gambar 4. 62 Pemblokiran IP Address Attacker atas Serangan LFI.....	113
Gambar 4. 63 Notifikasi Serangan LFI pada Slack.....	113
Gambar 4. 64 Uji Coba Serangan Cross Site Scripting dengan Log Event Management Web Application Firewall	114
Gambar 4. 65 Pemblokiran IP Address Attacker atas Serangan XSS.....	115
Gambar 4. 66 Notifikasi Serangan XSS pada Slack	115
Gambar 4. 67 Uji Coba Serangan SQL Injetcion dengan Log Event Management Web Application Firewall	116
Gambar 4. 68 Pemblokiran IP Address Attacker atas Serangan SQLI.....	117
Gambar 4. 69 Notifikasi Serangan SQLI pada Slack.....	117
Gambar 4. 70 Log ModSecurity pada Web Server	118
Gambar 4. 71 Tampilan Hasil Uji Coba Serangan pada Dashboard Kibana	119

BAB I

PENDAHULUAN

1.1. Latar Belakang

Aplikasi berbasis web merupakan salah satu jenis aplikasi yang banyak digunakan untuk berbagai keperluan seperti *internet banking*, *e-commerce* hingga *e-government*. Namun, maraknya penggunaan aplikasi berbasis web ini berbanding lurus dengan serangan siber yang dihadapi. Serangan siber bertujuan untuk bertujuan untuk mencuri, mengubah, merusak atau menghapus data-data rahasia yang ada pada suatu sistem.

Salah satu serangan yang sering dihadapi oleh aplikasi berbasis web yaitu *web defacement*. *Web defacement* adalah suatu tindakan perubahan tampilan suatu halaman web yang dilakukan seseorang yang tidak memiliki hak akses. Berdasarkan data Pusat Operasi Keamanan Siber Nasional Badan Siber dan Sandi Negara dalam Admi dkk menuliskan bahwa pada tahun 2018 dari total 16.939 serangan *web defacement*, 30,75% dialami oleh *website* pemerintahan dengan domain *.go.id*, diikuti oleh *website* kampus dengan domain *ac.id* dengan persentase sebanyak 28,38%, *website* sekolah dengan domain *.sch.id* sebanyak 12,58%, *website* dengan domain *.co.id* sebanyak 10,92% dan *website* dengan domain *.id* sebanyak 8,25% (Admi, Hakim, & Maulana, 2020).

Salah satu upaya yang dapat dilakukan guna mencegah penyerangan terhadap *website* yaitu dengan menerapkan *Web Application Firewall*. *Web Application Firewall* berfungsi untuk memfilter, memonitor serta melakukan blok terhadap data yang diminta oleh *client* terhadap web server. Salah satu *Web Application Firewall* yang cukup populer adalah *ModSecurity*. *ModSecurity* merupakan sebuah *web application firewall* yang bersifat *open source* dan lintas platform yang dikembangkan oleh *Trustwave's SpiderLabs*. Setiap serangan terhadap *website* yang terdeteksi oleh *modsecurity* akan diblokir dan dicatat dalam *log file*.

Namun, untuk membaca log tersebut seorang administrator diharuskan untuk mengakses server secara langsung. Hal ini menyebabkan seorang *system*

administrator hanya dapat mengetahui serangan yang terjadi ketika membaca log tersebut pada server. Selain itu server yang diharuskan berjalan selama 24 jam non stop akan menghasilkan banyak log. Log juga ditampilkan dalam bentuk tulisan pada *command line interface* sehingga *system administrator* diharuskan memiliki pengetahuan khusus untuk dapat memahami dan menganalisis log tersebut.

Oleh karena itu dibutuhkan suatu *tools* atau sistem untuk memvisualisasi data dari suatu *log file* guna mempermudah proses pembacaan dan analisis serta menghadirkan notifikasi terhadap serangan siber yang terjadi. Menurut M. Arifin dkk dalam membangun *log event management, Elasticsearch Logstash Kibana (ELK Stack)* merupakan *tools* yang tepat karena dapat menampilkan tren, statistik dan anomali yang terjadi (Arifin, Susilowati, & Sugiartowo, 2018). Adapun kelebihan *ELK Stack* antara lain (Sholihah, Pripambudi, & Mardiyono, 2020) :

1. Skalabilitas, di mana *ELK Stack* memiliki potensi untuk dikembangkan.
2. Keandalan, di mana *elasticsearch* dapat secara otomatis mendistribusikan data.
3. Otomatis, di mana *ELK Stack* secara otomatis akan menyimpan dan melakukan pengindeksan data dalam bentuk JSON.
4. Ramah pengguna, di mana setiap jenis data sumber yang di indeks dapat divisualisasikan oleh *ELK Stack*.

Berdasarkan latar belakang di atas penulis membuat skripsi dengan judul **“IMPLEMENTASI LOG EVENT MANAGEMENT WEB APPLICATION FIREWALL MENGGUNAKAN ELASTICSEARCH LOGSTASH KIBANA”**.

1.2. Identifikasi Masalah

Berdasarkan latar belakang di atas, identifikasi masalah dari skripsi ini yaitu aplikasi berbasis web sangat rawan terhadap serangan siber yang dilakukan oleh *Attacker*. Dengan tidak adanya keamanan yang diterapkan pada aplikasi

berbasis web, memungkinkan *attacker* melakukan berbagai jenis serangan siber terhadap web server. Serangan *attacker* tersebut memungkinkan terjadinya pencurian, perubahan, perusakan hingga penghapusan data yang ada pada suatu sistem.

Berbagai serangan siber di atas tidak dapat diidentifikasi oleh aplikasi web server karena tidak menggunakan web *application firewall*, sehingga tidak ada jejak serangan siber pada log *file* web server. Dengan tidak adanya jejak yang ditinggalkan oleh *attacker*, seorang *system administrator* tidak dapat mengetahui serangan siber yang dialami web server. Selain itu apabila terjadi peretasan, seorang *system administrator* juga tidak dapat mengetahui bagian pada web *application* yang memiliki celah keamanan. Hal ini menyulitkan identifikasi celah keamanan untuk perbaikan web *application* pasca peretasan.

Untuk mengatasi permasalahan tersebut dapat digunakan *Web Application Firewall*. Akan tetapi *Web Application Firewall Modsecurity* yang berjalan 24 jam non stop menghasilkan log yang sangat banyak. Log juga ditampilkan dalam bentuk tulisan pada *command line interface*. Hal ini menyebabkan log sulit dianalisis dan diperlukan pengetahuan khusus untuk dapat membaca dan memahami isi log tersebut.

Permasalahan lain yang muncul adalah tidak adanya notifikasi ketika terdapat serangan siber yang dilakukan. Dengan demikian serangan siber hanya dapat diketahui ketika log *modsecurity* tersebut dibaca dengan cara mengakses server secara langsung. Hal ini menyebabkan serangan siber sering terlambat untuk ditangani.

1.3. Rumusan Masalah

Adapun rumusan masalah berdasarkan latar belakang dan identifikasi masalah di atas antara lain :

1. Bagaimana cara meningkatkan keamanan dari suatu *web application* dengan menggunakan *Web Application Firewall*?

2. Bagaimana cara mendeteksi serangan siber terhadap web server agar tercatat pada *log file*?
3. Bagaimana cara melakukan *log event management* terhadap *log ModSecurity* yang ada pada web server?
4. Bagaimana cara menghadirkan notifikasi apa bila terjadi serangan pada web server?

1.4. Batasan Masalah

Batasan masalah digunakan penulis untuk menghindari melebarnya pembahasan. Adapun batasan masalah tersebut antara lain :

1. *Web Application Firewall* yang digunakan yaitu *ModSecurity*.
2. Menggunakan Ubuntu Server 20.04 LTS sebagai server dengan *Web Application Firewall*.
3. Menggunakan Ubuntu Server 20.04 LTS sebagai server dengan *Elasticsearch Logstash Kibana (ELK Stack)*.
4. Menggunakan *OWASP ModSecurity Core Rule Set* sebagai *rules* untuk memfilter serangan yang diterima web server.
5. Pengujian dilakukan dengan melakukan serangan *SQL Injection*, *Local File Inclusion (LFI)* dan *Cross Site Scripting (XSS)* terhadap server dengan *Web Application Firewall*.
6. Notifikasi terhadap serangan siber yang diterima web server dikirimkan ke *channel Slack*.

1.5. Tujuan Penelitian

Adapun tujuan dari penelitian ini antara lain :

1. Meningkatkan keamanan dari *website* terhadap serangan-serangan siber yang diterima dengan menerapkan *Web Application Firewall*.

2. Mengaktifkan *ModSecurity* untuk mendeteksi serangan siber seperti *Local File Inclusion*, *Cross Site Scripting* dan *SQL Injection*, serta menghasilkan log serangan.
3. Mengintegrasikan *ModSecurity* sebagai *Web Application Firewall* dengan *Elasticsearch Logstash Kibana (ELK Stack)* agar setiap serangan yang terdeteksi oleh *Web Application Firewall* dapat diurai, disimpan dan divisualisasikan pada *ELK Stack* server.
4. Menghadirkan notifikasi terhadap serangan siber yang diterima web server agar jika terjadi serangan pada web server dapat diketahui dalam jangka waktu yang singkat.

1.6. Metodologi Penelitian

Untuk mendapatkan hasil yang optimal, pada penelitian ini dilakukan langkah-langkah yang sistematis. Adapun metodologi penelitian pada laporan skripsi ini antara lain :

1. Studi Literatur

Studi literatur dilakukan dengan cara mencari dan mempelajari buku-buku, jurnal-jurnal, artikel ilmiah dan sumber lainnya yang berkaitan dengan *Log Management*, *Cyber Attack*, *Web Application Firewall* dan *ELK Stack*.

2. Analisis

Proses analisis diperlukan mengidentifikasi permasalahan yang ada serta cara penyelesaiannya. Selain itu tahap analisis ini juga diperlukan untuk mengetahui kebutuhan-kebutuhan perangkat keras dan perangkat lunak untuk merancang dan mengimplementasikan *log event management web application firewall* serta untuk mengetahui serangan-serangan siber yang nantinya akan digunakan pada tahap uji coba.

3. Perancangan

Setelah mengumpulkan data-data yang diperlukan dan melakukan analisis terhadap data-data yang terkumpul tersebut, tahap selanjutnya yakni dengan

melakukan proses perancangan. Proses perancangan ini meliputi perancangan infrastruktur perangkat keras serta alur kerja yang akan digunakan untuk mengimplementasikan *modsecurity* sebagai *web application firewall*, *ELK Stack* dan menghadirkan notifikasi dari serangan yang diterima.

4. Implementasi

Tahap selanjutnya yakni melakukan implementasi berdasarkan data-data dan perancangan yang telah dilakukan sebelumnya. Proses implementasi ini dilakukan dengan melakukan instalasi dan konfigurasi *ELK Stack* dan *Web Application Firewall ModSecurity* pada masing-masing servernya kemudian mengintegrasikannya serta menghadirkan notifikasi terhadap serangan yang diterima. Selain itu pada tahap ini juga dibuatkan *rules* untuk memfilter log yang masuk dari *ModSecurity* ke *ELK Stack*.

5. Uji Coba

Pada tahap uji coba ini dilakukan pengujian serangan siber terhadap server dengan *Web Application Firewall ModSecurity*. Adapun pengujian serangan-serangan siber yang akan dilakukan yakni *SQL Injection*, *Local File Inclusion* dan *Cross Site Scripting (XSS)*.

1.7. Sistematika Penulisan

Sistematika penulisan pada laporan skripsi ini disusun ke dalam lima bab, yaitu antara lain :

BAB I PENDAHULUAN

Bab ini membahas seputar latar belakang, identifikasi masalah, rumusan masalah, batasan masalah, tujuan penelitian, metodologi penelitian dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Pada bab ini berisikan landasan terhadap teori-teori yang digunakan pada laporan skripsi ini, seperti log, *elasticsearch*, *logstash*, *kibana*, *ubuntu*, *local file*

inclusion, SQL Injection, cross site scripting dan teori lainnya yang berkaitan dengan *log event management web application firewall*.

BAB III METODOLOGI PENELITIAN

Bab ini berisikan pembahasan mengenai langkah-langkah untuk membangun *log event management web application firewall* dengan *elasticsearch logstash* dan *kibana*. Dimulai dari pembuatan topologi jaringan, kebutuhan perangkat keras dan perangkat lunak, alur *log management* pada dengan *ELK Stack*, pemberian alamat *IP address*, desain visualisasi log dan skenario uji coba untuk menguji *log event management* dan notifikasi.

BAB IV HASIL DAN PEMBAHASAN

Bab ini berisikan pembahasan mengenai proses instalasi dan konfigurasi, baik terhadap server dengan *web application firewall modsecurity* maupun terhadap server dengan *ELK Stack*. Pada bab ini juga dilakukan pembuatan *rules* untuk memfilter log yang terima dari *modsecurity*. Selain itu pada bab ini juga berisikan hasil dari pengujian serta dokumentasi dari *log event management web application firewall* yang telah dibuat.

BAB V KESIMPULAN DAN SARAN

Pada bab ini terdapat kesimpulan berdasarkan uraian yang telah dipaparkan pada bab sebelumnya. Selain itu pada bab ini juga berisikan saran untuk pengembangan lebih lanjut terhadap *log event management* ataupun *web application firewall*.

BAB II

TINJAUAN PUSTAKA

2.1. Log

Log merupakan sebuah catatan atau rekaman dari aktivitas yang terjadi sebelumnya. *Logging* merupakan kemampuan untuk menuliskan semua catatan yang terjadi seperti pada *server*, *client* dan seluruh perangkat lainnya yang ada pada infrastruktur dari sebuah sistem IT. Log dapat membantu untuk melakukan audit untuk melacak tentang apa yang terjadi, sumber daya yang digunakan dengan detail waktu dan informasi lainnya tentang *user* dan kegiatannya (Al-Mahbashi, Potdar, & Chauhan, 2017). Adapun kesulitan dalam melakukan manajemen log antara lain (GÜL & YILMAZ, 2019):

1. Log diproduksi di berbagai sistem dalam jumlah yang besar.
2. Setiap sistem membuat tipe log yang berbeda.
3. Isi dari setiap log berbeda satu sama lain.

Sedangkan *Log event management* merupakan suatu cara untuk melakukan manajemen terhadap aktivitas-aktivitas yang dicatat pada suatu log *file*. Setiap upaya pengaksesan yang berhasil maupun tidak berhasil pada sistem IT tercatat pada log *file* (Jankovic, 2012). Informasi kunci pada log yaitu entitas yang bertukar data serta waktu kejadian pertukaran data tersebut (Jankovic, 2012). Pada Dengan melakukan *management* terhadap log pada suatu server, data-data yang tertulis pada log *file* akan diurai dan disimpan sehingga tersebut dapat digunakan untuk pemantauan aktivitas *user* ataupun sebagai dasar analisis keamanan pada server tersebut.

2.2. Serangan Siber

Serangan siber (*cyber attack*) adalah suatu jenis serangan yang dilakukan dengan sengaja oleh individu atau sekelompok orang untuk mendapatkan akses ilegal terhadap sistem komputer atau jaringan komputer suatu individu, organisasi atau instansi. Tujuan dari serangan siber yaitu memberikan dampak yang nyata pada aktivitas korban di dunia nyata dengan melakukan pemusnahan integritas

(*loss integrity*), ketersediaan (*availability*), kerahasiaan (*confidentiality*) dan pemusnahan fisik (*physical destruction*) (Tampubolon, 2019).

Adapun menurut Kementerian Pertahanan Republik Indonesia, dampak yang mungkin ditimbulkan akibat serangan siber antara lain (Kementerian Pertahanan, 2014) :

1. Gangguan fungsional.
2. Pengendalian sistem secara *remote*.
3. Penyalahgunaan informasi.
4. Kerusakan, ketakutan, kekerasan, kekacauan, konflik.
5. Serta kondisi lain yang sangat merugikan, sehingga memungkinkan dapat mengakibatkan kehancuran.

2.2.1. SQL Injection

SQL *Injection* merupakan salah satu serangan yang paling sering dilancarkan hingga saat ini. SQL *Injection* merupakan salah satu ancaman terhadap *database system* di mana *attacker* menambahkan SQL *statement* pada sebuah aplikasi melalui *input box* untuk mendapatkan akses atau mengubah data yang tersimpan pada *database* (Alwan & Younis, 2017). *Open Web Application Security Project* (OWASP) menuliskan bahwa SQL *Injection* berada di urutan paling atas pada daftar *Top 10 Web Application Security Risk*. Adapun beberapa imbas jika *attacker* berhasil melakukan SQL *Injection* pada web server antara lain (Robinson, Akbar, & Ridha, 2018):

1. Seorang *attacker* dapat menggunakan SQL *Injection* untuk mem-*bypass* autentikasi atau meniru salah satu user.
2. Sebuah celah keamanan SQL *Injection* memungkinkan terbukanya data yang ada pada *database* server.
3. Seorang *attacker* dapat menggunakan SQL *Injection* untuk mengubah data yang tersimpan dalam *database*.

4. Seorang *attacker* dapat menggunakan *SQL Injection* untuk menghapus data yang tersimpan pada *database*.

2.2.2. Cross Site Scripting (XSS)

Selain *SQL Injection*, *Cross Site Scripting (XSS)* juga masuk ke dalam daftar OWASP *Top 10 Web Application Security Risk*. *Cross Site Scripting (XSS)* merupakan sebuah serangan di mana *attacker* memasukkan kode berbahaya khususnya dalam *JavaScript* pada aplikasi berbasis web. Adapun jenis-jenis serangan XSS antara lain (Chinprutthiwong, Vardhan, Yang, & Gu, 2020):

1. *Stored XSS Attack*, yaitu suatu jenis serangan XSS di mana seorang *attacker* membuat kode berbahaya dan menavigasi ke URL dengan parameter akan disimpan dalam *database* server. Sehingga ketika korban membuka halaman dengan kode berbahaya tersebut, memungkinkan *attacker* untuk mencuri informasi sensitif dari korban.
2. *Reflected XSS*, yaitu suatu jenis serangan XSS di mana seorang *attacker* melakukan *redirect* pada korban untuk mengunjungi URL dengan parameter berbahaya. Pada kasus ini, parameter berbahaya tersebut tidak disimpan, tetapi dengan cepat memantulkan dan mengeksekusi parameter berbahaya tersebut pada browser korban.
3. *Document Object Model (DOM) XSS*, yaitu suatu jenis serangan XSS di mana *attacker* mengarahkan korban kepada URL yang berbahaya. Sehingga ketika *website* membaca URL tersebut akan menuliskan URL parameter yang berbahaya tersebut tanpa *sanitization* yang benar.

2.2.3. Local File Inclusion (LFI)

Local File Inclusion (LFI) merupakan salah satu jenis celah keamanan yang ada pada *web application*. *Local File Inclusion* merupakan celah keamanan di mana *attacker* mengumpulkan informasi rahasia dengan mengakses *file* pada server korban (Hasan & Meva, 2018). Cara kerja serangan *Local File Inclusion* yaitu dengan mengubah URL pada halaman *website* dengan *file* yang ingin dibaca. Salah satu penyebab terjadinya serangan *Local File Inclusion* yaitu

kesalahan pada proses *coding*, sebagai contoh tidak divalidasi dan difilternya fungsi *include()* dengan baik dan benar (Koprawi, 2020).

2.3. ELK Stack

ELK Stack merupakan kumpulan dari beberapa komponen perangkat lunak (*software*) *open source* yakni *Elasticsearch*, *Logstash* dan *Kibana* yang dikembangkan oleh sebuah perusahaan yang bernama *elastic.co*. ELK Stack dapat digunakan sebagai solusi log manajemen, basis data, mesin pencari dan banyak operasi lainnya untuk organisasi yang ingin mendirikan *Security Operation Center* dengan biaya yang relatif rendah (Babu, Prasad, & Prasad, 2019). Adapun kelebihan ELK Stack antara lain (Sholihah et al., 2020) :

1. Skalabilitas, di mana ELK Stack memiliki potensi untuk dikembangkan.
2. Keandalan, di mana *elasticsearch* dapat secara otomatis mendistribusikan data.
3. Otomatis, di mana ELK Stack secara otomatis akan menyimpan dan melakukan pengindeksan data dalam bentuk JSON.
4. Ramah pengguna, di mana setiap jenis data sumber yang di indeks dapat divisualisasikan oleh ELK Stack.

ELK Stack juga dapat digunakan untuk keperluan log analisis yang dapat membantu untuk *issue debugging*, *performance anlysis*, *predictive analysis*, *security analysis*, *internet of things* dan *logging* (Chhaged, 2015).

2.3.1. Elasticsearch

Elasticsearch merupakan perangkat lunak *open source* yang menjadi komponen utama pada ELK Stack. *Elasticsearch* merupakan sebuah mesin pencari yang *scalable* sehingga memungkinkan penggunaanya untuk melakukan penyimpanan, pencarian dan analisis data yang besar dalam waktu yang cepat (Atmaja & Yulianto, 2019). *Elasticsearch* mengambil data yang belum terstruktur dari tempat lain, kemudian menyimpan dan melakukan indeks sesuai dengan

pemetaan yang ditentukan oleh penggunaannya sehingga data-data tersebut dapat dicari.

Adapun beberapa fitur kunci yang ada pada *Elasticsearch* antara lain (Chhajed, 2015):

1. *Elasticsearch* didistribusikan secara *open source*, *scalable* dan dapat menyimpan dokumen secara *real-time*.
2. *Elasticsearch* menyediakan pencarian secara *real-time* dan kemampuan untuk menganalisis.
3. *Elasticsearch* menyediakan RESTful API yang canggih untuk keperluan pencarian.
4. *Elasticsearch* dapat diintegrasikan dengan mudah pada infrastruktur berbasis *cloud*, seperti *Amazon Web Service (AWS)* dan lainnya.

2.3.2. Logstash

Pada saat data dikirim kepada *ELK Stack*, *Logstash* merupakan *tool* pertama yang mengolah data tersebut sebelum diteruskan kepada *Elasticsearch*. *Logstash* merupakan sebuah data *pipeline* yang membantu mengumpulkan, mengurai dan menganalisis bermacam-macam variasi data baik yang terstruktur maupun tidak terstruktur yang dihasilkan dari berbagai sistem (Chhajed, 2015). Pada *pipeline* tersebut data diproses kedalam tiga tahap, yakni *input*, *filter* dan *output*.

Tahap yang pertama yaitu proses *input*, di mana *Logstash* akan menerima *input* yang dikirimkan dari sumber melalui *port* tertentu. *Logstash* mendukung beberapa tipe *input* dari semua sumber data yang umum termasuk *CSV file*, *TCP/UDP socket*, *HTTP API endpoint*, *Elasticsearch* dan lain sebagainya (Bajer, 2017). Setelah menerima data tersebut, *Logstash* akan memproses data tersebut pada tahap *filter*. Pada tahap filter ini, *Logstash* akan mengurai data, menghapus data, mengubah data ataupun menambahkan data sesuai dengan *rules* yang dibuat. Selanjutnya yang terakhir ialah *output*, di mana data yang telah diproses tersebut akan dikirimkan ke tujuan yang ditetapkan. Pada umumnya, *Logstash* akan

mengirimkan data tersebut kepada *Elasticsearch*, namun *Logstash* secara independen juga dapat menyimpan data dalam *CSV file*, pada *SQL database*, data *analytic* seperti *Azure Machine Learning* atau hanya menampilkannya pada *console* untuk keperluan *debugging* (Bajer, 2017).

2.3.3. Kibana

Pada *ELK Stack*, *Kibana* merupakan platform yang digunakan untuk melakukan visualisasi. *Kibana* merupakan tampilan antar muka berbasis web yang digunakan untuk menampilkan data yang tersimpan pada *Elasticsearch* (Babu et al., 2019). *Kibana* menggunakan kemampuan pencarian dan *indexing* pada *Elasticsearch* melalui *RESTful API* untuk menampilkannya dalam bentuk grafis kepada *end user*. *Kibana* dapat menampilkan data-data tersebut dalam bentuk histogram, *geomaps*, *pie charts*, *graphs*, *tables* dan lain sebagainya (Chhaged, 2015).

Kibana terbagi ke dalam 4 komponen utama yakni (Bajer, 2017):

1. *Management*, yaitu komponen yang memungkinkan pengguna untuk melakukan konfigurasi internal pada *Kibana*.
2. *Discover*, yaitu komponen yang memungkinkan pengguna untuk melihat dan menganalisis data murni yang masuk.
3. *Visualize*, yaitu komponen yang memungkinkan penggunanya untuk memvisualisasikan data dengan salah satu *plugin* visualisasi yang tersedia.
4. *Dashboard*, yaitu komponen yang memungkinkan pengguna untuk menggabungkan beberapa *Discover* dan *Visualize* yang tersimpan dalam satu tampilan.

2.4. Filebeat

Filebeat merupakan salah satu *data shipper* ringan yang didesain untuk di-*install* pada mesin yang menghasilkan data tanpa mempengaruhi performa mesin (Hamilton, Gonzalez Berges, Tournier, & Schofield, 2018). Pada penelitian ini

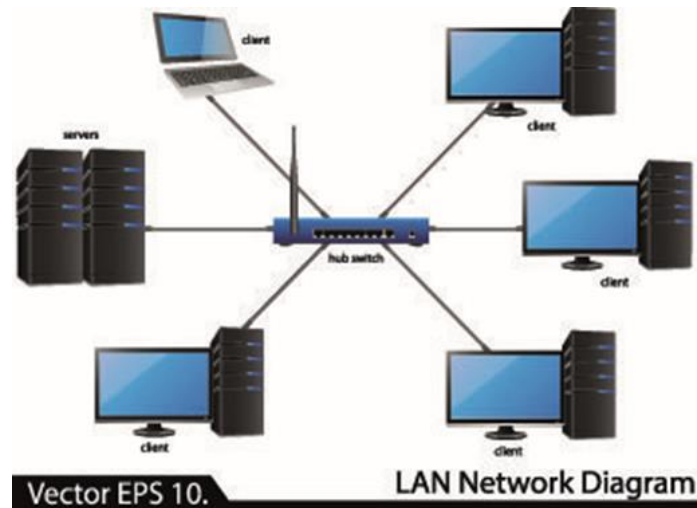
filebeat bertugas untuk mengirimkan log dari server dengan *ModSecurity* kepada server dengan *ELK Stack*. Log yang dikirimkan melalui *filebeat* tersebut nantinya akan diterima oleh *Logstash* dan difilter sesuai dengan *rules* yang dibuat.

2.5. Jaringan Komputer

Jaringan komputer merupakan sekumpulan perangkat-perangkat seperti komputer, printer dan perangkat lainnya yang terhubung menggunakan media transmisi baik kabel maupun nirkabel untuk saling bertukar data, informasi dan berbagi sumber daya (Riska, Sugiartawan, & Wiratama, 2018). Suatu jaringan komputer dapat dikatakan terkoneksi jika dua komputer atau lebih dapat saling bertukar data, informasi dan berbagi sumber daya satu sama lain. Berdasarkan cakupan geografis jaringan komputer dibagi ke dalam beberapa jenis yakni Local Area Network (LAN), Metropolitan Area Network (MAN) dan Wide Area Network (WAN).

2.5.1. Local Area Network (LAN)

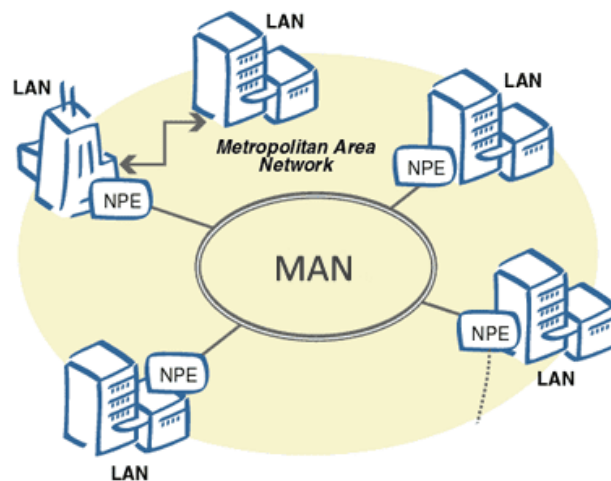
Local Area Network (LAN) adalah jaringan komputer dengan cakupan area yang terbatas, seperti pada suatu ruangan atau suatu gedung (Fauzi & Desmulyati, 2020). Local Area Network biasanya dihubungkan dengan dua jenis media transmisi yakni *wire* dan *wireless*. Local Area Network yang terhubung tanpa perantara kabel (*wireless*) biasanya disebut dengan Wireless Local Area Network (WLAN). Adapun contoh dari jaringan komputer Local Area Network tertera pada gambar 2.1.



Gambar 2. 1 Jaringan Local Area Network (K. T. Prasetyo, n.d.)

2.5.2. Metropolitan Area Network (MAN)

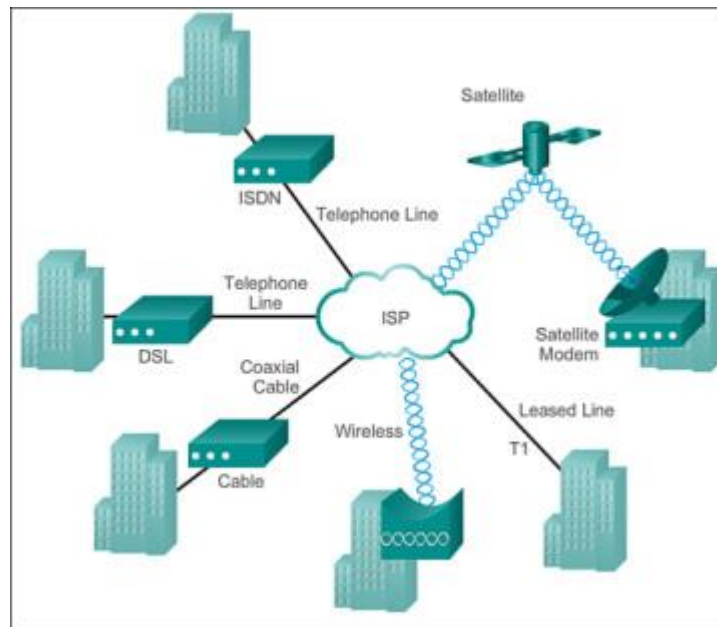
Metropolitan Area Network (MAN) merupakan jaringan komputer yang menghubungkan beberapa jaringan LAN ke dalam cakupan yang lebih besar. Metropolitan Area Network menghubungkan jaringan-jaringan yang lebih kecil tersebut yang ada di lokasi yang berbeda dengan kecepatan tinggi. Jaringan MAN biasanya digunakan untuk menghubungkan kampus, bank dan lain sebagainya. Adapun contoh dari jaringan Metropolitan Area Network tertera pada gambar 2.2.



Gambar 2. 2 Jaringan Metropolitan Area Network (Isky, 2018)

2.5.3. Wide Area Network (WAN)

Wide Area Network (WAN) merupakan jaringan komputer dengan cakupan yang paling luas. Jaringan WAN merupakan gabungan dari LAN dan MAN yang terpisah sangat jauh secara geografis. WAN biasanya dihubungkan dengan media satelit dan kabel fiber optic (Widodo, Mushansyah, & Ambarsari, 2019). Adapun contoh jaringan WAN tertera pada gambar 2.3.



Gambar 2. 3 Jaringan Wide Area Network (Isky, 2018)

2.6. Topologi Jaringan

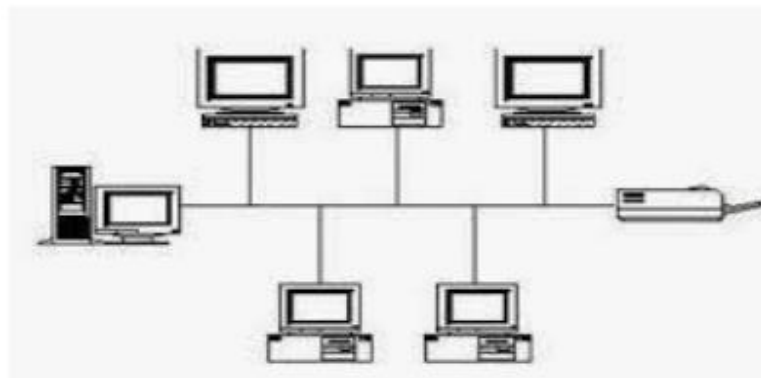
Menurut Madcom dalam Armanto menyatakan bahwa topologi jaringan adalah suatu bentuk pola penggambaran dari komponen-komponen jaringan yang saling terhubung seperti komputer server, komputer *client/workstation*, *hub/switch*, penggunaan kabel serta komponen jaringan lainnya (Armanto, 2017). Topologi jaringan diterapkan guna mempermudah *maintanance* komponen-komponen yang terlibat. Selain itu dengan diterapkannya suatu topologi, pengembangan jaringan komputer akan lebih mudah dilakukan. Jenis-jenis topologi jaringan di antaranya :

1. Topologi *Bus*
2. Topologi *Ring*

3. Topologi *Mesh*
4. Topologi *Tree*
5. Topologi *Star*

2.6.1. Topologi Bus

Topologi Bus adalah suatu topologi yang memiliki satu kabel *coaxial* utama yang kedua ujungnya diakhiri dengan terminator (Haerudin, Aksara, & Yamin, 2017). Untuk terhubung dengan node, kabel utama tersebut disambungkan menggunakan *T-Connector* dan konektor BNC. Adapun contoh dari topologi bus tertera pada gambar 2.4.

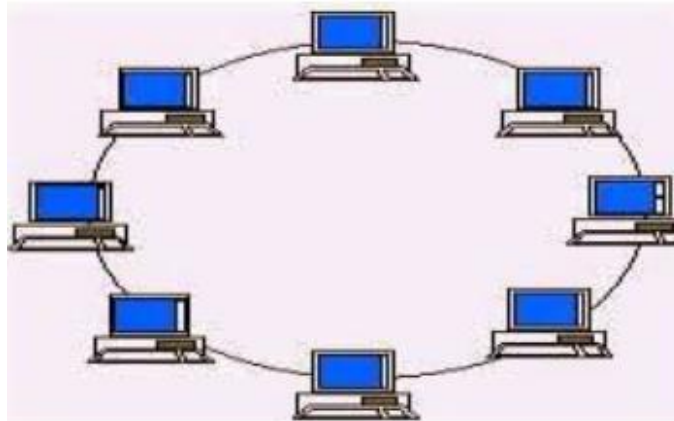


Gambar 2. 4 Contoh Topologi Bus (Haerudin et al., 2017)

Gambar 2.4 di atas memperlihatkan contoh dari topologi bus. Kelebihan dari topologi bus yaitu pada topologi ini tidak diperlukan banyak kabel sehingga menghemat biaya instalasi. Sedangkan kekurangan dari topologi bus yaitu apabila terjadi gangguan terhadap satu komputer akan mengganggu jaringan komputer lain.

2.6.2. Topologi Ring

Topologi ring merupakan suatu topologi yang menghubungkan satu komputer dengan komputer lain dan seterusnya hingga kembali ke komputer pertama hingga membentuk lingkaran (Haerudin et al., 2017). Pada topologi ini setiap perangkat akan terhubung secara langsung dengan dua perangkat lain sehingga setiap node akan menggunakan 2 kabel. Adapun contoh dari topologi ring tertera pada gambar 2.5.

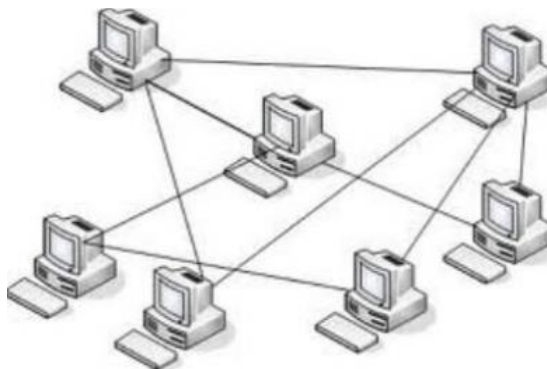


Gambar 2. 5 Contoh Topologi Ring (Haerudin et al., 2017)

Gambar 2.5 di atas memperlihatkan contoh dari topologi ring. Keunggulan yang dimiliki topologi ring adalah kemudahan dalam melakukan instalasi jaringan tersebut dan penggunaan kabel yang relatif sedikit. Sedangkan kekurangan pada topologi ini yaitu apabila terjadi kerusakan pada salah satu komputer ataupun kabel, akan mengganggu pengiriman data.

2.6.3. Topologi Mesh

Topologi mesh merupakan suatu jenis topologi yang menghubungkan setiap perangkat pada suatu jaringan komputer secara penuh. Dengan diterapkannya topologi mesh, kerusakan pada suatu perangkat tidak akan mempengaruhi perangkat lain atau jaringan secara keseluruhan. Adapun contoh dari topologi mesh tertera pada gambar 2.6.

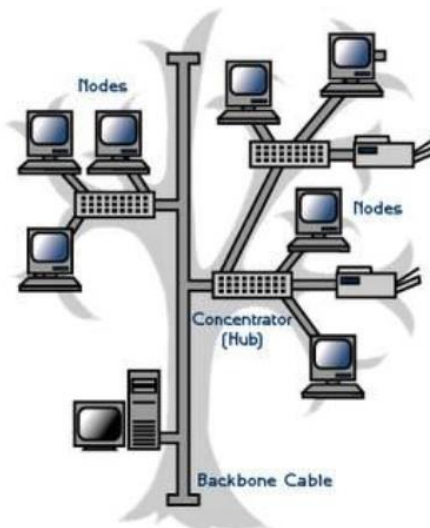


Gambar 2. 6 Contoh Topologi Mesh (Haerudin et al., 2017)

Gambar 2.6 di atas memperlihatkan contoh dari topologi mesh. Kelebihan dari topologi mesh yaitu pada topologi ini kerusakan pada salah satu komputer tidak akan mempengaruhi komputer lain atau jaringan secara keseluruhan. Sedangkan kekurangan pada topologi mesh yaitu topologi ini membutuhkan banyak sekali kabel sehingga biaya yang diperlukan penerapan topologi ini cukup banyak. Selain itu proses instalasi pada topologi mesh ini sangat rumit (Haerudin et al., 2017).

2.6.4. Topologi Tree

Topologi tree merupakan suatu jenis topologi yang terdiri dari topologi star yang terhubung dengan topologi bus, sehingga suatu topologi star akan terhubung dengan topologi star lainnya melalui topologi bus (Haerudin et al., 2017). Adapun contoh dari topologi tree tertera pada gambar 2.7.

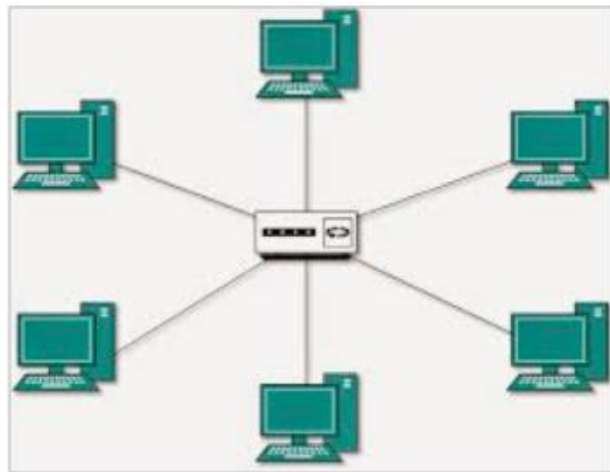


Gambar 2. 7 Contoh Topologi Tree

Gambar 2.7 di atas memperlihatkan contoh dari topologi tree. Kelebihan dari topologi tree yakni topologi tree sangat mudah untuk dikembangkan. Sedangkan kekurangan dari topologi tree yakni topologi ini memerlukan kabel yang cukup banyak sehingga memakan banyak biaya.

2.6.5. Topologi Star

Topologi Star merupakan topologi yang paling banyak diterapkan dalam pembentukan jaringan komputer. Topologi *star* merupakan topologi yang menghubungkan semua komputer dengan konsentrator (Armanto, 2017). Konsentrator pada topologi *star* dapat berupa *hub*, *switch* ataupun *router*. Adapun contoh dari topologi star tertera pada gambar 2.8.



Gambar 2. 8 Contoh Topologi Star (Simanullang, Napitupulu, Jamaluddin, & Purba, 2018)

Gambar 2.8 memperlihatkan contoh dari desain topologi *star*. Dengan terhubungnya setiap komputer dengan node pusat / konsentrator, maka setiap pengiriman data akan melalui konsentrator tersebut terlebih dahulu. Kemudian data tersebut akan dikirimkan secara langsung oleh konsentrator ke node tujuan. Adapun kelebihan topologi *star* antara lain (Simanullang et al., 2018):

- a. Identifikasi komputer yang mengalami gangguan sangat mudah dilakukan.
- b. Penambahan atau pengurangan komputer pada topologi ini sangat mudah dilakukan tanpa harus mengganggu komputer lain yang telah terhubung.
- c. Keamanan suatu data yang ditransmisikan lebih tinggi.

Sedangkan kekurangan dari topologi star yakni topologi ini sangat bergantung pada central node. Jika central node mengalami gangguan atau kerusakan, maka seluruh jaringan komputer pada setiap *device* yang terhubung akan mengalami gangguan.

2.7. Media Transmisi

Pada jaringan komputer, terdapat dua jenis media transmisi yakni kabel (*wired*) dan nirkabel (*wireless*). Pada perancangan jaringan komputer dengan menggunakan media transmisi kabel, jenis kabel yang sering digunakan yaitu kabel *Unshielded Twisted Pair* (UTP) dan *Shielded Twisted Pair* (STP). Kedua kabel tersebut dapat digunakan untuk menghubungkan komputer dengan perangkat jaringan lain seperti *switch*, *hub* dan *router*.

2.7.1. Kabel *Unshielded Twisted Pair* (UTP)

Kabel *Unshielded Twisted Pair* (UTP) merupakan jenis kabel yang terdiri dari empat pasang kabel yang dipilin (*twisted*) dan tidak memiliki pelindung (*unshielded*) (Rajagukguk, Hardani, & Haryono, 2020). Adapun contoh dari kabel UTP tertera pada gambar 2.9.

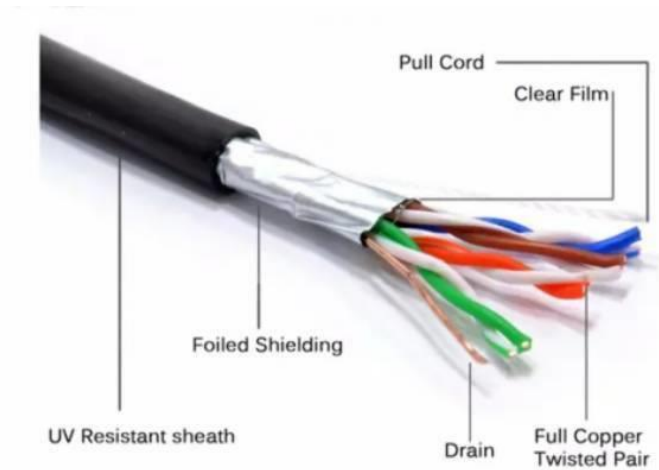


Gambar 2. 9 Kabel *Unshielded Twisted Pair* (UTP) (Sora, 2015)

Gambar 2.9 di atas merupakan contoh kabel UTP. Kabel UTP biasanya digunakan untuk menghubungkan komputer dengan perangkat jaringan lainnya seperti *switch*, *hub*, *router* dan komputer lain. Kabel UTP digunakan untuk jaringan komputer skala kecil (LAN), karena jarak jangkauannya hanya 100 meter. Kabel UTP biasanya dihubungkan ke perangkat jaringan lainnya menggunakan konektor RJ-45.

2.7.2. Kabel *Shielded Twisted Pair* (STP)

Kabel *Shielded Twisted Pair* (STP) merupakan kabel yang memiliki fungsi yang sama dengan kabel UTP, hanya saja kabel STP memiliki pembungkus kabel ganda. Adapun contoh kabel STP tertera pada gambar 2.10.



Gambar 2. 10 Kabel *Shielded Twisted Pair* (Mail, 2020)

Gambar 2.10 di atas merupakan contoh gambar dari kabel STP. Berbeda dengan kabel UTP, kabel STP memiliki pembungkus kabel ganda sehingga lebih cocok diimplementasikan di luar gedung (*outdoor*) dalam pembangunan jaringan. Sedangkan kabel UTP biasanya digunakan untuk pembangunan jaringan dalam gedung (*indoor*) (Nugroho & Kurniawan, 2018).

2.8. Router

Pada jaringan komputer, perangkat keras yang memfasilitasi transmisi paket data adalah *router* (Rizal, 2019). Adapun contoh *router* tertera pada gambar 2.11.



Gambar 2. 11 Router (Kurniawan, 2020)

Gambar 2.11 di atas memperlihatkan contoh dari perangkat keras *router*. *Router* memiliki fungsi sebagai penghubung antara dua jaringan atau lebih dan meneruskan paket data dari suatu jaringan ke jaringan tujuan.

2.9. Switch

Switch merupakan salah satu perangkat pada jaringan komputer yang berfungsi untuk menghubungkan beberapa komponen jaringan komputer agar dapat berkomunikasi. Adapun contoh dari *switch* tertera pada gambar 2.12.



Gambar 2. 12 Contoh Switch (Peniarsih, 2020)

Gambar 2.12 di atas merupakan contoh dari perangkat keras *switch*. *Switch* meneruskan paket data yang dikirim dari suatu perangkat jaringan ke perangkat lainnya. Cara kerja *switch* yakni dengan mempelajari alamat *host* tujuan, dengan demikian informasi tersebut dapat dikirim secara langsung ke *host* tujuan (Fauzi & Desmulyati, 2020).

2.10. Network Interface Card (NIC)

Network Interface Card (NIC) merupakan sebuah perangkat yang menjembatani komputer agar dapat terhubung ke suatu jaringan. NIC dapat menghubungkan komputer melalui kabel (*wired*) ataupun nirkabel (*wireless*)

tergantung jenis NIC yang digunakan. Adapun contoh NIC tertera pada gambar 2.13.



Gambar 2. 13 *Network Interface Card* (Peniarsih, 2020)

Gambar 2.13 di atas memperlihatkan contoh dari *Network Interface Card* (NIC). NIC memiliki fungsi untuk melakukan pengubahan aliran data paralel dalam bus komputer ke dalam bentuk data serial agar dapat ditransmisikan pada media jaringan (Peniarsih, 2020).

2.11. IP (Internet Protocol) Address

IP (*Internet Protocol*) *Address* adalah suatu proses pemberian alamat yang berupa sederet angka yang diberikan kepada perangkat yang terhubung ke jaringan komputer seperti komputer, *router* dan perangkat lainnya (Anshori, 2019). Saat ini versi IP *address* yang digunakan adalah IPv4, di mana pada IPv4 terdiri dari 4 oktet (32 bit) dan setiap oktet (8 bit) dipisahkan oleh titik (dot). Angka yang IPv4 diklasifikasikan ke dalam lima kelas. Adapun menurut Hasrul dan Lawani klasifikasi kelas pada IPv4 sebagai berikut (Hasrul & Lawani, 2017):

- a. Kelas A, yaitu IP *address* yang terdiri dari 1 oktet pertama sebagai *network* id dan 3 oktet lainnya sebagai *host* id. *Range* oktet pertama pada IP *address* kelas A yaitu 0 sampai dengan 127.
- b. Kelas B, yaitu IP *address* yang terdiri dari 2 oktet pertama sebagai *network* id dan 2 oktet lainnya sebagai *host* id. *Range* oktet pertama pada IP *address* kelas B yaitu 128 sampai dengan 191.
- c. Kelas C, yaitu IP *address* yang terdiri dari 3 oktet pertama sebagai *network* id dan 1 oktet lainnya sebagai *host* id. *Range* oktet pertama pada IP *address* kelas C yaitu 192 sampai dengan 223.

- d. Kelas D, yaitu *IP address* yang digunakan untuk *multicasting*. *Range* oktet pertama pada *IP address* kelas D yaitu 224 sampai dengan 239.
- e. Kelas E, yaitu *IP address* yang digunakan untuk penelitian atau eksperimen. *Range* oktet pertama pada *IP address* kelas E yaitu 240 sampai dengan 255.

2.12. Web Server

Untuk dapat mengoperasikan dan mengakses suatu *web application* pada web browser, dibutuhkan sebuah web server. Kurniawan dalam Muzawi dkk menyebutkan bahwa “Web Server adalah sebuah perangkat lunak yang berfungsi menerima permintaan HTTP atau HTTPS dari klien yang dikenal sebagai web browser dan mengirimkan hasilnya dalam halaman-halaman web yang umumnya berbentuk dokumen HTML” (Muzawi, Efendi, & Agustin, 2018). Untuk membuat *web application* dapat diakses dan dioperasikan, perlu dilakukan konfigurasi pada web server. Terdapat beberapa jenis web server saat ini, mulai dari *Apache* HTTP Server, *Nginx*, dan lain sebagainya. Web server akan menampilkan respons dari permintaan *user* pada web browser dalam berbagai bentuk seperti teks, gambar, video, audio dan lain sebagainya.

2.12.1. Apache HTTP Server

Apache HTTP Server atau yang diketahui juga dengan sebutan *httpd* merupakan salah satu web server yang paling terkenal. Apache HTTP Server merupakan web server yang dapat digunakan di berbagai platform (lintas platform) seperti Linux, Windows dan MacOS (Liu, Xu, Wang, & Zhang, 2018). Apache HTTP Server merupakan *unix-based* web server yang digunakan lebih dari 42% yang menjadikan Apache HTTP server sebagai web server yang paling banyak digunakan dan paling populer di berbagai domain di internet (Chandra, 2019).

2.12.2. Nginx

Selain Apache HTTP Server, pada penelitian ini juga digunakan Nginx sebagai web server. *Nginx* (di baca *Engine X*) merupakan web server yang lintas

platform dan bersifat *open source*. Selain fungsinya sebagai web server dengan penggunaan memori yang lebih rendah, *Nginx* juga menyediakan fitur lain seperti *reverse proxy*, *IPv6*, *load balancing*, *FastCGI support*, *web socket*, *handling static files* dan *SSL/TLS* (Chandra, 2019). *Nginx* juga merupakan web server peringkat ke dua sebagai web server yang paling banyak digunakan di berbagai situs (Liu et al., 2018).

2.13. ModSecurity

ModSecurity merupakan sebuah *Web Application Firewall* yang dapat digunakan untuk lintas platform dan bersifat *open source*. *ModSecurity* merupakan *Web Application Firewall* yang dapat digunakan untuk *monitoring* web application secara *real-time*, *logging* dan *access control* (Betarte, Gimenez, Martinez, & Pardo, 2019). *ModSecurity* memfilter *request* yang diminta oleh *client*, jika *request* tidak terdeteksi sebagai serangan *modsecurity* akan meneruskan *request* tersebut ke server. Sebaliknya jika *request* tersebut terdeteksi sebagai serangan, *ModSecurity* akan memblokir *request* tersebut tanpa diteruskan ke server. Adapun kelebihan dari *ModSecurity* antara lain sebagai berikut (Riska & Alamsyah, 2021):

1. *ModSecurity* melakukan pemeriksaan menyeluruh terhadap *request* yang ditujukan ke server termasuk isi *request* dan memberikan respons terhadap hasil pemeriksaan *request* tersebut.
2. *ModSecurity* memiliki *rules* berdasarkan aturan *regular expression*, sehingga *rules ModSecurity* lebih fleksibel.
3. *ModSecurity* melakukan pemeriksaan terhadap berkas yang diunggah.
4. *ModSecurity* memiliki validasi *real-time* dan juga perlindungan *buffer overflow*.

2.14. Fail2ban

Fail2ban merupakan perangkat lunak yang memiliki fungsi untuk membatasi akses dari suatu *IP address* berdasarkan *rules* yang dibuat. *Fail2ban* akan melakukan pemblokiran terhadap *IP address* yang tercatat dalam sebuah log dalam kurun waktu tertentu. Menurut *Elingwood* dalam Prasetyo dkk *fail2ban*

cara kerja dari fail2ban yaitu dengan menambahkan konfigurasi *firewall* dengan konfigurasi yang ada pada fail2ban, sehingga ketika berjalan fail2ban akan mengambil alih fungsi *firewall* yang ada pada server (Prasetyo, Idhom, & Wahanani, 2020).

2.15. Ubuntu

Ubuntu merupakan salah satu distro yang ada pada sistem operasi Linux yang bersifat *freeware*. Ubuntu tersedia dalam dua versi yakni server dan desktop. Ubuntu dalam versi desktop menyediakan tampilan antarmuka (*user interface*) berbasis *Graphic User Interface* (GUI) meskipun dapat digunakan dalam *Command Line Interface* (CLI) melalui terminal. Sedangkan Ubuntu Server hanya menyediakan tampilan antarmuka dalam *Command Line Interface* untuk mengoperasikannya. Ubuntu merupakan distro Linux yang paling banyak digunakan di seluruh dunia, terutama oleh developer dan *computer engineer* (Widodo et al., 2019).

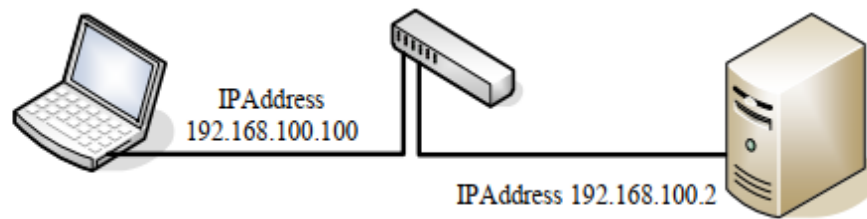
2.16. Slack

Slack adalah aplikasi digital *workspace* yang digunakan untuk berkolaborasi dan berdiskusi dalam satu platform. Slack merupakan aplikasi lintas platform yang tidak hanya tersedia dalam versi aplikasi desktop seperti Linux, Windows dan MacOS, namun juga pada *mobile application* seperti android dan iOS bahkan slack juga dapat diakses melalui web browser. Slack telah digunakan lebih dari 50.000 perusahaan di seluruh dunia, termasuk 43 perusahaan yang ada pada *Fortune* 100 (Teckchandani, 2018).

2.17. Contoh Implementasi Keamanan pada Web Server

Sebuah web server bekerja dengan merespons setiap HTTP *request* dari *client*. *Client* akan melakukan *request* sumber daya yang ada pada server melalui *tools* seperti web browser. Kemudian server akan mendengar respons tersebut melalui *port* HTTP dan meresponsnya kepada browser dan memprosesnya menjadi halaman web. Tanpa adanya *firewall*, web server akan melakukan respons terhadap setiap *request* oleh *client*. Berikut ini merupakan contoh dari

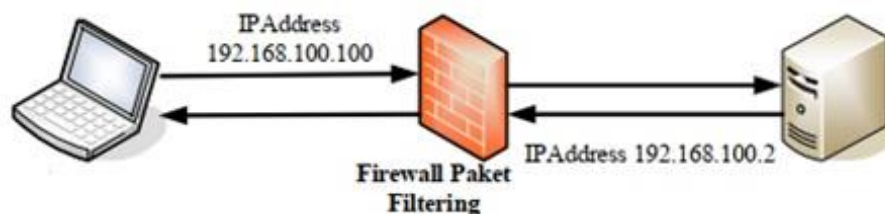
topologi jaringan komputer sederhana tanpa diterapkannya *firewall* tertera pada gambar 2.14.



Gambar 2. 14 Contoh Topologi Jaringan Tanpa *Firewall* (Sahren, 2021)

Gambar 2.14 di atas memperlihatkan contoh topologi jaringan komputer tanpa *firewall*. Pada topologi di atas terdapat beberapa *device* yang digunakan yakni 1 buah laptop, 1 buah switch dan 1 buah server. Pada gambar topologi di atas, dapat dilihat bahwa koneksi antara laptop dan server dilakukan secara langsung tanpa melalui dinding keamanan terlebih dahulu, sehingga tidak adanya penyaring terhadap lalu lintas data yang lewat (Sahren, 2021). Dengan tidak adanya penyaring terhadap lalu lintas data, server akan merespons setiap *request client* termasuk *request* yang berbahaya.

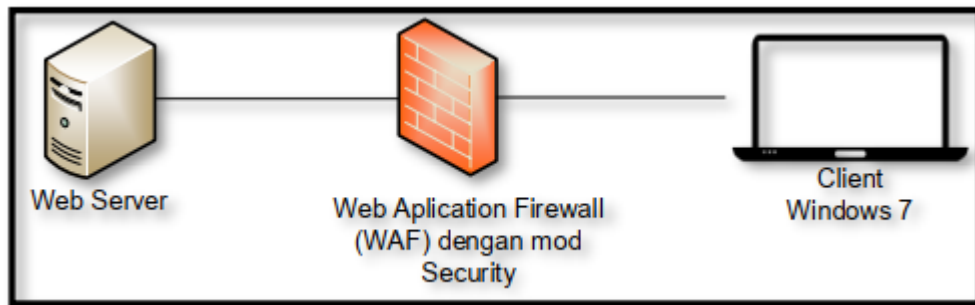
Salah satu upaya yang dapat dilakukan untuk memfilter setiap request yang ditujukan terhadap server yaitu dengan menerapkan *firewall*. Dengan diterapkannya *firewall*, setiap *request* yang ditujukan kepada server akan difilter terlebih dahulu untuk dilakukan pengecekan. Berikut ini merupakan contoh dari *topologi* jaringan dengan menerapkan *firewall* tertera pada gambar 2.15.



Gambar 2. 15 Topologi Jaringan Dengan *Firewall* (Sahren, 2021)

Gambar 2.15 di atas memperlihatkan contoh dari topologi jaringan yang menerapkan *firewall*. *Firewall* akan memblokir *request* yang terindikasi sebagai jenis serangan. Setiap tindakan pemblokiran tersebut akan dicatat dalam bentuk log. Selain topologi jaringan di atas, pada penelitian yang dilakukan oleh Riska

dan Alamsyah dilakukan penerapan *Web Application Firewall ModSecurity*. Adapun topologi yang diterapkan pada penelitian tersebut tertera pada gambar 2.16.

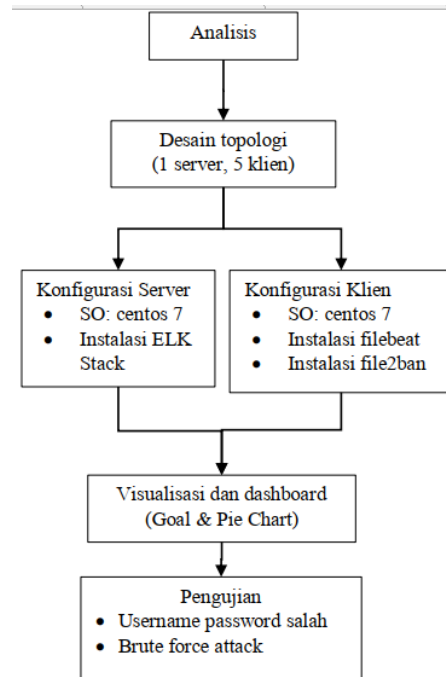


Gambar 2. 16 Topologi Jaringan Penerapan *ModSecurity* (Riska & Alamsyah, 2021)

Gambar 2.16 di atas merupakan topologi jaringan yang digunakan untuk menerapkan *Web Application Firewall ModSecurity*. Pada penelitian tersebut dilakukan 3 jenis uji coba serangan siber yaitu *SQL Injection*, *Cross Site Scripting* dan *Command Execution*. Hasil dari uji coba ketiga serangan tersebut menunjukkan *ModSecurity* berhasil melindungi web server dari serangan *SQL Injection*, *Cross Site Scripting* dan *Command Execution* (Riska & Alamsyah, 2021).

2.18. Metodologi Pengembangan

Untuk menerapkan *log event management*, diperlukan metode pengembangan sebagai pedoman dari tahapan dalam proses penelitian. Berikut ini merupakan contoh metodologi yang digunakan untuk menerapkan *log event management* tertera pada gambar 2.17.



Gambar 2. 17 Metode Pengembangan (Sholihah et al., 2020)

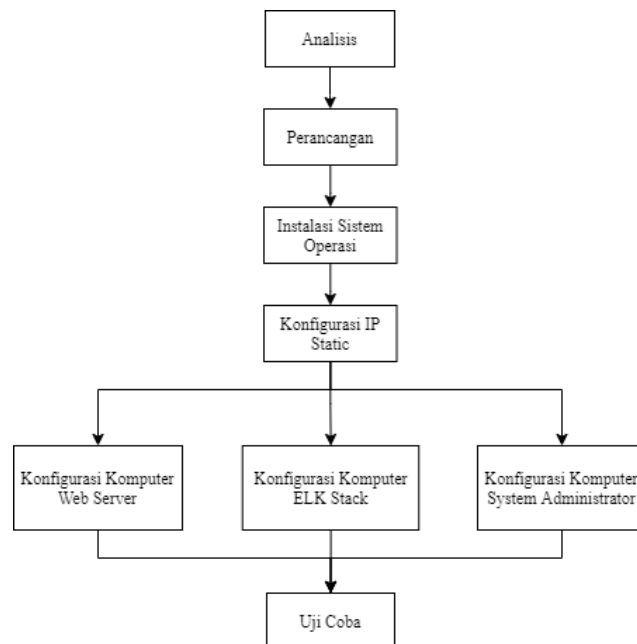
Gambar 2.17 di atas memperlihatkan contoh metodologi yang digunakan dalam penerapan log *event management*. Berikut ini merupakan penjelasan dari tahapan pada gambar di atas (Sholihah et al., 2020) :

1. Tahap analisis dilakukan untuk mengidentifikasi permasalahan yang ada serta solusi untuk menyelesaikannya. Selain itu pada tahap analisis ini juga dilakukan perancangan kebutuhan perangkat keras dan perangkat lunak yang digunakan.
2. Tahap desain topologi yang dilakukan untuk mendesain topologi jaringan untuk menyelesaikan permasalahan yang teridentifikasi pada proses analisis.
3. Tahap konfigurasi server dilakukan dengan melakukan instalasi dan konfigurasi perangkat lunak yang diperlukan baik pada server *ELK Stack* maupun server target uji coba.
4. Tahap visualisasi dan *dashboard* dilakukan untuk menentukan jenis dan informasi yang ditampilkan pada visualisasi yang ada pada *dashboard*.
5. Tahap pengujian dilakukan untuk mengetahui apakah log *file* yang telah dikirimkan dari server uji coba ke *ELK Stack* server dapat divisualisasikan.

BAB III METODOLOGI PENELITIAN

Web Application Firewall merupakan perangkat lunak yang berfungsi untuk melakukan filter terhadap setiap *request* dari *client* sebelum direspons oleh web server. Apabila terdapat *request* yang terindikasi sebagai bentuk serangan *Web Application Firewall* akan memblokir *request* tersebut dan mencatatnya pada *log file*. *Log event management* merupakan suatu cara untuk melakukan manajemen terhadap aktivitas-aktivitas yang dicatat pada suatu *log file*. Dengan melakukan *management* terhadap log pada sebuah web server, data-data yang tertulis pada *log file* akan diurai dan disimpan sehingga tersebut dapat digunakan untuk pemantauan aktivitas *user* ataupun sebagai dasar analisis keamanan pada web tersebut.

Adapun tahapan atau metodologi dalam penerapan *Log Event Management Web Application Firewall* adalah sebagai berikut :



Gambar 3. 1 Metodologi Penerapan *Log Event Management Web Application Firewall*

Gambar 3.1 di atas memperlihatkan metodologi yang digunakan dalam penerapan *Log Event Management Web Application Firewall*. Adapun penjelasan dari gambar di atas sebagai berikut :

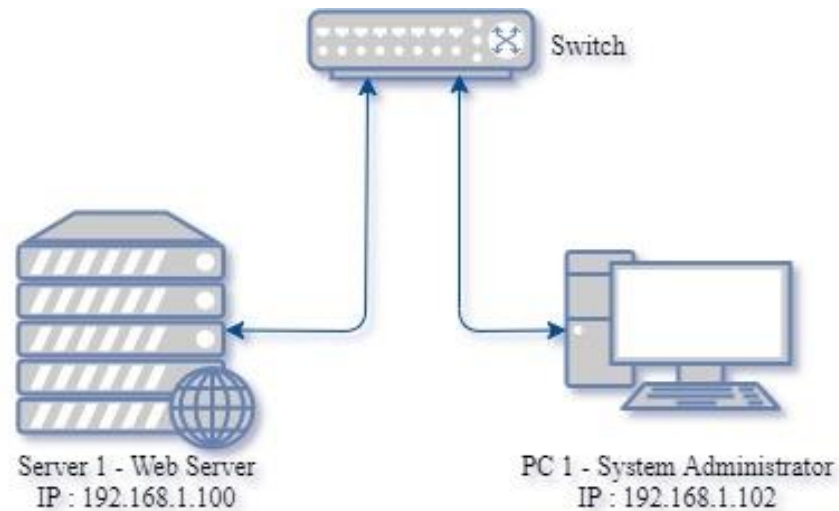
1. Tahap analisis, di mana pada tahap ini dilakukan dengan mempelajari topologi jaringan, spesifikasi perangkat keras dan perangkat lunak yang digunakan serta alur pengecekan log. Hal ini dilakukan untuk mengidentifikasi permasalahan yang ada sebelum diterapkannya *log event management web application firewall*.
2. Tahap perancangan, di mana pada tahap ini dilakukan perancangan topologi jaringan komputer, spesifikasi perangkat keras dan perangkat lunak untuk menerapkan *log event management web application firewall*. Selain itu dilakukan pembuatan alur *log event management*, penggunaan *IP address*, pembuatan *rules logstash* dan desain visualisasi. Pada tahap ini juga dibuat skenario uji coba sebagai gambaran pengujian yang nantinya akan dilakukan.
3. Tahap instalasi sistem operasi, di mana pada tahap ini dilakukan instalasi sistem operasi Ubuntu Server pada komputer ELK Stack server dan Web Server, serta ubuntu desktop pada komputer System Administrator.
4. Tahap konfigurasi, di mana pada tahap ini dilakukan instalasi dan konfigurasi pada komputer ELK Stack server, Web Server dan System Administrator untuk mengimplementasikan *log event management web application firewall*.
5. Tahap uji coba, di mana pada tahap ini dilakukan uji coba serangan pada web server. Serangan yang diuji cobakan yaitu *SQL Injection*, *Local File Inclusion* dan *Cross Site Scripting*. Uji coba akan dilakukan pada web server baik sebelum maupun sesudah penerapan *Log Event Management Web Application Firewall*.

3.1. Web Server Tanpa Log Event Management Web Application Firewall

Pada sub bab ini akan dilakukan analisis terhadap pengaplikasian web server tanpa *log event management* dan *web application firewall* seperti pada topologi jaringan, perangkat keras yang digunakan dan perangkat lunak yang terpasang.

3.1.1. Topologi Jaringan Komputer

Pada topologi jaringan ini, terdapat 1 buah server, 1 PC System Administrator dan 1 buah *switch*. Adapun topologi jaringan komputer tersebut tertera pada gambar 3.2.



Gambar 3. 2 Topologi Jaringan Komputer

Gambar 3.2 di atas memperlihatkan topologi jaringan komputer, adapun penjelasan dari gambar di atas antara lain :

1. Server 1 - Web Server dengan *hostname* *webservice* terhubung ke *switch* dengan IP Address 192.168.1.100.
2. *Switch* sebagai *central node* penghubung perangkat-perangkat yang ada.
3. *Personal Computer (PC)* 1 - System Administrator dengan *hostname* *sysadmin* terhubung ke *switch* dengan IP Address 192.168.1.102.

Berdasarkan gambar 3.1 di atas, dapat dilihat bahwa setiap *request* dari *client* akan langsung diarahkan kepada web server tanpa melewati proses filter terhadap *request* tersebut terlebih dahulu.

3.1.2. Spesifikasi Perangkat Keras

Berdasarkan topologi di atas, terdapat beberapa perangkat keras atau *hardware* yang digunakan seperti server, *personal computer* (PC), *switch* dan media transmisi untuk menghubungkan server dan PC dengan *switch*.

a. Komputer Web Server

Spesifikasi dari komputer yang digunakan sebagai web server tertera pada tabel 3.1.

Tabel 3. 1 Spesifikasi Perangkat Keras Komputer Web Server

No.	Nama Perangkat Keras	Keterangan
1	<i>Mainboard</i>	Lenovo SDK0E50510 Pro
2	<i>Processor</i>	Intel(R) Core(TM) i3-4000M CPU @ 2.40 GHz
3	<i>Memory</i>	4 GB DDR3
4	<i>Storage</i>	500GB HDD
5	<i>Network Adapter</i>	Inter(R) Ethernet Connection I217-V
6	<i>Disk Drive</i>	Optical Drive
7	<i>Power Supply</i>	External AC Adapter 65 W

Tabel 3.1 di atas menjelaskan komponen-komponen yang digunakan pada komputer yang digunakan untuk web server. Berdasarkan tabel di atas dapat diketahui spesifikasi secara rinci komputer web server seperti *processor*, *memory*, *storage* dan lainnya.

b. Komputer System Administrator

Selain komputer untuk web server, terdapat *Personal Computer* (PC) *System Administrator* yang digunakan untuk melakukan *maintenance*, konfigurasi dan pengecekan log yang ada pada web server. Adapun spesifikasi dari komputer *system administrator* tertera pada tabel 3.2.

Tabel 3. 2 Spesifikasi Perangkat Keras Komputer *System Administrator*

No.	Nama Perangkat Keras	Keterangan
1	<i>Mainboard</i>	Toshiba Portege R30-A

Lanjutan Tabel 3.2 Spesifikasi Perangkat Keras Komputer System Administrator

2	<i>Processor</i>	Intel(R) Core(TM) i5-4210M CPU @ 2.60 GHz
3	<i>Memory</i>	4 GB DDR3
4	<i>Storage</i>	500GB HDD
5	<i>Network Adapter</i>	Intel(R) <i>Ethernet Connection I217-V</i>
6	<i>Disk Drive</i>	<i>Optical Drive</i>
7	<i>Power Supply</i>	<i>External AC Adapter 45 W</i>

Tabel 3.2. di atas menjelaskan komponen-komponen yang digunakan pada komputer *system administrator*. Berdasarkan tabel di atas dapat diketahui spesifikasi secara rinci komputer web server seperti *processor*, *memory*, *storage* dan lainnya.

c. Media Penghubung dan Transmisi

Untuk menghubungkan setiap perangkat yang ada, digunakan *switch* sebagai *central node* dan kabel *Unshielded Twisted Pair (UTP)* sebagai media transmisi. Adapun spesifikasi dari *switch* dan kabel UTP yang digunakan tertera pada tabel 3.3.

Tabel 3. 3 Spesifikasi Media Penghubung dan Transmisi

No.	Nama Perangkat Keras	Keterangan
1	<i>Switch</i>	<ul style="list-style-type: none"> - TP-Link TL-SF 1005D - 5 RJ-45 <i>Ports (LAN)</i> - <i>Speed up to 100Mbps</i>
2	Kabel UTP	<ul style="list-style-type: none"> - <i>Category 5e</i> - Konektor RJ-45 - <i>Speed up to 100Mbps</i>

Tabel 3.3 di atas memperlihatkan spesifikasi dari media penghubung dan transmisi yang digunakan. *Switch* digunakan sebagai *central node* pada topologi *star* untuk menghubungkan beberapa perangkat lain seperti server dan PC. Untuk menghubungkan perangkat-perangkat yang ada dengan *switch*, dibutuhkan media transmisi. Pada topologi jaringan komputer ini digunakan media transmisi yakni kabel UTP. Kabel tersebut dikoneksikan pada *switch* dengan PC dan server menggunakan konektor RJ-45. Kabel ini memiliki kecepatan maksimum sebesar 100 Mbps.

3.1.3. Spesifikasi Perangkat Lunak

Terdapat beberapa perangkat lunak (*software*) yang digunakan untuk membangun web server dan melakukan monitor terhadap log yang terdapat pada server.

a. Komputer Web Server

Perangkat lunak yang digunakan pada komputer web server tertera pada tabel 3.4.

Tabel 3. 4 Perangkat Lunak Komputer Web Server

No.	Perangkat Lunak	Keterangan
1	Sistem Operasi	Linux Ubuntu Server 20.04 LTS Focal Fossa
2	Program Aplikasi	SSH Server, Apache, MySQL

Tabel 3.4 di atas memperlihatkan beberapa perangkat lunak yang digunakan pada komputer web server. Komputer web server menggunakan Linux Ubuntu Server 20.04 LTS *Focal Fossa* sebagai sistem operasi. Selain itu juga terdapat beberapa aplikasi yang digunakan seperti SSH Server untuk menyediakan *remote access* pada komputer web server. Selain itu juga terdapat aplikasi Apache sebagai web server dan MySQL sebagai *database*.

b. Komputer System Administrator

Perangkat lunak yang digunakan pada komputer *system* administrator tertera pada tabel 3.5.

Tabel 3. 5 Perangkat Lunak Komputer *System* Administrator

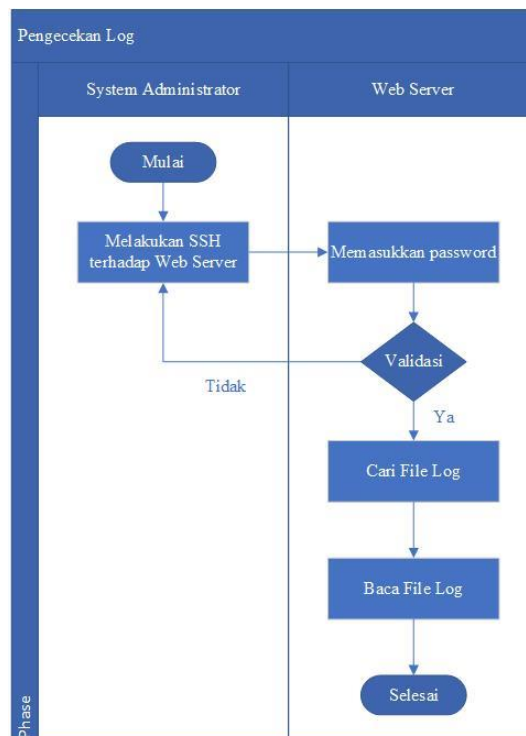
No.	Perangkat Lunak	Keterangan
1	Sistem Operasi	Linux Ubuntu Desktop 20.04 LTS Focal Fossa
2	Program Aplikasi	SSH <i>Client</i>

Tabel 3.5 di atas memperlihatkan aplikasi yang terdapat pada komputer *system* administrator. Komputer *system* administrator menggunakan Linux Ubuntu Desktop 20.04 LTS *Focal Fossa* sebagai sistem operasi. Selain itu terdapat SSH

client sebagai *tool* yang dapat digunakan untuk melakukan *access remote* terhadap web server.

3.1.4. Flow Map Pengecekan Log

Flow map alur dari langkah-langkah yang dilakukan untuk melakukan pengecekan terhadap log sebelum diterapkannya *log event management* dan *web application firewall* tertera pada gambar 3.3.



Gambar 3. 3 *Flow Map* Pengecekan Log

Gambar 3.3 di atas memperlihatkan alur untuk melakukan pengecekan log. Untuk melakukan pengecekan log seorang *system administrator* diharuskan melakukan *remote access* terhadap web server dengan menggunakan SSH. Setelah berhasil masuk ke dalam server, *system administrator* diharuskan membuka direktori tempat menyimpan log kemudian membacanya dan menganalisisnya. Adapun contoh dari log yang dibaca tertera pada gambar 3.4.

```

192.168.1.4 - - [13/Apr/2021:07:52:50 +0000] "GET / HTTP/1.1" 200 3477 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:87.0) Gecko/20100101 Firefox/87.0"
192.168.1.4 - - [13/Apr/2021:07:52:50 +0000] "GET /icons/ubuntu-logo.png HTTP/1.1" 200 3623 "http://192.168.1.101/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:87.0) Gecko/20100101 Firefox/87.0"
192.168.1.4 - - [13/Apr/2021:07:52:50 +0000] "GET /favicon.ico HTTP/1.1" 404 491 "http://192.168.1.101/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:87.0) Gecko/20100101 Firefox/87.0"
192.168.1.4 - - [13/Apr/2021:07:53:04 +0000] "GET /jsdhsjdjja HTTP/1.1" 404 492 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:87.0) Gecko/20100101 Firefox/87.0"
192.168.1.4 - - [13/Apr/2021:07:53:10 +0000] "GET /jsdhsjdjja HTTP/1.1" 404 492 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:87.0) Gecko/20100101 Firefox/87.0"
192.168.1.4 - - [13/Apr/2021:07:54:00 +0000] "GET /script.php?page=../../../../../../../../etc/passwd HTTP/1.1" 403 495 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:87.0) Gecko/20100101 Firefox/87.0"
192.168.1.4 - - [13/Apr/2021:07:59:04 +0000] "GET /script.php?page=../../../../../../../../etc/passwd HTTP/1.1" 404 492 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:87.0) Gecko/20100101 Firefox/87.0"
192.168.1.4 - - [13/Apr/2021:07:59:04 +0000] "GET /favicon.ico HTTP/1.1" 404 491 "http://192.168.1.101/script.php?page=../../../../../../../../etc/passwd" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:87.0) Gecko/20100101 Firefox/87.0"
192.168.1.4 - - [13/Apr/2021:07:59:08 +0000] "GET /script.php?page=../../../../../../../../etc/passwd HTTP/1.1" 404 492 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:87.0) Gecko/20100101 Firefox/87.0"
192.168.1.4 - - [13/Apr/2021:07:59:08 +0000] "GET /script.php?page=../../../../../../../../etc/passwd HTTP/1.1" 404 492 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:87.0) Gecko/20100101 Firefox/87.0"
192.168.1.4 - - [13/Apr/2021:07:59:34 +0000] "GET /shdsdhags HTTP/1.1" 404 492 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:87.0) Gecko/20100101 Firefox/87.0"
192.168.1.4 - - [13/Apr/2021:08:03:24 +0000] "GET /shdsdhags HTTP/1.1" 404 492 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:87.0) Gecko/20100101 Firefox/87.0"

```

Gambar 3. 4 Contoh Log pada *Command Line Interface* (CLI)

Gambar 3.4 di atas memperlihatkan log yang ada pada web server dalam bentuk teks. Hal ini dikarenakan pengaksesan server melalui *Command Line Interface* (CLI) sehingga seluruh interaksi antara *system* administrator dan server dilakukan dalam bentuk teks. Dengan ditampilkan dalam bentuk teks, server yang berjalan selama 24 jam non stop akan menghasilkan log yang sangat banyak sehingga log akan menumpuk dan sulit untuk dilakukan analisis. Selain itu, tidak adanya notifikasi terhadap serangan siber yang terjadi. Hal ini mengakibatkan serangan siber terlambat untuk diketahui dan ditangani.

3.2. Analisis Permasalahan Web Server tanpa Log Event

Management Web Application Firewall

Log *event management* memiliki peran yang sangat penting dalam keamanan jaringan. Dengan manajemen log, *system* administrator dapat memonitor setiap kejadian yang terjadi pada suatu server. Dengan demikian apabila terdapat ancaman terhadap suatu sistem yang terdeteksi oleh *firewall*, dapat dicegah sebelum terjadi insiden peretasan. Selain itu dengan manajemen log, apabila terjadi suatu peretasan dan log yang ada pada server dihapus, log masih dapat dilihat pada server manajemen log, sehingga jejak dari *attacker* tersebut dapat digunakan sebagai panduan untuk memperbaiki sistem tersebut.

Berikut ini merupakan permasalahan yang dapat diidentifikasi terhadap web server tanpa log *event management web application firewall* antara lain :

1. Aplikasi berbasis web sangat rawan terhadap serangan siber yang dilakukan oleh *attacker*. Hal ini dikarenakan seluruh *request* dari *client* terhadap web server akan langsung direspons oleh web server tanpa

dilakukan filter terhadap *request* tersebut terlebih dahulu. Hal ini memungkinkan terjadinya peretasan melalui *request* oleh *attacker* dengan memasukkan kode-kode tertentu pada *request* tersebut.

2. Aplikasi web server tidak dapat mendeteksi serangan siber, hal ini menyebabkan tidak adanya jejak pada log *file* terkait serangan siber yang diterima web server.
3. *Modsecurity* yang berjalan selama 24 jam tanpa henti akan menghasilkan banyak log. Hal ini mengakibatkan log yang dihasilkan akan menumpuk dan sulit untuk dilakukan analisis. Untuk membaca dan menganalisis log seorang *system* administrator diharuskan melakukan *remote access* terhadap server. Seluruh log akan ditampilkan dalam bentuk tulisan sehingga diperlukan pengetahuan khusus untuk dapat membaca log dan menganalisisnya.
4. Tidak terdapat notifikasi yang memberikan peringatan apabila ada indikasi upaya peretasan terhadap server, sehingga penanganan terhadap upaya peretasan tersebut kerap kali terlambat karena *system* administrator baru mengetahui upaya tersebut ketika membaca log yang ada pada server.

3.3. Perancangan Log Event Management Web Application

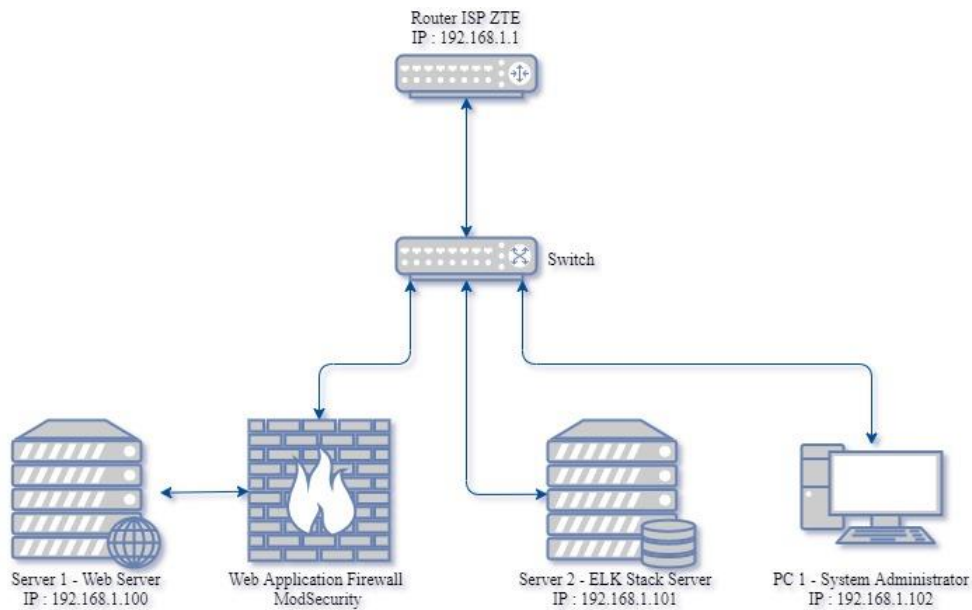
Firewall

Pada tahap ini akan dilakukan perancangan jaringan untuk membangun *log event management web application firewall* mulai dari topologi jaringan, perangkat keras dan perangkat lunak untuk menyelesaikan permasalahan di atas.

3.3.1. Perancangan Topologi Jaringan

Pada perancangan topologi jaringan untuk menerapkan *log event management web application firewall*, dilakukan penambahan *Web Application Firewall ModSecurity* pada komputer web server, 1 buah server yang berisi *Elasticsearch*, *Logstash* dan *Kibana* (*ELK Stack*) dan 1 buah *router* sebagai

penghubung setiap perangkat yang ada dengan jaringan internet. Adapun topologi jaringan tersebut tertera pada gambar 3.5.



Gambar 3. 5 Topologi Jaringan *Log Event Management Web Application Firewall*

Gambar 3.5 di atas memperlihatkan topologi jaringan yang digunakan untuk membangun *Log Event Management Web Application Firewall*. Adapun penjelasan dari gambar topologi jaringan di atas antara lain :

1. *Router* menggunakan *Internet Service Provider (ISP)* ZTE untuk akses internet dan sebagai penghubung pada jaringan *Local Area Network (LAN)*.
2. *Web Server* dengan *hostname webserver* terhubung ke *switch* dengan *IP Address* 192.168.1.100.
3. *Web Application Firewall ModSecurity* untuk memfilter setiap *request* yang ditujukan kepada *web server*.
4. *ELK Stack Server* dengan *hostname elkserver* terhubung ke *switch* dengan *IP Address* 192.168.1.101.
5. *Switch* sebagai *central node* penghubung perangkat-perangkat yang ada.
6. *Personal Computer (PC)* *System Administrator* terhubung ke *switch* dengan *IP Address* 192.168.1.102.

Pada gambar di atas terlihat diterapkannya *Web Application Firewall ModSecurity* dan *ELK Stack Server*. *Web Application Firewall ModSecurity* diterapkan untuk memfilter setiap *request* yang diajukan. Apabila request yang diajukan terindikasi sebagai jenis serangan, maka *ModSecurity* akan melakukan blok terhadap *request* tersebut tanpa diteruskan ke web server. Setiap blok yang dilakukan oleh *ModSecurity* akan tercatat pada log dari web server. Namun jika *request* tersebut tidak terindikasi sebagai jenis serangan, *ModSecurity* akan meneruskan *request* tersebut ke web server.

Sedangkan *ELK Stack Server* diterapkan untuk melakukan log *management*. Log yang dicatat oleh *ModSecurity* akan dikirimkan ke *ELK Stack Server* yang kemudian log tersebut akan difilter, disimpan dan divisualisasikan. Sehingga untuk melakukan analisis terhadap log, *system administrator* tidak perlu melakukan *access remote* terhadap web server.

3.3.2. Spesifikasi Perangkat Keras Tambahan

Pada penelitian ini, diperlukan beberapa tambahan untuk membangun log *event management web application firewall* yaitu 1 buah server dan 1 buah *router*. Pada sub bab ini akan dilakukan pembahasan mengenai spesifikasi dari kedua perangkat keras yang ditambahkan tersebut.

a. Komputer ELK Stack Server

Perangkat keras yang digunakan untuk *ELK Stack Server* tertera pada tabel 3.6.

Tabel 3. 6 Spesifikasi Perangkat Keras *ELK Stack Server*

No.	Nama Perangkat Keras	Keterangan
1	<i>Mainboard</i>	ASUSTek COMPUTER INC.
2	<i>Processor</i>	Intel(R) Core(TM) i5-7200U CPU @ 2.50 GHz
3	<i>Memory</i>	8 GB DDR4
4	<i>Storage</i>	1TB HDD
5	<i>Network Adapter</i>	Realtek PCIe GbE Family Controller
6	<i>Disk Drive</i>	<i>Optical Drive</i>
7	<i>Power Supply</i>	<i>External AC Adapter 65 W</i>

Tabel 3.6 di atas memperlihatkan spesifikasi komputer yang digunakan sebagai ELK *Stack Server*. Pada server ini, nantinya akan di-*install Elasticsearch, Logstash* dan *Kibana* untuk melakukan *log management*.

b. Router

Selain penambahan 1 buah server, terdapat juga penambahan 1 buah router. Adapun spesifikasi dari *router* tersebut tertera pada tabel 3.7.

Tabel 3. 7 Spesifikasi *Router*

No.	Nama Perangkat Keras	Keterangan
1	<i>Router ISP ZTE</i>	<ul style="list-style-type: none"> - 4 RJ-45 <i>Ports</i> (LAN) - 2 RJ-11 <i>Ports</i> (<i>Phone</i>) - 2.4 GHz <i>Wireless</i>

Tabel 3.7 di atas menjelaskan spesifikasi dari *router* yang digunakan pada sistem jaringan berjalan. *Router* berperan untuk menghubungkan *device-device* yang ada dengan jaringan internet.

3.3.3. Spesifikasi Perangkat Lunak Tambahan

Pada sub bab ini akan menjelaskan perangkat lunak yang dibutuhkan dalam membangun *log event management*. Selain perangkat lunak yang dibutuhkan untuk ELK *Stack Sever*, pada sub bab ini juga terdapat perangkat lunak pada web server dan komputer system administrator untuk mendukung penerapan *log event management*.

a. Komputer ELK *Stack Server*

Perangkat lunak yang digunakan pada ELK *Stack Server* tertera pada tabel 3.8.

Tabel 3. 8 Perangkat Lunak Komputer ELK *Stack Server*

No.	Perangkat Lunak	Keterangan
1	Sistem Operasi	Linux Ubuntu Server 20.04 LTS Focal Fossa
2	Program Aplikasi	SSH Server, Elasticsearch, Logstash, Kibana, Elastalert, Nginx

Tabel 3.8 di atas memperlihatkan perangkat lunak yang digunakan pada ELK *Stack* Server. SSH Server diperlukan agar komputer ELK *Stack* Server dapat diakses secara *remote*. *Elasticsearch* diterapkan untuk menyimpan log-log yang dikirim oleh Web Server dan disimpan dalam bentuk JSON. Sebelum log-log tersebut disimpan pada *Elasticsearch*, log-log tersebut akan difilter oleh *Logstash* sesuai dengan *rules* yang dibuat. Setelah itu, log-log tersebut dapat divisualisasikan dengan menggunakan *Kibana*.

Selain 3 komponen perangkat utama di atas, pada ELK *Stack* Server ini juga diterapkan *Elastalert* untuk memberikan notifikasi apabila ada serangan yang terdeteksi. Untuk menjalankan *Elastalert*, komputer ELK *Stack* Server diharuskan meng-*install python* terlebih dahulu. Selain *Elastalert*, digunakan juga *Nginx* untuk melakukan *reverse proxy* untuk mengizinkan akses eksternal pada *Kibana*.

b. Komputer Web Server

Untuk mengimplementasikan log *event management web application firewall*, diperlukan beberapa perangkat lunak tambahan pada komputer Web Server sebagaimana yang tertera pada tabel 3.9.

Tabel 3. 9 Perangkat Lunak Komputer Web Server

No.	Perangkat Lunak	Keterangan
1	Sistem Operasi	Linux Ubuntu Server 20.04 LTS Focal Fossa
2	Program Aplikasi	SSH Server, Apache, MySQL, ModSecurity, Filebeat, Fail2ban

Tabel 3.9 memperlihatkan perangkat lunak yang diperlukan komputer web server untuk membangun log *event management*. Untuk menunjang implementasi log *event management web application firewall*, digunakan tambahan perangkat lunak yakni *ModSecurity*, *Filebeat* dan *Fail2ban*. *ModSecurity* berfungsi sebagai *Web Application Firewall* yang akan melakukan filter terhadap setiap *request* dari *client* terhadap Web Server. Jika terdapat *request* yang terindikasi sebagai serangan, *ModSecurity* akan memblok *request* tersebut dan mencatatnya dalam

log. Kemudian *Filebeat* akan mengirim log tersebut ke *ELK Stack Server* yang nantinya akan diolah untuk disimpan dan divisualisasikan. Sedangkan *Fail2ban* berfungsi untuk memblokir *ip address* dari *attacker* dalam jangka waktu tertentu sesuai dengan waktu yang ditetapkan.

c. Komputer System Administrator

Penambahan perangkat lunak yang digunakan pada komputer *system administrator* tertera pada tabel 3.10.

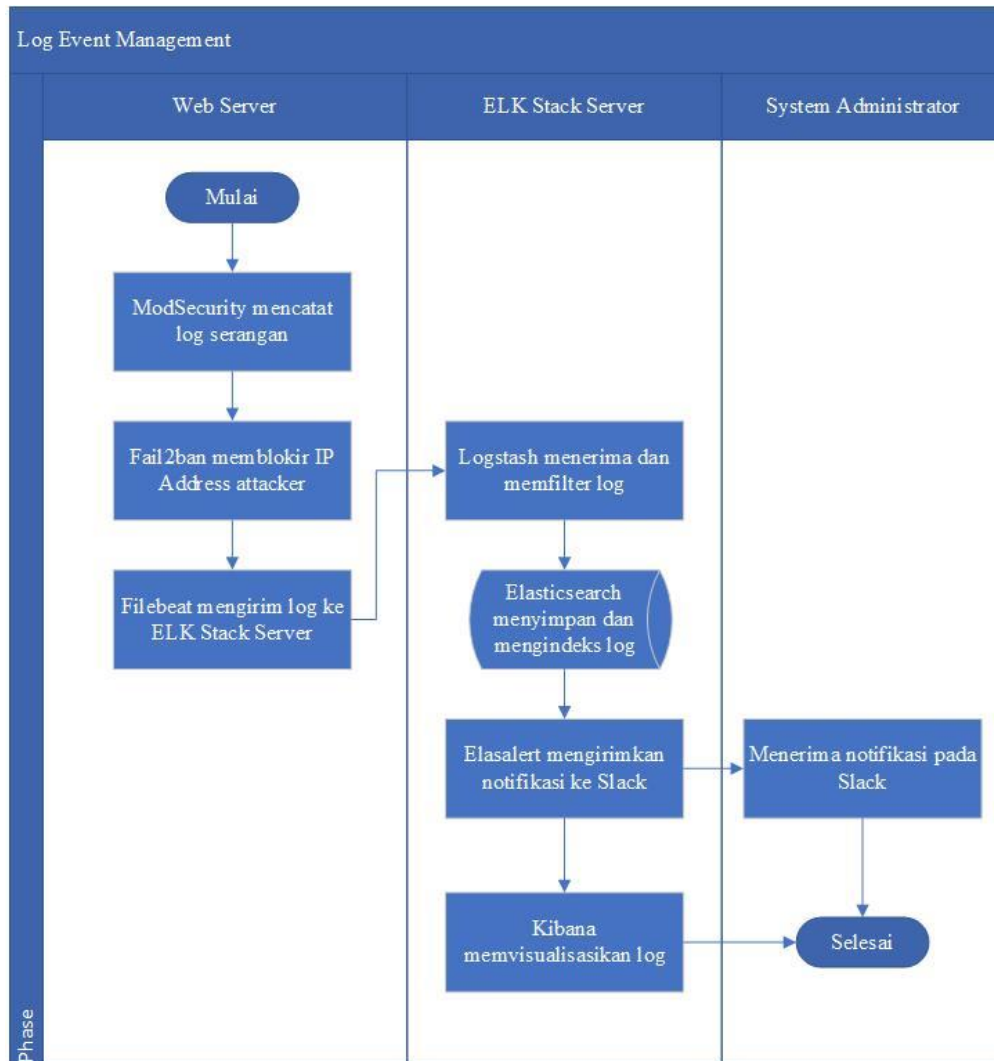
Tabel 3. 10 Perangkat Lunak Komputer *System Administrator*

No.	Perangkat Lunak	Keterangan
1	Sistem Operasi	Linux Ubuntu Desktop 20.04 LTS Focal Fossa
2	Program Aplikasi	SSH Client, Slack

Tabel 3.10 memperlihatkan perangkat lunak yang digunakan komputer *system administrator*. Pada komputer *system administrator* diberikan tambahan perangkat lunak *Slack* untuk mengirimkan notifikasi serangan. Sehingga seorang *system administrator* dapat melakukan *monitoring* keamanan web server dengan waktu yang mendekati *real time*.

3.3.4. Flow Map Log Event Management

Pada penelitian ini, dibuatkan *flow map* untuk menggambarkan alur kerja dari *log event management*. Untuk membuat *flow map* tersebut digunakan *tool Visio 2019*. Adapun *flow map* dari *log event management* pada gambar 3.6.



Gambar 3. 6 *Flow Map Log Event Management*

Gambar 3.6 di atas memperlihatkan alur dari log *event management*. Adapun penjelasan dari gambar di atas antara lain sebagai berikut :

1. *ModSecurity* melakukan filter terhadap setiap *request* yang menuju kepada web server.
 - a. Jika *request* tersebut terindikasi sebagai jenis serangan, maka *ModSecurity* akan memblokir *request* tersebut dan mencatat *event* tersebut pada log.
 - b. Jika *request* tidak terindikasi sebagai jenis serangan, *ModSecurity* akan meneruskan *request* tersebut ke web server.

2. Fail2ban akan memblokir *IP Address attacker* selama 60 menit, sehingga *attacker* untuk sementara tidak dapat mengakses *web application*.
3. *Filebeat* akan mengirimkan log yang dicatat *ModSecurity* kepada *ELK Stack Server*.
4. *Logstash* mengelola log yang dikirim *Filebeat* dalam 3 tahap yakni *input*, *filter* dan *output*. Pada tahap *input* *Logstash* akan menerima log yang dikirim *Filebeat* melalui *port* tertentu. Kemudian log akan difilter untuk membuang data yang tidak diperlukan. Terakhir pada tahap *output*, log akan dikirim ke *Elasticsearch* untuk disimpan dan dilakukan *indexing*.
5. *Elasticsearch* akan menyimpan log yang telah difilter oleh *Logstash* dalam bentuk *JSON*.
6. *Elasalert* akan mengirimkan notifikasi ke *channel Slack* yang telah didaftarkan.
7. *Kibana* akan memvisualisasikan log yang tersimpan dalam *Elasticsearch*, sehingga dapat dilihat dengan menggunakan *web browser*.

3.3.5. Penggunaan IP Address

Penggunaan *IP address* yang diterapkan pada pembuatan *log event management web application firewall* tertera pada tabel 3.11.

Tabel 3. 11 Penggunaan IP Address

No.	Hardware	Hostname	IP Address	Subnetmask	Gateway
1	Router ISP ZTE	-	192.168.1.1	255.255.255.0	-
2	Server 1 – Web Server	webserver	192.168.1.100	255.255.255.0	192.168.1.1
3	Server 2 – ELK Stack Server	elkserver	192.168.1.101	255.255.255.0	192.168.1.1
4	PC 1– System Administrator	sysadmin	192.168.1.102	255.255.255.0	192.168.1.1

Tabel 3.11 di atas memperlihatkan IP *address* yang digunakan pada pembuatan *log event management web application firewall*. Setiap perangkat di atas terhubung dengan *switch* sebagai *central node* dan kabel UTP sebagai media transmisi. *Router* terhubung dengan *switch* melalui *port 1 switch*, Web Server terhubung melalui *port 2 switch*, ELK Stack Server terhubung melalui *port 3 switch* dan *system administrator* terhubung melalui *port 4 switch*.

3.3.6. Pembuatan Rules Logstash

Pada ELK Stack Server, *rules* pada *Logstash* diperlukan untuk memfilter data yang dikirimkan dari Web Server sebelum disimpan pada *Elasticsearch*. Pada *rules Logstash* terdapat tahapan yakni *inputs*, *filters* dan *outputs* yang disebut *Logstash Pipeline*. Berikut ini merupakan *rules* yang akan digunakan pada tahap input :

```
input {
  beats {
    port => 5044
  }
}
```

Pada tahap *input*, *logstash* akan mencari sumber dari log yang akan difilter. Pada penelitian ini digunakan *tools filebeat* untuk mengirimkan log ke *logstash* melalui *port default logstash* yakni 5044. Setelah melalui tahap *input*, selanjutnya log akan melalui tahap filter untuk melakukan penguraian data. Pada penelitian ini digunakan *apache* sebagai web server, oleh karena itu serangan yang terdeteksi oleh *modsecurity* akan dicatat pada *apache error.log*.

Sebelum data disimpan pada *elasticsearch* dan divisualisasikan pada *kibana*, log yang diterima akan diurai menggunakan *grok* filter. *Grok* filter dapat melakukan *parsing* log data yang tidak terstruktur menjadi terstruktur dengan menggunakan *regular expression* (regex). Pada penelitian ini digunakan *Grok Debugger* yang ada pada *Kibana* untuk menguji *rules* yang dibuat pada *Logstash*. Berikut ini merupakan seluruh *rules* yang akan digunakan pada tahap filter :

```

filter{
  #Mengurai log lengkap ke dalam beberapa bagian yaitu time_stamp, log_level,
  ip_attacker dan modsec_message
  grok{
    match => { "message" =>
"(?<time_stamp>%{MONTH:bulan}\s%{MONTHDAY:tanggal}\s%{TIME:wak
tu}\s%{YEAR:tahun})\]
\[:\s%{LOGLEVEL:log_level}.*client\s%{IPORHOST:ip_attacker};\d+\s%{GR
EEDYDATA:modsec_message}" }
  }

  #Ekstrak attack_file dari modsec_message
  grok{
    match => { "modsec_message" => "(?<attack_file>file \"(.+).conf)" }
  }

  #Ekstrak attack_name dari attack_file
  grok{
    match => { "attack_file" => "(?<attack_name>[A-Z]+-[A-Z][^.]*)" }
  }

  #Ekstrak msg dari modsec_message
  grok{
    match => { "modsec_message" => "(?<msg>msg \"(.+?)\"" }
  }

  #Ekstrak alert_msg dari msg
  grok{
    match => { "msg" => "(?<alert_msg>\".+\"" }
  }

  #Ekstrak data dari modsec_message
  grok{
    match => { "modsec_message" => "(?<data>data \"(.+?)\"" }
  }

  #Ekstrak attack_data dari data
  grok{
    match => { "data" => "(?<attack_data>\\.:.*\\"" }
  }

  #Ekstrak uri dari modsec_message
  grok{
    match => { "modsec_message" => "(?<uri>uri \"(.+?)\"" }
  }

  #Ekstrak attack_uri dari uri
  grok{

```

```

match => { "uri" => "(?<attack_uri>\".*\")" }
}

#Ekstrak hostname dari modsec_maessage
grok{
  match => { "modsec_message" => "(hostname \"% {IPORHOST:dst_host})" }
}

#Hapus karakter yang tidak diperlukan dan tag yang tidak diperlukan
mutate{
  gsub => [
    "attack_uri", "[\\]", "",
    "attack_data", "[\\:]", "",
    "alert_msg", "[\\]", ""
  ]
  remove_field => ["modsec_message", "attack_file", "msg", "data", "uri",
"message"]
}
}

```

Setelah data terstruktur, tahap selanjutnya yaitu menyimpannya ke *elasticsearch*. Berikut ini merupakan *rules* yang akan digunakan pada tahap *output*:

```

output {
  elasticsearch {
    hosts => ["localhost:9200"]
    manage_template => false
    index => "logstash-waf-% {+YYYY.MM}"
  }
}

```

Pada penelitian ini log yang telah diurai akan disimpan di *elasticsearch* melalui *port default elasticsearch* yakni 9200. Setelah tersimpan di *elasticsearch*, log dapat divisualisasikan dengan menggunakan *kibana*.

3.3.7. Desain Visualisasi Log

Untuk mempermudah analisis terhadap serangan yang diterima web server, log dapat divisualisasikan dengan menggunakan *kibana*. Dengan menggunakan *kibana*, log dapat divisualisasikan dalam bentuk *bar chart*, *pie chart*, tabel dan lain sebagainya. Pada penelitian ini desain visualisasi log dibuat

untuk memberikan gambaran terhadap visualisasi log yang akan diterapkan dengan menggunakan *tool Balsamiq Mock-up*. Adapun desain visualisasi log tersebut tertera pada gambar 3.7.



Gambar 3.7 *Mock-up* Desain Dashboard Log Modsecurity

Gambar 3.7 di atas memperlihatkan *mock-up* desain visualisasi log *modsecurity* pada *dashboard kibana*. Dengan divisualisasikan log *modsecurity* pada *dashboard kibana*, *system administrator* dapat melakukan *monitoring* dan analisis celah keamanan berdasarkan log tersebut tanpa harus membacanya dalam bentuk tulisan pada *Command Line Interface (CLI)*.

3.3.8. Skenario Uji Coba

Skenario uji coba untuk *log event management web application firewall* ini yaitu dengan melakukan simulasi penyerangan terhadap web server. Terdapat 3 jenis serangan yang akan diuji cobakan pada web server antara lain *SQL Injection*, *Local File Inclusion (LFI)* dan *Cross Site Scripting (XSS)*. Setiap serangan dilakukan dengan memasukkan kode-kode tertentu pada URL website. Adapun daftar serangan yang akan diuji cobakan pada web server tertera pada tabel 3.12.

Tabel 3. 12 Skenario Uji Coba

No.	Nama Serangan	Bentuk Serangan
1	<i>Local File Inclusion</i>	http://targeturl?page=/etc/passwd
2	<i>Cross Site Scripting</i>	http://targeturl?cari=<script>alert(1)</script>
3	<i>SQL Injection</i>	http://targeturl?id=999 UNION SELECT 1,2,3,4

Tabel 3.12 memperlihatkan jenis serangan yang akan diujicobakan. Uji coba tersebut bertujuan untuk menguji apakah *Web Application Firewall* dan *ELK Stack* berfungsi dengan benar dan terintegrasi serta memberikan notifikasi apabila terjadi penyerangan terhadap web server.

BAB IV

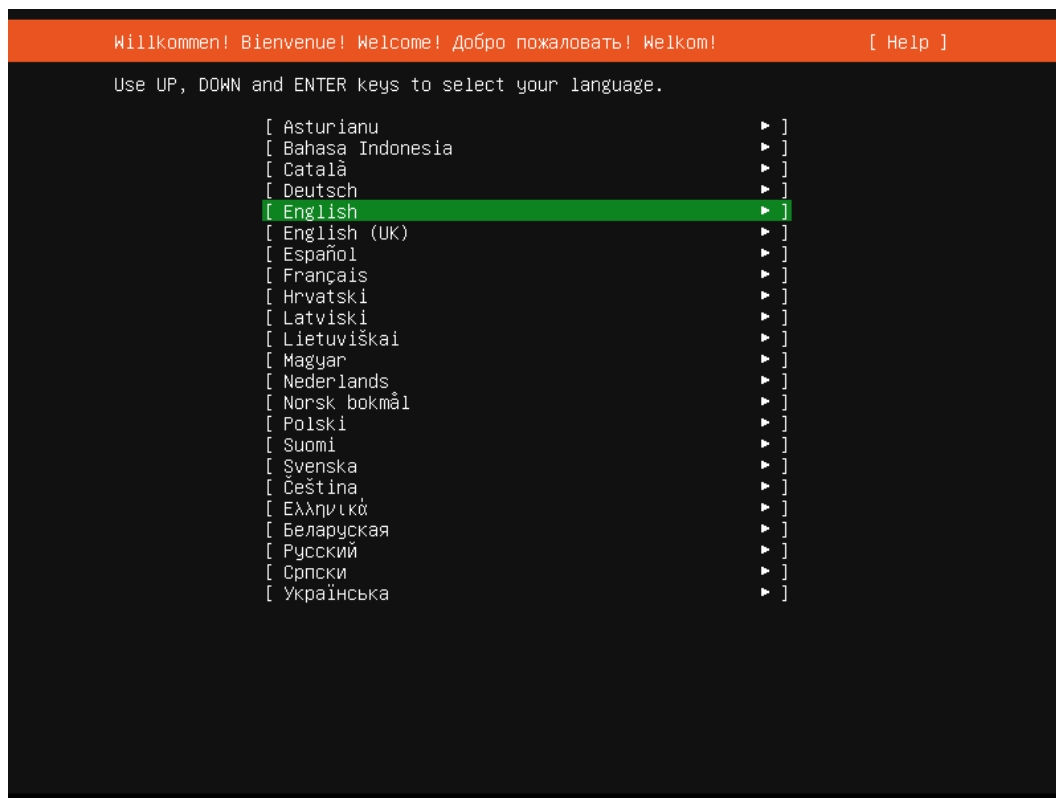
HASIL DAN PEMBAHASAN

Setelah melakukan perancangan perangkat keras dan perangkat lunak, langkah selanjutnya dilakukan implementasi *log event management web application firewall* dengan menggunakan *elasticsearch*, *logstash* dan *kibana* (ELK Stack). Untuk mengimplementasikan *log event management* tersebut langkah-langkah yang perlu dilakukan yaitu :

1. Instalasi Sistem Operasi Ubuntu Server
2. Instalasi Sistem Operasi Ubuntu Desktop
3. Konfigurasi IP *Static*
4. Konfigurasi Web Server
 - 4.1. Instalasi perangkat lunak pada Web Server
 - 4.2. Konfigurasi *Apache Web Server* dan MySQL
 - 4.3. Konfigurasi *Web Application Firewall*
 - 4.4. Konfigurasi *Filebeat*
 - 4.5. Konfigurasi *Fail2ban*
5. Konfigurasi Komputer *System Administrator*
6. Konfigurasi ELK Stack Server
 - 6.1. Instalasi perangkat lunak pada ELK Stack Server
 - 6.2. Konfigurasi *Elasticsearch*
 - 6.3. Pembuatan Filter *Logstash*
 - 6.4. Konfigurasi *Nginx*
 - 6.5. Konfigurasi *Kibana*
 - 6.6. Konfigurasi *Elastalert*
7. Pengujian *Log Event Management Web Application Firewall*.

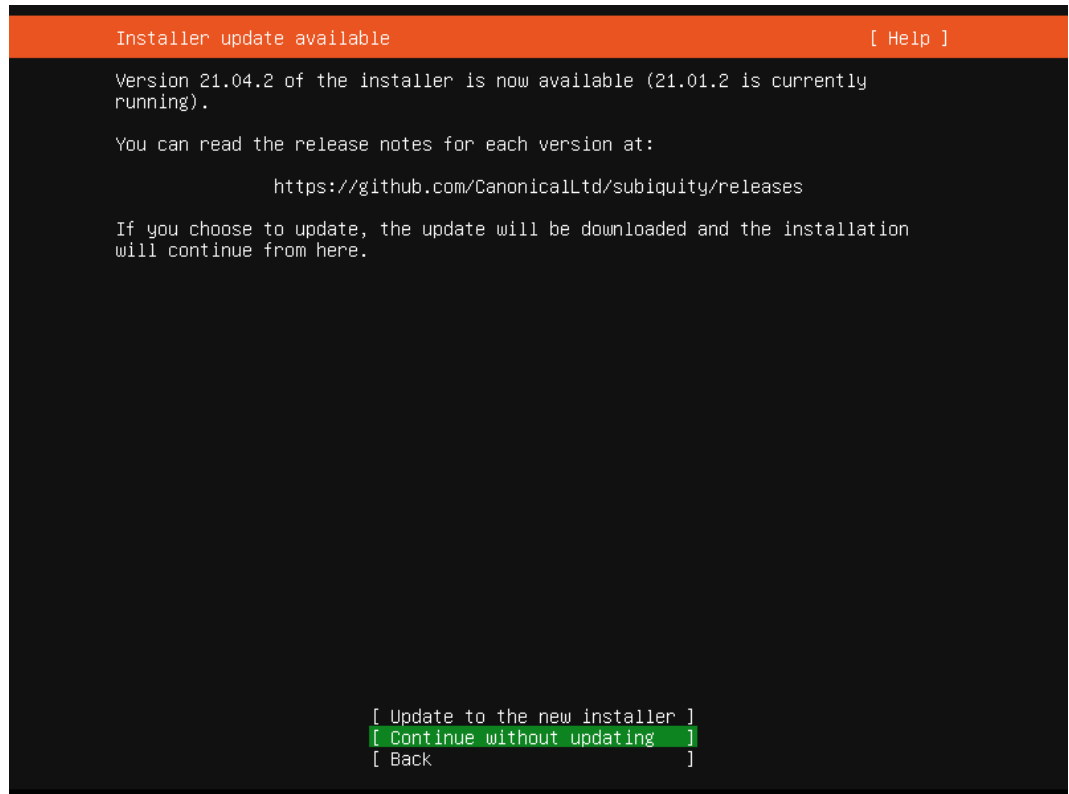
4.1. Instalasi Sistem Operasi Ubuntu Server

Pada penelitian ini, sistem operasi yang digunakan untuk Web Server dan ELK Stack Server adalah Ubuntu Server 20.04 LTS *Focal Fossa*. Untuk melakukan instalasi sistem operasi tersebut, langkah pertama yang dilakukan yaitu memilih bahasa yang akan digunakan sistem operasi sebagaimana yang tertera pada gambar 4.1.



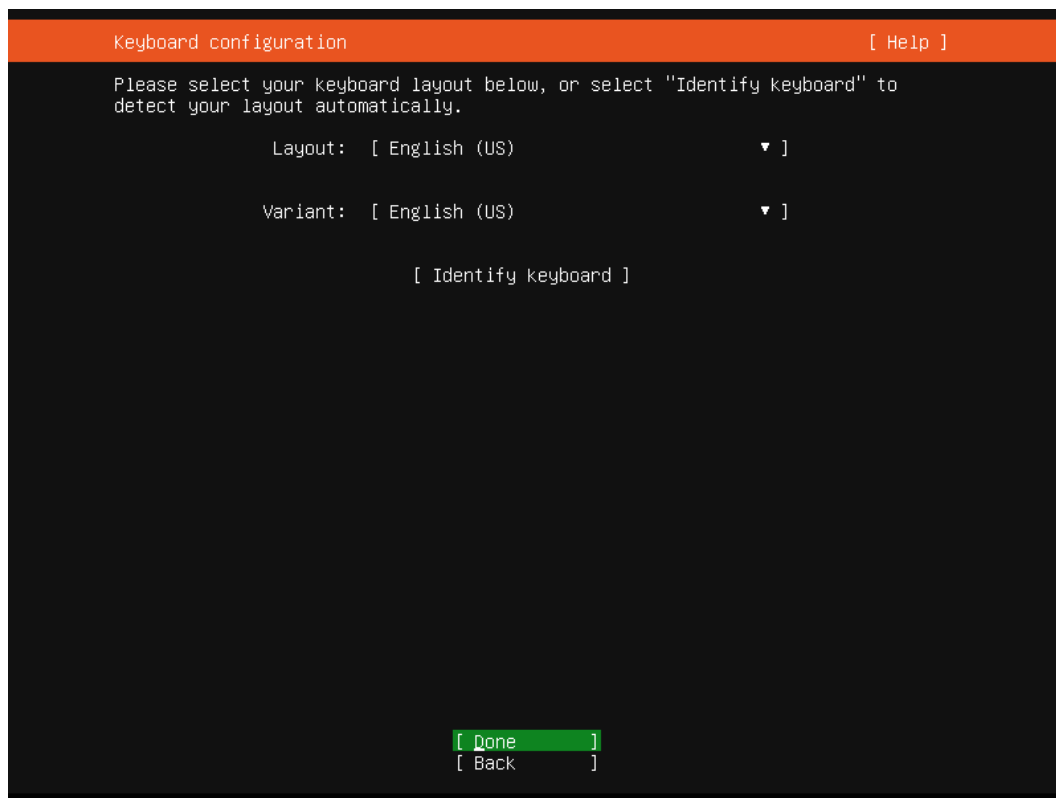
Gambar 4. 1 Pilih Bahasa Ubuntu Server

Gambar 4.1 di atas memperlihatkan bahasa-bahasa yang dapat digunakan pada ubuntu server. Pada penelitian ini, bahasa inggris digunakan sebagai bahasa sistem operasi Ubuntu server. Selanjutnya sistem operasi Ubuntu server akan menawarkan untuk melakukan *update* versi ubuntu server sebagaimana yang tertera pada gambar 4.2.



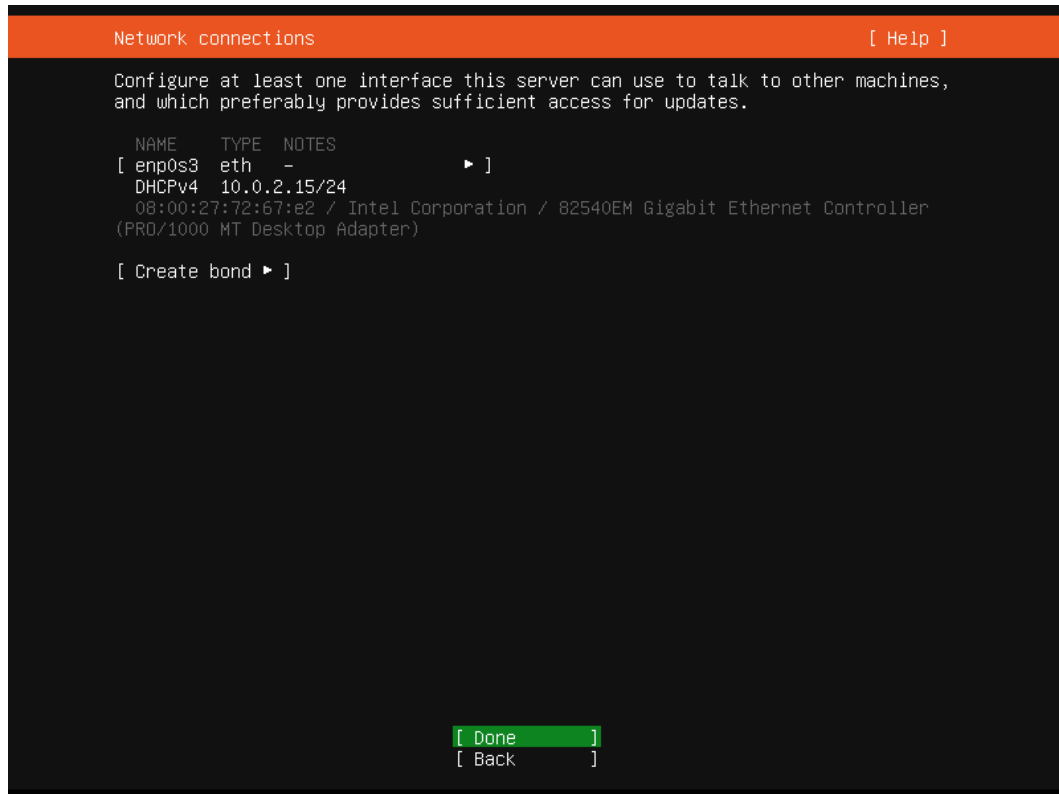
Gambar 4. 2 *Update* Ubuntu Server

Gambar 4.2. di atas memperlihatkan opsi untuk melakukan *update* versi Ubuntu server pada saat melakukan instalasi. Untuk melakukan *update* saat instalasi, pengguna dapat memilih “*Update to the new installer*”. Selanjutnya sistem akan mengunduh terlebih dahulu Ubuntu server versi terbaru untuk selanjutnya dilakukan instalasi. Pada penelitian ini, penulis tidak melakukan *update* versi Ubuntu server untuk mempercepat proses instalasi dengan memilih “*Continue without updating*”. Langkah selanjutnya yaitu melakukan konfigurasi *layout variant keyboard* sebagaimana yang tertera pada gambar 4.3.



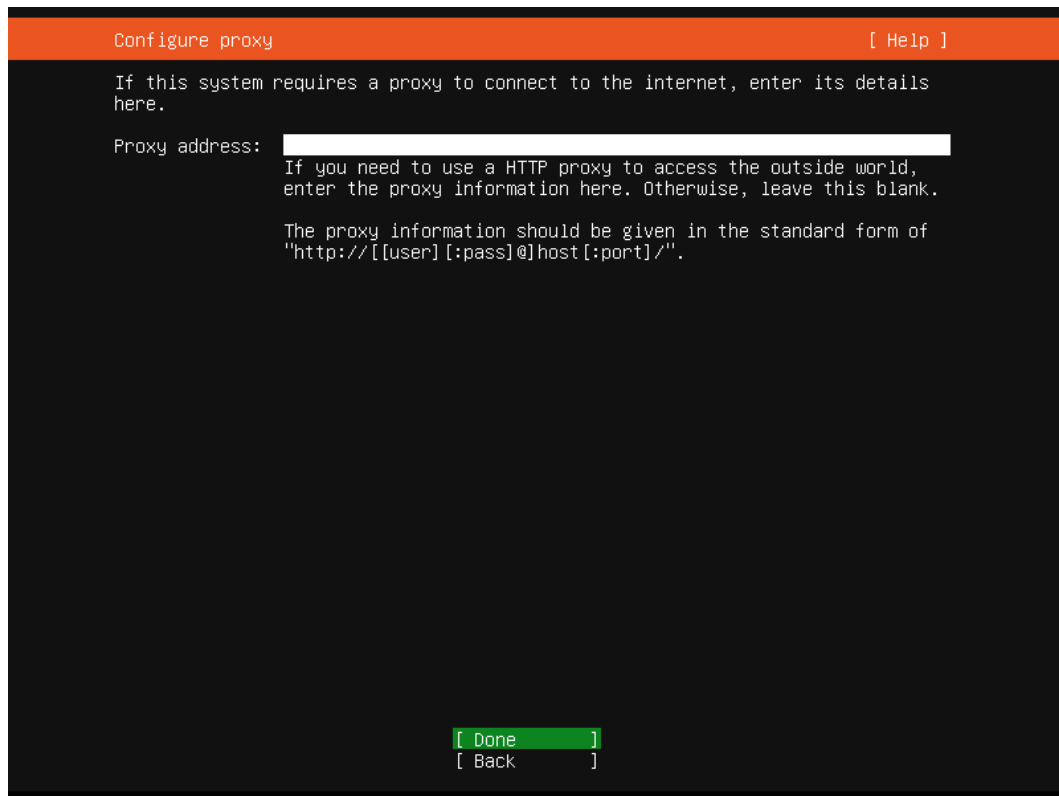
Gambar 4. 3 Konfigurasi *Keyboard* Ubuntu Server

Gambar 4.3 di atas memperlihatkan pilihan jenis *keyboard* yang akan digunakan pada sistem operasi Ubuntu server. Pada penelitian ini digunakan konfigurasi *keyboard default* yakni “*English (US)*”. Setelah konfigurasi *keyboard*, langkah selanjutnya yakni melakukan konfigurasi jaringan seperti *Network Adapter* dan *IP Address* sebagaimana yang tertera pada gambar 4.4.



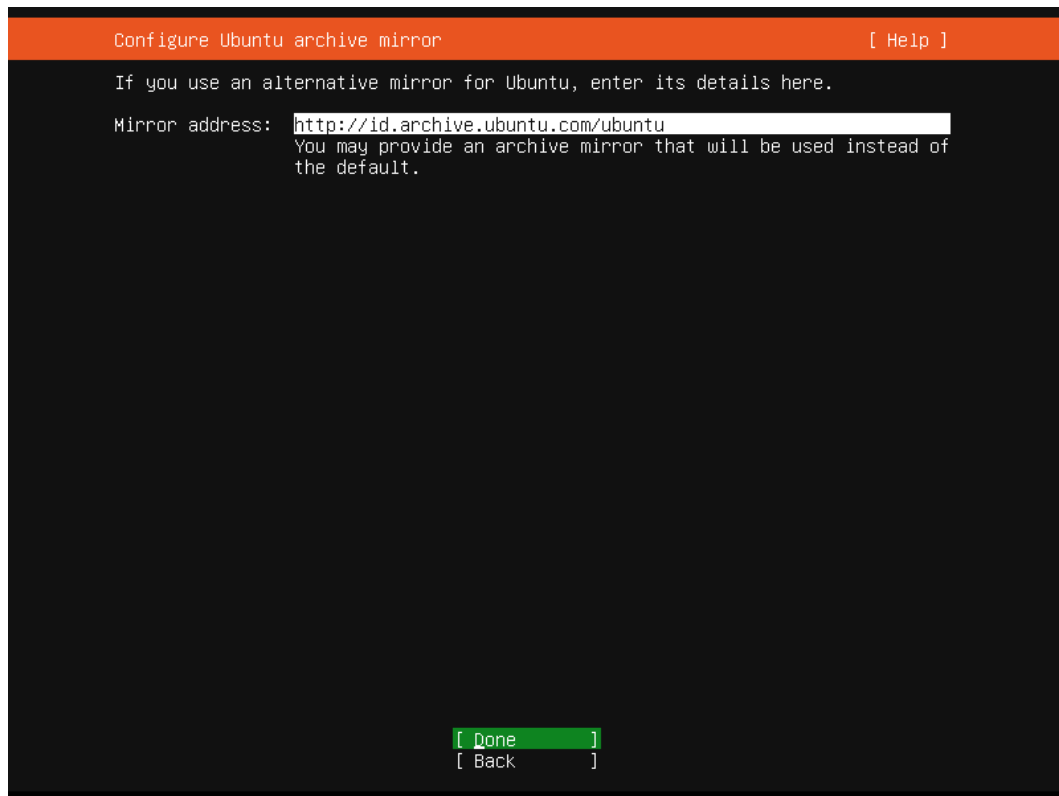
Gambar 4. 4 Konfigurasi Jaringan Ubuntu Server

Gambar 4.4 di atas memperlihatkan konfigurasi jaringan pada Ubuntu server. Pada tahap ini pengguna dapat memilih dan melakukan konfigurasi terhadap network adapter dan IP *address* yang akan digunakan. Pada penelitian ini digunakan konfigurasi *default* untuk mempercepat proses instalasi sistem operasi. Setelah melakukan konfigurasi jaringan, tahap selanjutnya adalah melakukan konfigurasi *proxy* sebagaimana yang tertera pada gambar 4.5.



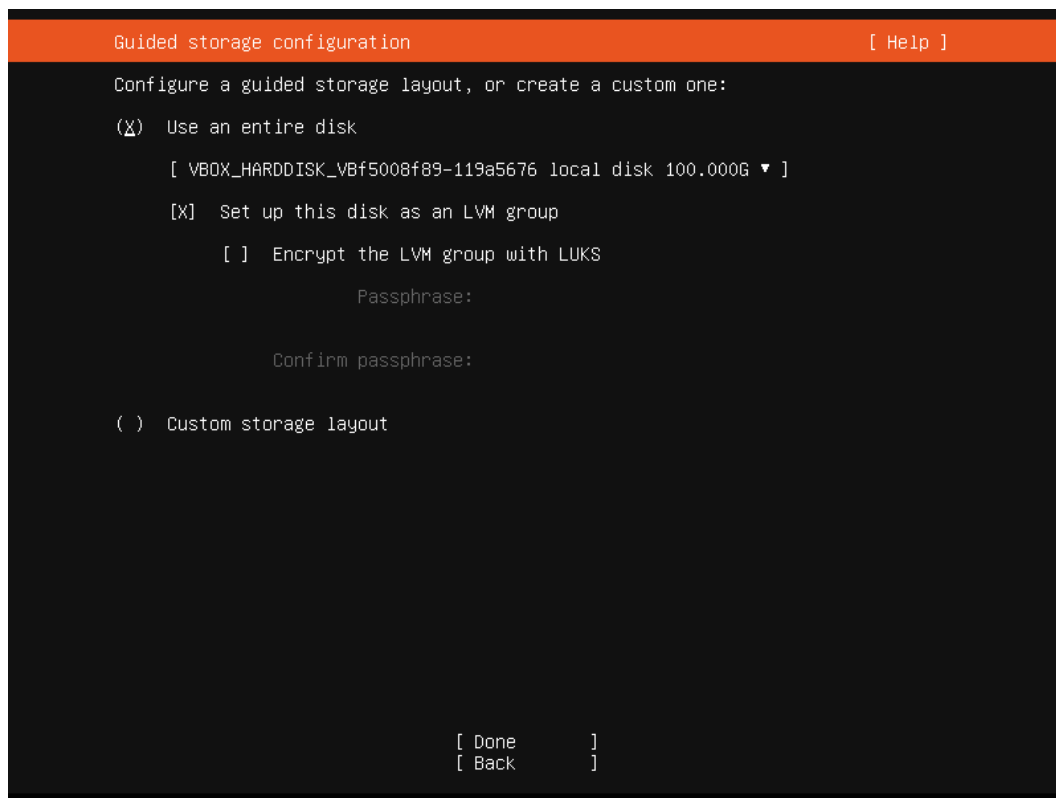
Gambar 4. 5 Konfigurasi *Proxy* Ubuntu Server

Gambar 4.5 di atas memperlihatkan konfigurasi *proxy* pada proses instalasi Ubuntu server. Pada penelitian ini, tidak digunakan *proxy* sehingga kolom *proxy address* dapat dibiarkan kosong. Setelah melalui tahap konfigurasi *proxy*, tahap selanjutnya yaitu konfigurasi Ubuntu *archive mirror* sebagaimana yang tertera pada gambar 4.6.



Gambar 4. 6 Konfigurasi Ubuntu *Archive Mirror* Ubuntu Server

Gambar 4.6 di atas memperlihatkan konfigurasi Ubuntu *archive mirror* pada Ubuntu server. Pada penelitian ini digunakan situs *archive mirror default* sebagai penyedia *file-file* yang akan diunduh. Setelah melakukan konfigurasi Ubuntu *archive mirror*, tahap selanjutnya yaitu melakukan konfigurasi penyimpanan sebagaimana yang tertera pada gambar 4.7.



Gambar 4. 7 Konfigurasi Penyimpanan Ubuntu Server

Gambar 4.7 di atas memperlihatkan konfigurasi penyimpanan pada Ubuntu server. Pada tahap ini pengguna dapat melakukan pembagian ruang penyimpanan (*storage*) pada sistem operasi Ubuntu server. Pada penelitian ini, penulis menggunakan konfigurasi *default* untuk ruang penyimpanan. Setelah melalui tahap konfigurasi penyimpanan, selanjutnya Ubuntu server akan menampilkan *file system summary* yang berisikan catatan penggunaan ruang penyimpanan sebelum dilakukan instalasi sebagaimana yang tertera pada gambar 4.8.

```

Storage configuration [ Help ]

FILE SYSTEM SUMMARY

MOUNT POINT      SIZE      TYPE      DEVICE TYPE
[ /              49.498G  new ext4  new LVM logical volume ▶ ]
[ /boot         1.000G  new ext4  new partition of local disk ▶ ]

AVAILABLE DEVICES

DEVICE                                TYPE                                SIZE
[ ubuntu-vg (new)                      LVM volume group                   98.996G ▶ ]
  free space                            49.498G

[ Create software RAID (md) ▶ ]
[ Create volume group (LVM) ▶ ]

USED DEVICES

DEVICE                                TYPE                                SIZE
[ ubuntu-vg (new)                      LVM volume group                   98.996G ▶ ]
  ubuntu-lv  new, to be formatted as ext4, mounted at /  49.498G ▶ ]

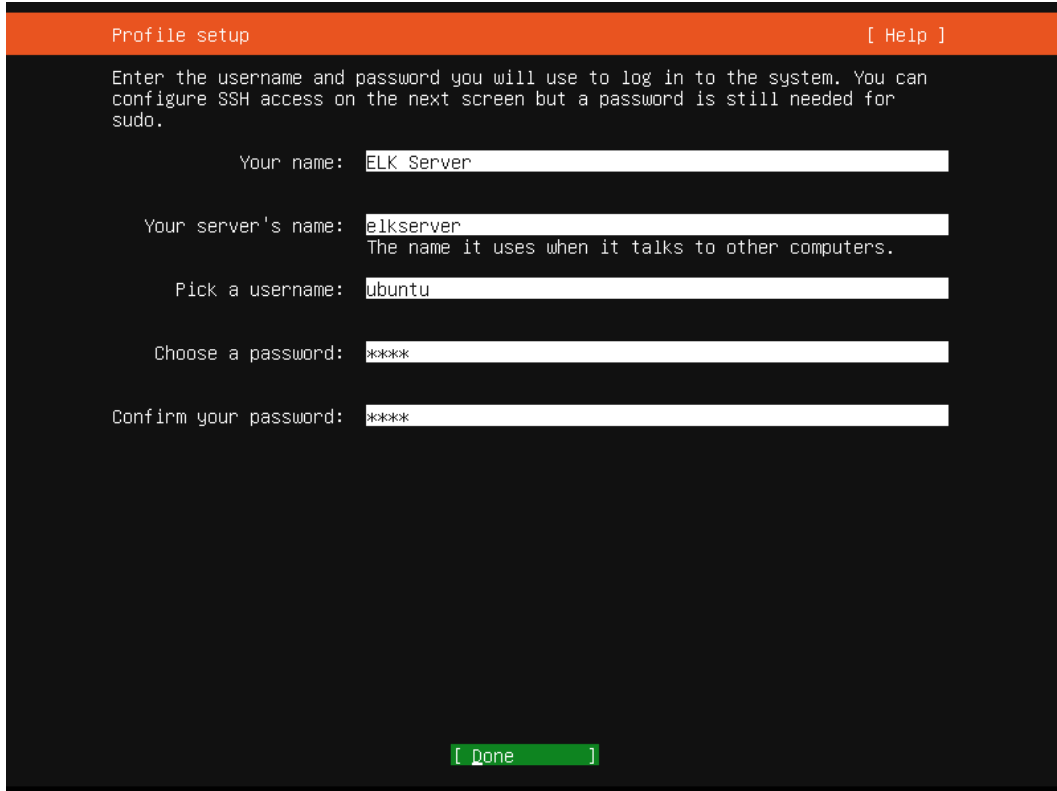
[ VBOX_HARDDISK_VBf5008f89-119a5676    local disk                          100.000G ▶ ]
  partition 1 new, bios_grub                            1.000M ▶ ]
  partition 2 new, to be formatted as ext4, mounted at /boot  1.000G ▶ ]
  partition 3 new, PV of LVM volume group ubuntu-vg           98.997G ▶ ]

[ Done ]
[ Reset ]
[ Back ]

```

Gambar 4. 8 *File System Summary* Ubuntu Server

Gambar 4.8 di atas memperlihatkan *file system summary* ubuntu server. Jika konfigurasi dirasa telah sesuai, pengguna dapat memilih “*Done*” untuk lanjut ke tahap berikutnya. Namun jika pengguna ingin mengulangi tahap konfigurasi sebelumnya, pengguna dapat memilih “*Reset*”. Setelah melalui tahap *file system summary*, langkah selanjutnya yaitu melakukan *profile* setup sebagaimana yang tertera pada gambar 4.9.



```
Profile setup [ Help ]

Enter the username and password you will use to log in to the system. You can
configure SSH access on the next screen but a password is still needed for
sudo.

Your name: ELK Server

Your server's name: elkserver
The name it uses when it talks to other computers.

Pick a username: ubuntu

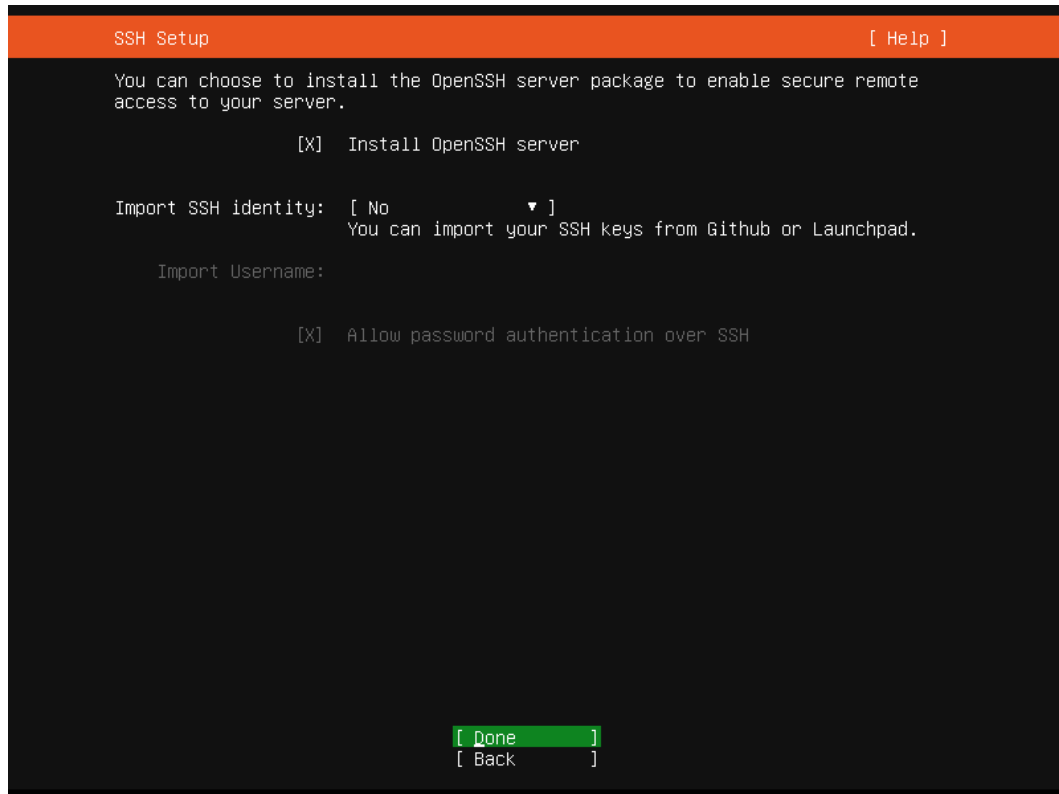
Choose a password: ****

Confirm your password: ****

[ Done ]
```

Gambar 4.9 *Profile Setup* Ubuntu Server

Gambar 4.9 di atas memperlihatkan *profile setup* yang diperlukan pada proses instalasi Ubuntu server. Pada tahap ini pengguna dapat memasukkan nama, nama server, *username*, *password* dan konfirmasi *password*. Setelah melakukan *profile setup*, langkah selanjutnya yaitu melakukan SSH Setup sebagaimana yang tertera pada gambar 4.10.



Gambar 4. 10 SSH Setup Ubuntu Server

Gambar 4.10 di atas memperlihatkan opsi untuk melakukan instalasi *OpenSSH* server pada Ubuntu server. SSH server berfungsi untuk mengizinkan *remote access* pada Ubuntu server. Pengguna dapat menekan tombol spasi untuk menandai pilihan “*Install OpenSSH server*” kemudian pilih “*Done*”. Setelah melakukan SSH Setup, langkah selanjutnya pengguna akan dihadapkan dengan *featured server snaps* yang berisikan pilihan perangkat lunak yang dapat di-*install* secara bersamaan dengan instalasi Ubuntu server sebagaimana yang tertera pada gambar 4.11.

```

Featured Server Snaps [ Help ]

These are popular snaps in server environments. Select or deselect with SPACE,
press ENTER to see more details of the package, publisher and versions
available.

[ ] microk8s           Lightweight Kubernetes for workstations and appliance ▶
[ ] nextcloud         Nextcloud Server - A safe home for all your data ▶
[ ] wekan             Open-Source kanban ▶
[ ] kata-containers  Lightweight virtual machines that seamlessly plug int ▶
[ ] docker           Docker container runtime ▶
[ ] canonical-livepatch Canonical Livepatch Client ▶
[ ] rocketchat-server Group chat server for 100s, installed in seconds. ▶
[ ] mosquito         Eclipse Mosquitto MQTT broker ▶
[ ] etcd             Resilient key-value store by CoreOS ▶
[ ] powershell      PowerShell for every system! ▶
[ ] stress-ng        A tool to load, stress test and benchmark a computer ▶
[ ] sabnzbd          SABnzbd ▶
[ ] wormhole         get things from one computer to another, safely ▶
[ ] aws-cli          Universal Command Line Interface for Amazon Web Servi ▶
[ ] google-cloud-sdk Command-line interface for Google Cloud Platform prod ▶
[ ] slcli           Python based SoftLayer API Tool. ▶
[ ] doctl           The official DigitalOcean command line interface ▶
[ ] conjure-up       Package runtime for conjure-up spells ▶
[ ] minidlna-escoand server software with the aim of being fully compliant ▶
[ ] postgresql10    PostgreSQL is a powerful, open source object-relation ▶
[ ] heroku          CLI client for Heroku ▶
[ ] keepalived      High availability VRRP/BFD and load-balancing for Lin ▶
[ ] prometheus      The Prometheus monitoring system and time series data ▶
[ ] juju           A model-driven operator lifecycle manager ▶

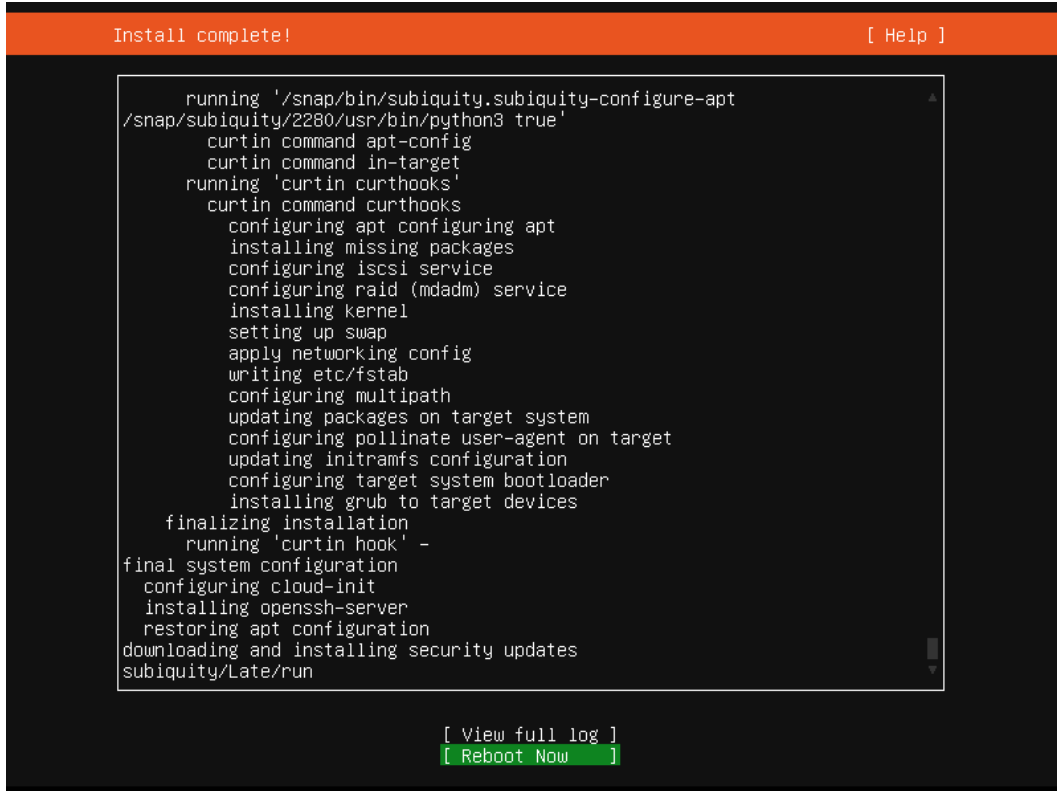
[ Done ]
[ Back ]

```

Gambar 4. 11 *Featured Server Snaps* Ubuntu Server

Gambar 4.11 di atas memperlihatkan *featured server snaps* yang berisikan pilihan perangkat lunak. Pengguna dapat menekan tombol spasi pada perangkat lunak yang ingin di-*install*. Namun pada penelitian ini, penulis tidak memerlukan aplikasi-aplikasi yang ada pada daftar, sehingga dapat dibiarkan kosong dan melanjutkan proses instalasi dengan memilih “*Done*”.

Selanjutnya sistem operasi akan mulai di-*install* pada server, proses instalasi ini memakan waktu yang cukup lama. Setelah proses instalasi selesai, reboot sistem operasi dengan memilih “*Reboot Now*” sebagaimana yang tertera pada gambar 4.12.



```

Install complete! [ Help ]

running '/snap/bin/subiquity.subiquity-configure-apt
/snap/subiquity/2280/usr/bin/python3 true'
curtin command apt-config
curtin command in-target
running 'curtin curthooks'
curtin command curthooks
configuring apt configuring apt
installing missing packages
configuring iscsi service
configuring raid (mdadm) service
installing kernel
setting up swap
apply networking config
writing etc/fstab
configuring multipath
updating packages on target system
configuring pollinate user-agent on target
updating initramfs configuration
configuring target system bootloader
installing grub to target devices
finalizing installation
running 'curtin hook' -
final system configuration
configuring cloud-init
installing openssh-server
restoring apt configuration
downloading and installing security updates
subiquity/Late/run

[ View full log ]
[ Reboot Now ]

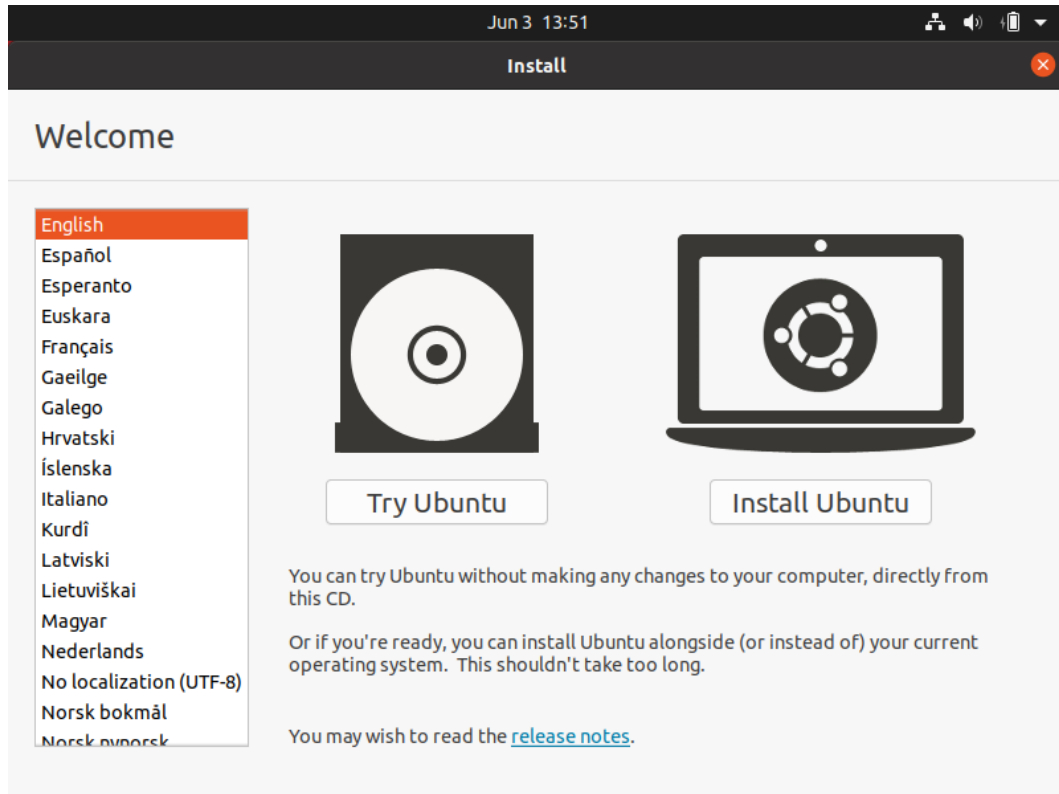
```

Gambar 4. 12 *Reboot* Ubuntu Server

Gambar 4.12 di atas memperlihatkan pilihan untuk melakukan *reboot* pada server setelah instalasi berhasil dilakukan. Pengguna dapat memilih “*Reboot Now*” untuk memuat ulang sistem operasi agar dapat digunakan.

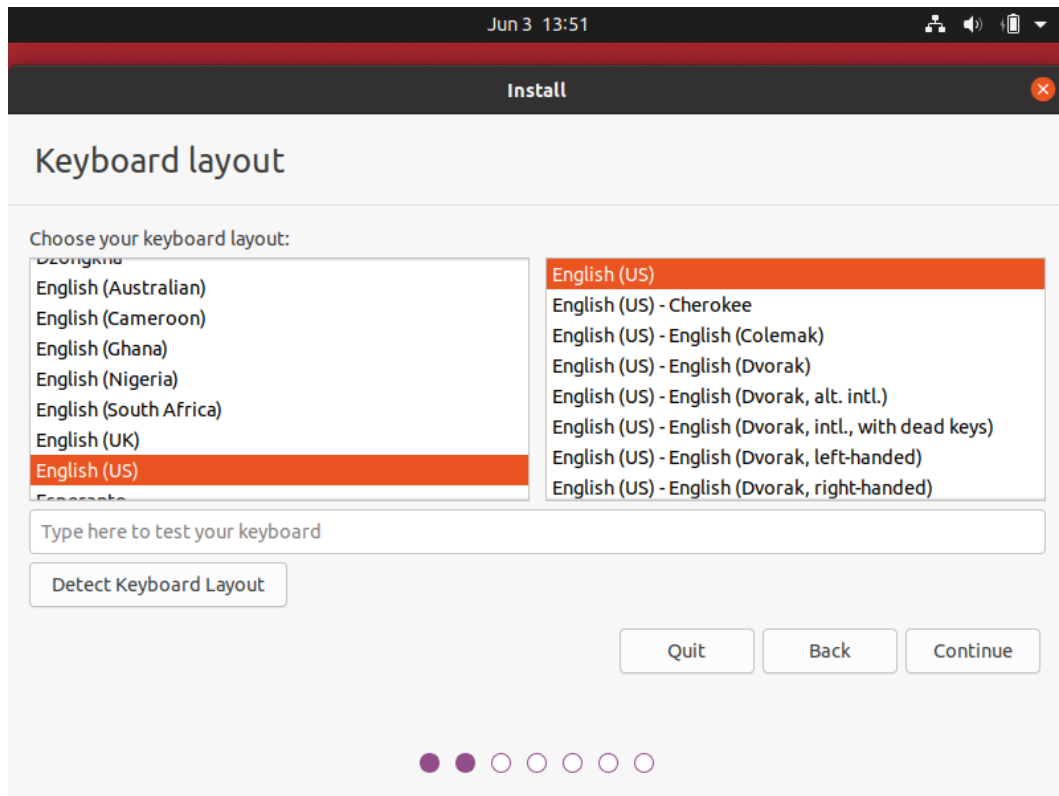
4.2. Instalasi Sistem Operasi Ubuntu Desktop

Selain Ubuntu Server, pada penelitian ini juga digunakan sistem operasi Ubuntu Desktop yang akan digunakan pada komputer System Administrator. Berbeda dengan Ubuntu server yang hanya menyediakan tampilan dalam bentuk *Command Line Interface* (CLI), Ubuntu Desktop juga menyediakan tampilan dalam bentuk *Graphical User Interface* (GUI). Adapun versi Ubuntu yang akan digunakan yakni Ubuntu Desktop 20.04 LTS *Focal Fossa*. Langkah pertama yang dilakukan untuk melakukan instalasi Ubuntu Desktop yakni dengan memilih bahasa sistem operasi sebagaimana yang tertera pada gambar 4.13.



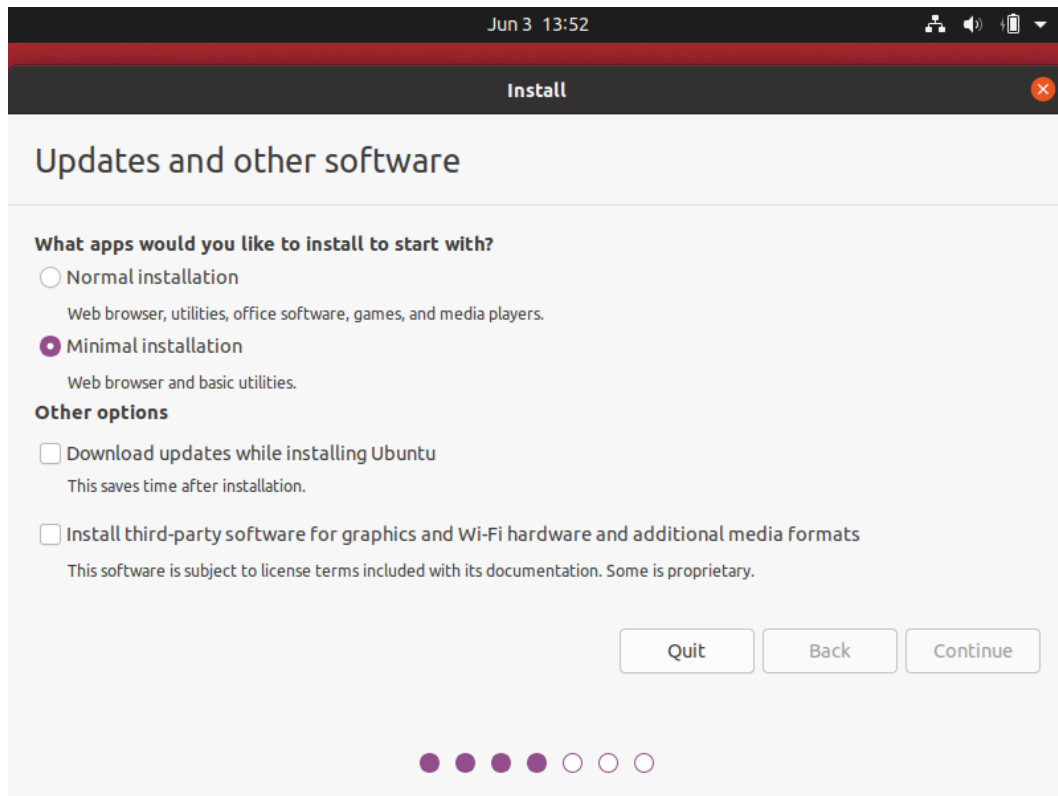
Gambar 4. 13 Tampilan Awal Instalasi Ubuntu Desktop

Gambar 4.13 di atas memperlihatkan tampilan awal instalasi ubuntu, di mana pada tahap awal ini selain memilih bahasa, pengguna juga diberikan pilihan untuk mencoba atau meng-*install* Ubuntu. Pada penelitian ini, penulis menggunakan bahasa inggris dan memilih “*Install Ubuntu*”. Tahap selanjutnya yakni melakukan konfigurasi *keyboard* sebagaimana yang tertera pada gambar 4.14.



Gambar 4. 14 Konfigurasi *Keyboard* Ubuntu Desktop

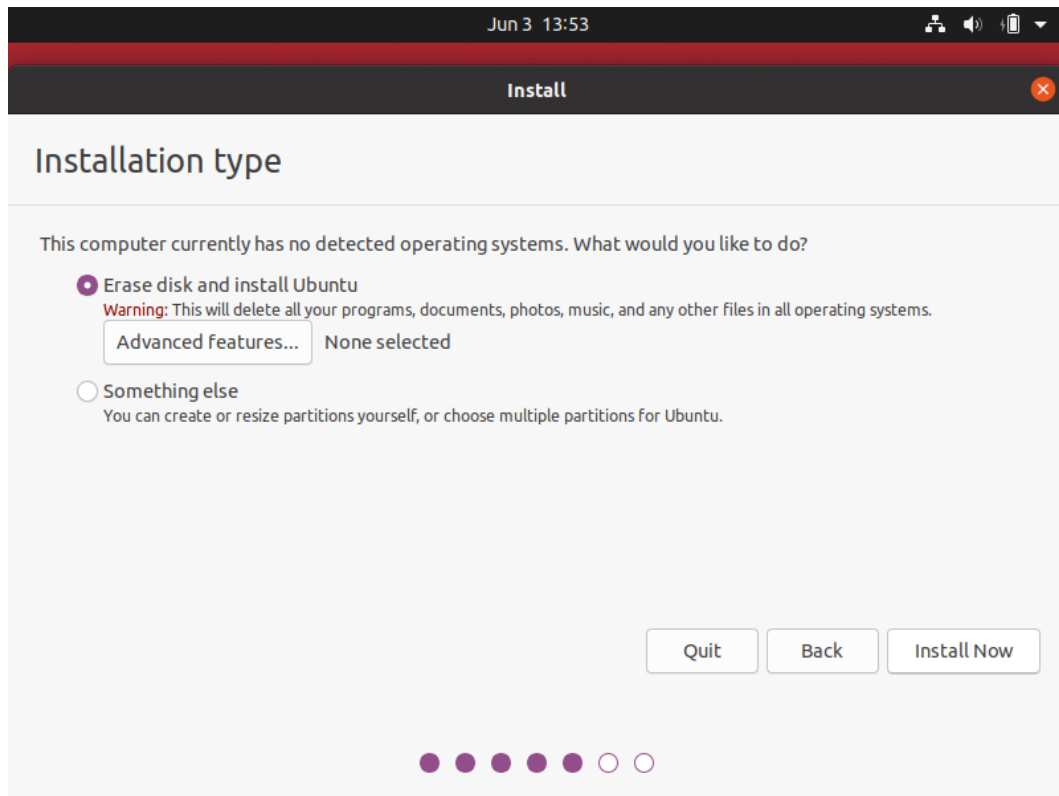
Gambar 4.14 di atas memperlihatkan pilihan *layout keyboard* yang dapat digunakan pada Ubuntu server. Pada penelitian ini, penulis menggunakan *layout default* yakni “*English (US)*”. Setelah itu pengguna akan dihadapkan dengan pilihan instalasi sebagaimana yang tertera pada gambar 4.15.



Gambar 4. 15 Jenis Instalasi Ubuntu Desktop

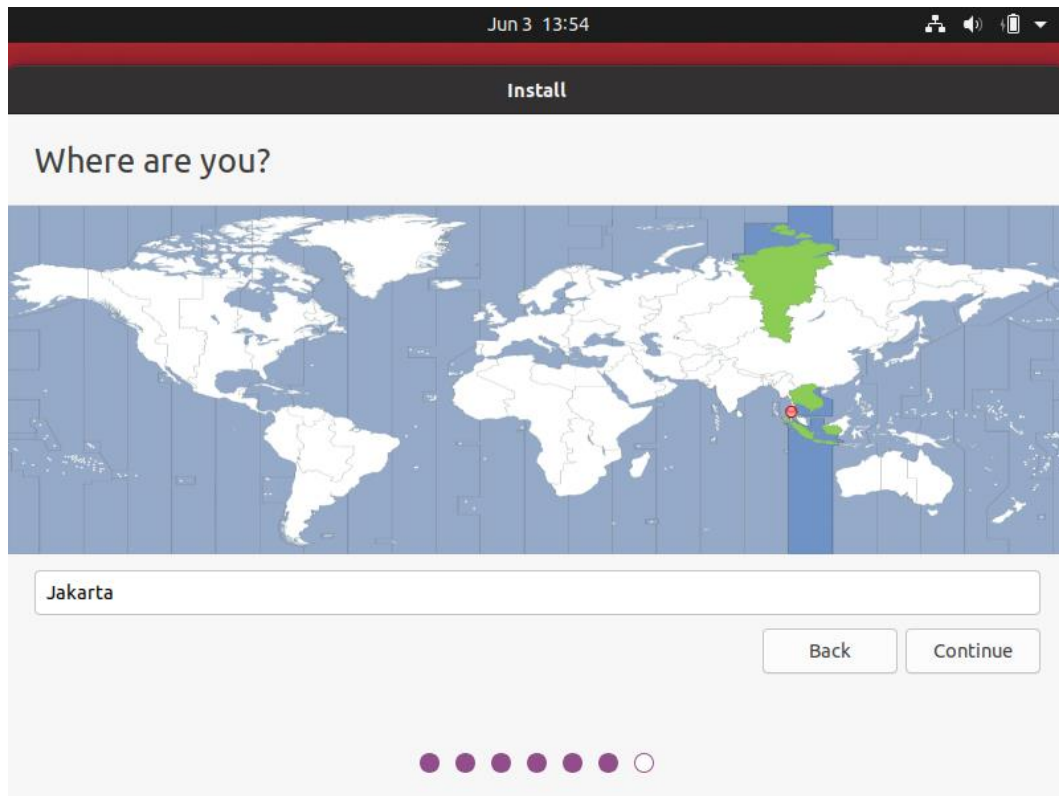
Gambar 4.15 di atas memperlihatkan pilihan yang ada dalam instalasi Ubuntu Desktop. Pada tahap ini terdapat 2 pilihan yakni normal installation dan minimal installation. Normal installation akan melakukan instalasi sistem operasi sekaligus beberapa perangkat lunak seperti web browser, *office software*, *games* dan media player. Sedangkan jika memilih minimal *installation*, sistem operasi akan di-*install* hanya dengan beberapa perangkat lunak yang *basic*.

Pada penelitian ini, penulis memilih minimal *installation* agar proses instalasi lebih cepat. Setelah memilih jenis instalasi, tahap selanjutnya yaitu dengan melakukan konfigurasi ruang penyimpanan (*storage*) yang akan digunakan sebagaimana yang tertera pada gambar 4.16.



Gambar 4. 16 Konfigurasi Ruang Penyimpanan Ubuntu Desktop

Gambar 4.16 di atas memperlihatkan konfigurasi ruang penyimpanan (*storage*) pada Ubuntu desktop. Pengguna dapat memilih untuk menghapus seluruh data yang ada pada *disk* atau melakukan partisi terlebih dahulu. Pada penelitian ini, penulis memilih untuk menghapus data disk dan melakukan instalasi agar seluruh *storage* dapat digunakan untuk Ubuntu Desktop. Setelah melakukan konfigurasi ruang penyimpanan, tahap selanjutnya yaitu dengan menentukan lokasi sebagaimana yang tertera pada gambar 4.17.



Gambar 4. 17 Penentuan Lokasi Ubuntu Desktop

Gambar 4.17 memperlihatkan pemilihan lokasi di mana pengguna berada. Pemilihan lokasi ini diperlukan untuk menyesuaikan tanggal dan waktu dari sistem operasi. Setelah memilih lokasi, tahap selanjutnya adalah mengisi *profile* setup sebagaimana yang tertera pada gambar 4.18.

Jun 3 20:55

Install

Who are you?

Your name: ✓

Your computer's name: ✓
The name it uses when it talks to other computers.

Pick a username: ✓

Choose a password: Short password

Confirm your password: ✓

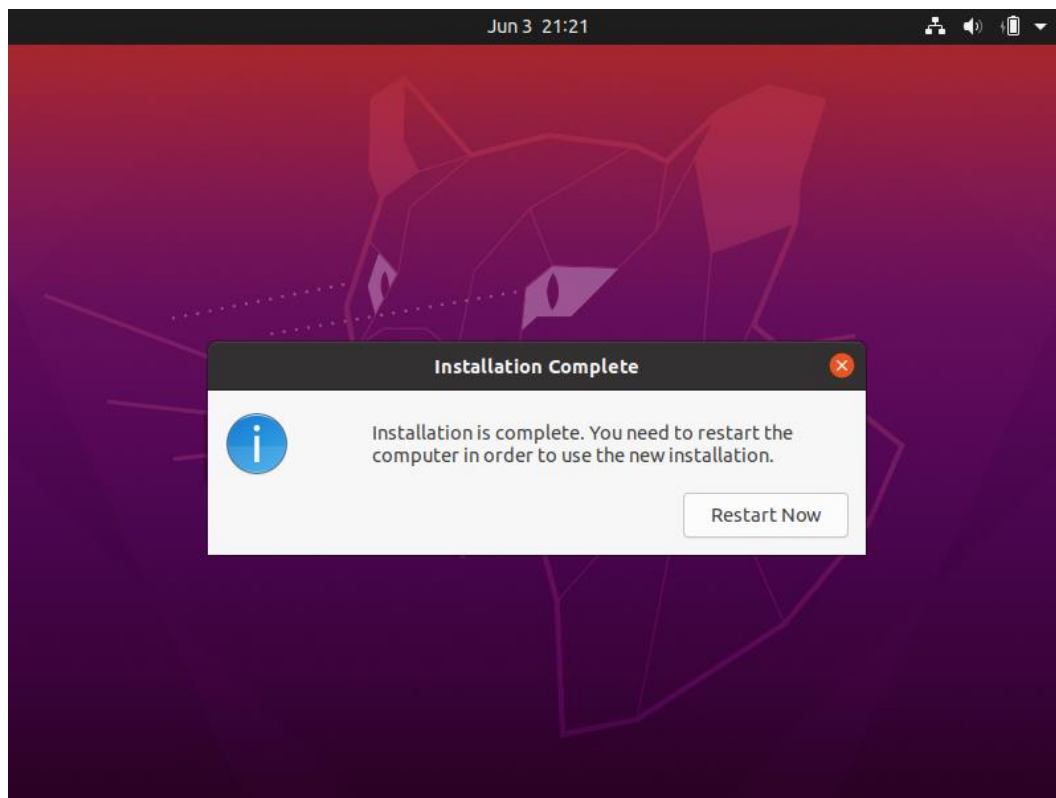
Log in automatically
 Require my password to log in

Back Continue

● ● ● ● ● ● ●

Gambar 4. 18 *Profile Setup Ubuntu Desktop*

Gambar 4.18 memperlihatkan data-data yang diperlukan untuk mengisi profil pengguna pada sistem operasi Ubuntu desktop. Pada tahap ini pengguna diminta untuk memasukkan nama, nama komputer, *username*, *password* dan konfirmasi *password*. Setelah melakukan profile setup, proses instalasi sistem operasi akan dimulai dan memakan waktu yang cukup lama. Setelah proses instalasi selesai Ubuntu desktop akan menampilkan kotak dialog untuk melakukan *restart* sebagai mana yang tertera pada gambar 4.19.



Gambar 4. 19 Kotak Dialog *Restart* Ubuntu Desktop

Gambar 4.19 di atas memperlihatkan kotak dialog restart setelah proses instalasi Ubuntu desktop selesai dilakukan. Pengguna dapat memilih “*Restart Now*” untuk memuat ulang sistem operasi agar dapat digunakan.

4.3. Konfigurasi IP *Static*

Sebelum dilakukan instalasi dan konfigurasi seluruh perangkat lunak yang akan diterapkan pada web server ELK *Stack* server maupun PC *System Administrator*, langkah pertama yang dilakukan adalah melakukan konfigurasi IP *Address*. Sebelum melakukan konfigurasi IP *Static*, hal yang perlu diketahui adalah nama dari *network interface* yang digunakan. Adapun perintah untuk mengetahui nama *network interface* tertera pada gambar 4.20.

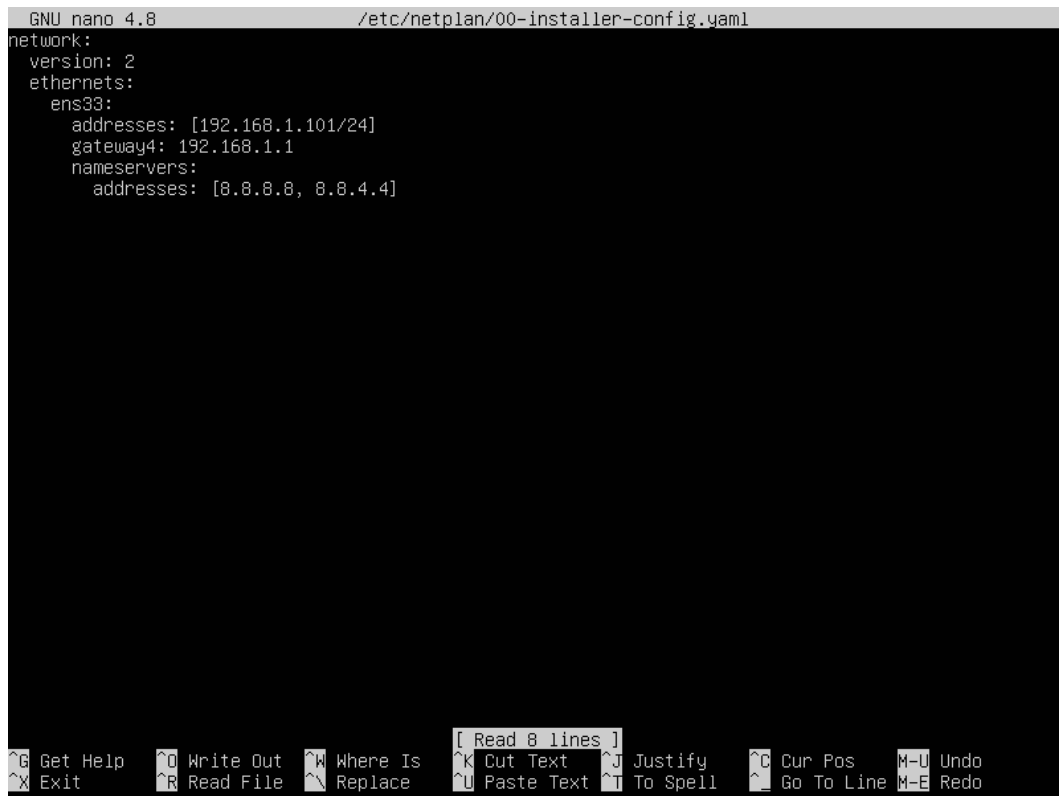
```

ubuntu@elkserver:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:6d:d6:b5 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.4/24 brd 192.168.1.255 scope global dynamic enp0s3
        valid_lft 84532sec preferred_lft 84532sec
    inet6 fe80::a00:27ff:fe6d:d6b5/64 scope link
        valid_lft forever preferred_lft forever
ubuntu@elkserver:~$

```

Gambar 4. 20 Nama *Network Interface*

Gambar 4.20 memperlihatkan perintah pada terminal Linux untuk mengetahui nama *network interface* pada web server. Pengguna dapat memasukkan perintah “ip a” pada terminal untuk mengetahui nama *network interface* tersebut. Pada penelitian ini nama *network interface* pada web server yaitu enp0s3. Setelah mengetahui nama dari *network interface*, langkah selanjutnya yaitu melakukan konfigurasi IP *Static*. Konfigurasi IP *Static* tersebut hanya dapat dilakukan oleh *user* dengan hak akses *root*. Konfigurasi IP *Static* ini dilakukan dengan mengubah *file* “/etc/netplan/00-installer-config.yaml”. Adapun contoh konfigurasi pada IP *Static* tertera pada gambar 4.21.



```

GNU nano 4.8 /etc/netplan/00-installer-config.yaml
network:
  version: 2
  ethernets:
    ens33:
      addresses: [192.168.1.101/24]
      gateway4: 192.168.1.1
      nameservers:
        addresses: [8.8.8.8, 8.8.4.4]
  
```

Terminal window showing the configuration of a static IP address in a netplan file using nano 4.8. The configuration is for the ens33 interface with IP 192.168.1.101/24, gateway 192.168.1.1, and DNS servers 8.8.8.8 and 8.8.4.4. The terminal also shows the nano editor's status bar with various commands like Get Help, Write Out, Where Is, Cut Text, Paste Text, Justify, Cur Pos, Go To Line, Undo, and Redo.

Gambar 4. 21 Konfigurasi IP *Static*

Gambar 4.21 di atas memperlihatkan konfigurasi IP *Static* pada Linux Ubuntu. Penerapan IP *Static* bertujuan untuk menetapkan IP *Address* dari setiap perangkat agar tidak mengalami perubahan ketika di-*restart*. Konfigurasi IP *Static* ini dilakukan pada Web Server, ELK *Stack* Server dan PC *System* Administrator dengan menyesuaikan nama *network interface* dan IP *Address* dari setiap perangkat.

4.4. Konfigurasi Web Server

Pada tahap ini akan dilakukan instalasi perangkat lunak yang dibutuhkan oleh web server agar dapat di akses dan diintegrasikan dengan ELK *Stack* server untuk membangun Log *Event Management*.

4.4.1. Instalasi Perangkat Lunak pada Web Server

Pada penelitian ini, dibutuhkan beberapa perangkat lunak seperti *Apache* Web Server, *MySQL*, PHP, dan *Filebeat*. Untuk melakukan instalasi tersebut, penulis membuat *bash script* yang berisikan perintah untuk melakukan instalasi

seluruh perangkat lunak yang dibutuhkan. Berikut merupakan *bash script* yang disimpan dengan nama `web-server-app.sh` :

```
#!/bin/bash

#Gunakan script ini dengan root privilege
#Install Apache
apt install apache2 -y

#Install MySQL
apt install mysql-server -y

#Install PHP
apt install php libapache2-mod-php php-mysql -y

#Install Filebeat
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key
add -
echo "deb https://artifacts.elastic.co/packages/7.x/apt-stable/main" | sudo tee
/etc/apt/sources.list.d/elastic-7.x.list
apt install apt-transport-https
apt update
apt install filebeat -y

#Tambah user MySQL
mysql -u root <<MYSQL_SCRIPT
CREATE USER 'admin'@'localhost' IDENTIFIED BY 'admin';
GRANT ALL PRIVILEGES ON *.* TO 'admin'@'localhost';
FLUSH PRIVILEGES;
MYSQL_SCRIPT

#Instalasi fail2ban
apt install fail2ban -y
touch /etc/fail2ban/jail.local

echo
"=====
echo "Seluruh aplikasi berhasil di install"
echo "User MySQL berhasil dibuat"
echo "Username: admin"
echo "Password: admin"
echo
"=====
```


Setelah menyimpan *script* di atas, dengan menggunakan hak akses *root* buat *script* tersebut *executable* dan jalankan *script* di atas dengan memasukkan perintah :

```
“chmod +x web-server-app.sh && ./web-server-app.sh“.
```

4.4.2. Konfigurasi Apache Web Server dan MySQL

Pada tahap ini dilakukan konfigurasi agar web server menampilkan web *application* yang telah dibuat sebelumnya. Untuk melakukan konfigurasi tersebut, penulis membuat *bash script* sebagai berikut :

```
#!/bin/bash

#Jalankan script ini dengan root permission
#Download web app dari github
git clone https://github.com/bukharihasan/vuln-web.git
mv vuln-web /var/www/

#Import database
mysql -u admin -p<<MYSQL_SCRIPT
CREATE DATABASE vuln;
MYSQL_SCRIPT
mysql -u admin -p vuln < /var/www/vuln-web/vuln.sql

#Ubah folder default apache
sed -i "s/html/vuln-web/" /etc/apache2/sites-available/000-default.conf
service apache2 restart && service mysql restart

echo "Konfigurasi berhasil dilakukan"
```

Bash script di atas berfungsi untuk membuat melakukan konfigurasi pada Apache dan MySQL meliputi pengunduhan web *application* dari *github*, pembuatan dan *import database* dan konfigurasi *sites-available* pada Apache. *Bash script* tersebut disimpan dengan nama *konfig-apache.sh*, kemudian buat *file* tersebut *executable* dan jalankan dengan perintah :

```
“chmod +x konfig-apache.sh && ./konfig-apache.sh”.
```

Setelah proses instalasi selesai, pastikan web *application* dapat diakses dengan cara memasukkan *IP address* dari web server pada web browser. Adapun tampilan dari web *application* tertera pada gambar 4.22.



Gambar 4. 22 Tampilan Web *Application*

Gambar 4.22 di atas memperlihatkan tampilan dari halaman indeks web *application*. Web *application* tersebut memiliki celah keamanan *SQL Injection*, *Cross Site Scripting* dan *Local File Inclusion*. Celah keamanan tersebut dibuat untuk proses uji coba web *application firewall*.

4.4.3. Konfigurasi Web Application Firewall

Setelah melakukan instalasi aplikasi dan konfigurasi *Apache* dan *MySQL*, pada tahap ini dilakukan proses konfigurasi *modsecurity*. *ModSecurity* merupakan *Web Application Firewall* yang akan mendeteksi dan memblokir setiap serangan yang ditujukan kepada web server. Untuk melakukan konfigurasi tersebut, penulis membuat *bash script* sebagai berikut :

```
#!/bin/bash

#Jalankan script ini dengan root permission
#Update dan install aplikasi pendukung
apt update -y
apt install g++ flex bison curl apache2-dev doxygen libyajl-dev ssdeep liblua5.2-dev libgeoip-dev libtool dh-autoreconf libcurl4-gnutls-dev libxml2 libpcre++-dev
```

```

libxml2-dev git -y

#Clone ModSecurity dari github dan install
wget
https://github.com/SpiderLabs/ModSecurity/releases/download/v3.0.4/modsecurity-v3.0.4.tar.gz
tar -xvzf modsecurity-v3.0.4.tar.gz
cd modsecurity-v3.0.4
./build.sh
./configure
make
make install

#Install apache modsecurity connector
cd ~
git clone https://github.com/SpiderLabs/ModSecurity-apache
cd ModSecurity-apache
./autogen.sh
./configure --with-libmodsecurity=/usr/local/modsecurity/
make
make install
echo "LoadModule security3_module
/usr/lib/apache2/modules/mod_security3.so" | sudo tee -a
/etc/apache2/apache2.conf
mkdir /etc/apache2/modsecurity.d
cp ~/modsecurity-v3.0.4/modsecurity.conf-recommended
/etc/apache2/modsecurity.d/modsecurity.conf
cp ~/modsecurity-v3.0.4/unicode.mapping /etc/apache2/modsecurity.d/
touch /etc/apache2/modsecurity.d/modsec_rules.conf
echo 'Include "/etc/apache2/modsecurity.d/modsecurity.conf"' >>
/etc/apache2/modsecurity.d/modsec_rules.conf
echo 'Include "/etc/apache2/modsecurity.d/owasp-crs/crs-setup.conf"' >>
/etc/apache2/modsecurity.d/modsec_rules.conf
echo 'Include "/etc/apache2/modsecurity.d/owasp-crs/rules/*.conf"' >>
/etc/apache2/modsecurity.d/modsec_rules.conf
sed -i 's/SecRuleEngine DetectionOnly/SecRuleEngine On/'
/etc/apache2/modsecurity.d/modsecurity.conf

#Download dan konfigurasi OWASP ModSecurity Core Rule Set
git clone https://github.com/SpiderLabs/owasp-modsecurity-crs.git
/etc/apache2/modsecurity.d/owasp-crs
cp /etc/apache2/modsecurity.d/owasp-crs/crs-setup.conf{.example,}

echo "Proses konfigurasi ModSecurity telah selesai"

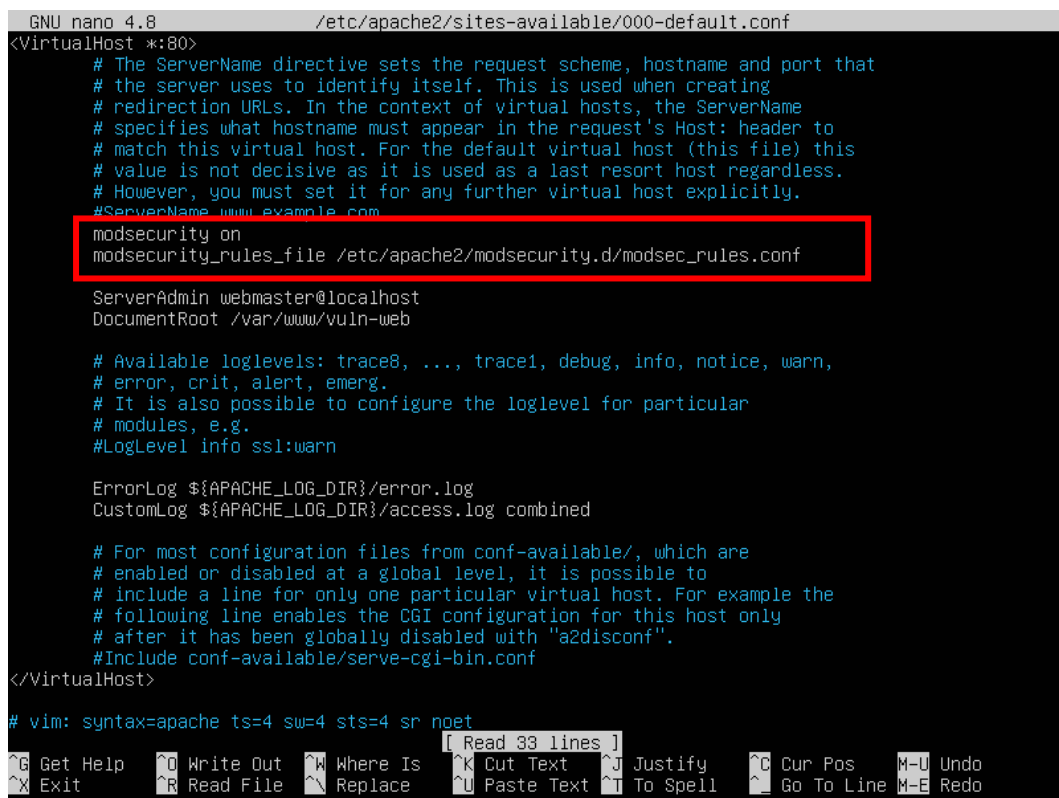
```

Bash script di atas berfungsi untuk melakukan konfigurasi *ModSecurity* meliputi instalasi perangkat lunak pendukung, pengunduhan *modsecurity*, instalasi

dan konfigurasi *modsecurity*, pengunduhan dan instalasi konektor *apache* dengan *modsecurity* dan pengunduhan OWASP *ModSecurity Core Rule Set*. Bash script di atas disimpan dengan nama *konfig-modsec.sh*, kemudian buat *file* di atas tersebut *executable* dan jalankan dengan perintah :

“`chmod +x konfig-modsec.sh && ./konfig-modsec.sh`”

Setelah menjalankan proses instalasi dan konfigurasi dengan *bash script* di atas selesai, selanjutnya tambahkan konfigurasi pada `/etc/apache2/sites-available/000-default.conf` sebagaimana yang tertera pada gambar 4.23.



```

GNU nano 4.8 /etc/apache2/sites-available/000-default.conf
<VirtualHost *:80>
# The ServerName directive sets the request scheme, hostname and port that
# the server uses to identify itself. This is used when creating
# redirection URLs. In the context of virtual hosts, the ServerName
# specifies what hostname must appear in the request's Host: header to
# match this virtual host. For the default virtual host (this file) this
# value is not decisive as it is used as a last resort host regardless.
# However, you must set it for any further virtual host explicitly.
#ServerName www.example.com
modsecurity on
modsecurity_rules_file /etc/apache2/modsecurity.d/modsec_rules.conf
ServerAdmin webmaster@localhost
DocumentRoot /var/www/vuln-web

# Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
# error, crit, alert, emerg.
# It is also possible to configure the loglevel for particular
# modules, e.g.
#LogLevel info ssl:warn

ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined

# For most configuration files from conf-available/, which are
# enabled or disabled at a global level, it is possible to
# include a line for only one particular virtual host. For example the
# following line enables the CGI configuration for this host only
# after it has been globally disabled with "a2disconf".
#Include conf-available/serve-cgi-bin.conf
</VirtualHost>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos M-U Undo
^X Exit ^R Read File ^M Replace ^U Paste Text ^T To Spell ^G Go To Line M-E Redo

```

Gambar 4. 23 Konfigurasi *ModSecurity* pada *Sites Available Apache*

Gambar 4.23 di atas memperlihatkan konfigurasi *modsecurity* pada *site available apache*. Agar *modsecurity* dapat aktif, tambahkan *code* dalam tanda merah pada gambar di atas pada file `000-default.conf`. Setelah menambahkan *code* tersebut, *restart apache* dengan perintah :

“`service apache2 restart`”.

4.4.4. Konfigurasi Filebeat

Setelah melakukan instalasi dan konfigurasi pada *apache* dan *modsecurity*, pada tahap ini dilakukan konfigurasi *filebeat*. Pada penelitian ini *filebeat* digunakan sebagai data *shipper* yang mengirimkan *file* log dari web server ke ELK *Stack* server. Untuk melakukan data *shipper* tersebut konfigurasi *file* log yang akan dikirim dan tujuannya pada *file* `/etc/filebeat/filebeat.yml` sebagaimana yang tertera pada gambar 4.24.

```

GNU nano 4.8 /etc/filebeat/filebeat.yml
# ===== Filebeat inputs =====
filebeat.inputs:
# Each - is an input. Most options can be set at the input level, so
# you can use different inputs for various configurations.
# Below are the input specific configurations.
- type: log
# Change to true to enable this input configuration.
enabled: true
# Paths that should be crawled and fetched. Glob based paths.
paths:
- /var/log/apache2/error.log
#- c:\programdata\elasticsearch\logs\*
# ===== Logstash Output =====
output.logstash:
# The Logstash hosts
hosts: ["192.168.1.101:5044"]
# Optional SSL. By default is off.
# List of root certificates for HTTPS server verifications
#ssl.certificate_authorities: ["/etc/pki/root/ca.pem"]
# Certificate for SSL client authentication
#ssl.certificate: "/etc/pki/client/cert.pem"
# Client Certificate Key
#ssl.key: "/etc/pki/client/cert.key"
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos M-U Undo
^X Exit ^R Read File ^M Replace ^U Paste Text ^T To Spell ^_ Go To Line M-E Redo

```

Gambar 4. 24 Konfigurasi *Filebeat*

Gambar 4.24 di atas memperlihatkan konfigurasi *filebeat* pada web server. Konfigurasi di atas bertujuan untuk mengaktifkan fungsi *filebeat*, menentukan log yang akan dikirim dan tujuan log tersebut.

4.4.5. Konfigurasi Fail2ban

Fail2ban merupakan tools yang akan memonitor log dan melakukan *blocking* terhadap *IP address* apabila sesuai dengan *rules* yang ditetapkan. *IP Address* tersebut akan diblokir sehingga tidak dapat mengakses server dalam

kurun waktu yang ditentukan. Langkah pertama, lakukan konfigurasi pada jail *file* yang terletak pada `/etc/fail2ban/jail.local` dengan konfigurasi sebagaimana yang tertera pada gambar 4.25.



```

GNU nano 4.8 /etc/fail2ban/jail.local Modified
[apache-modsecurity]
enabled = true
port = http,https
logpath = %(apache_error_log)s
bantime = 60m
maxretry = 1
  
```

Gambar 4. 25 Konfigurasi Jail *File* pada Fail2ban

Gambar 4.45 di atas memperlihatkan konfigurasi jail *file* pada fail2ban, konfigurasi tersebut berfungsi untuk mengaktifkan jail, menentukan *port* yang diproteksi, menentukan direktori log, menentukan lama waktu pemblokiran dan maksimal percobaan. Pada konfigurasi di atas *port* yang diproteksi merupakan *port http* dan *https*, log yang dimonitor yakni *error.log apache*, waktu pemblokiran selama 60 menit dan maksimum percobaan 1 kali.

Selain melakukan konfigurasi pada jail *file*, langkah selanjutnya yaitu melakukan konfigurasi pada filter *apache-modsecurity* yang ada pada *file /etc/fail2ban/filter.d/apache-modsecurity.conf*. Hal ini perlu dilakukan karena secara *default* filter *apache-modsecurity* mengikuti pola dari log *modsecurity* versi 2. Adapun konfigurasi yang dilakukan pada filter *apache-modsecurity* tertera pada gambar 4.26.

```

# Fail2Ban apache-modsec filter
#

[INCLUDES]

# Read common prefixes. If any customizations available -- read them from
# apache-common.local
before = apache-common.conf

[Definition]

failregex = ^%(_apache_error_client)s(?: \[client [^\]]+\])? ModSecurity:\s+Warning. Matched.+\[file
.+APPLICATION-ATTACK

ignoreregex =

# https://github.com/SpiderLabs/ModSecurity/wiki/ModSecurity-2-Data-Formats
# Author: Daniel Black
#       Sergey G. Brester aka sebres (review, optimization)
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~

"/etc/fail2ban/filter.d/apache-modsecurity.conf" 19L, 486C                               1,7                               All

```

Gambar 4. 26 Konfigurasi Filter *Apache Modsecurity* pada Fail2ban

Gambar 4.26 di atas memperlihatkan konfigurasi filter *apache-modsecurity* pada fail2ban. Pada konfigurasi ini dilakukan pembaruan *failregex* agar sesuai dengan pola dari log *modsecurity* versi 3.

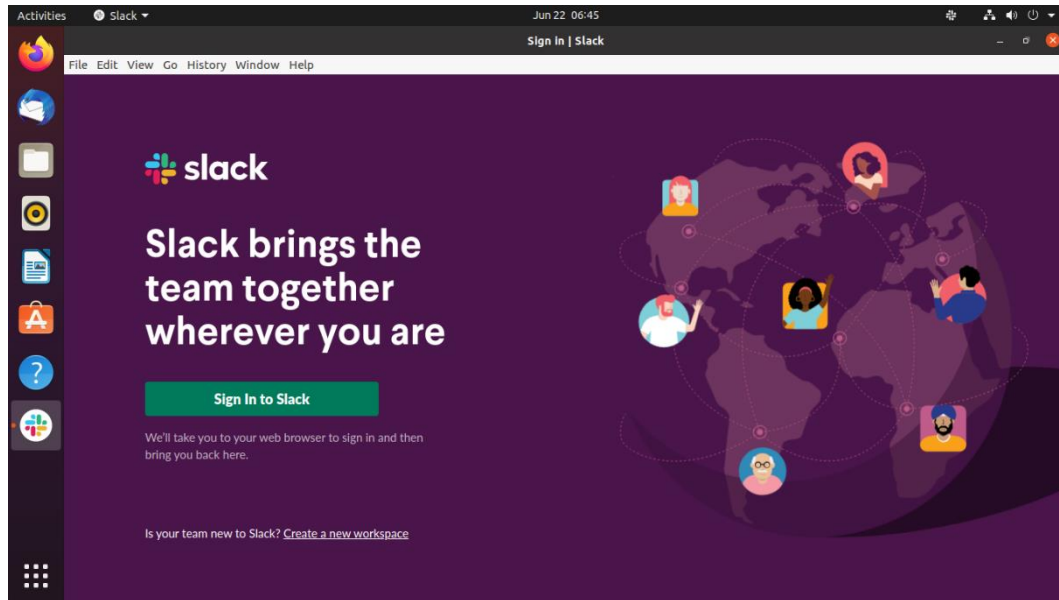
4.5. Konfigurasi Komputer System Administrator

Pada tahap ini, dilakukan instalasi dan konfigurasi pada komputer *system administrator*. Pada penelitian ini, komputer *system administrator* berfungsi sebagai komputer yang untuk melakukan konfigurasi dan menerima notifikasi dari serangan yang diterima web server melalui *slack*. Untuk melakukan instalasi *slack*, langkah awal yaitu memasukkan perintah berikut pada terminal dengan hak akses *root*:

“apt install snapd”

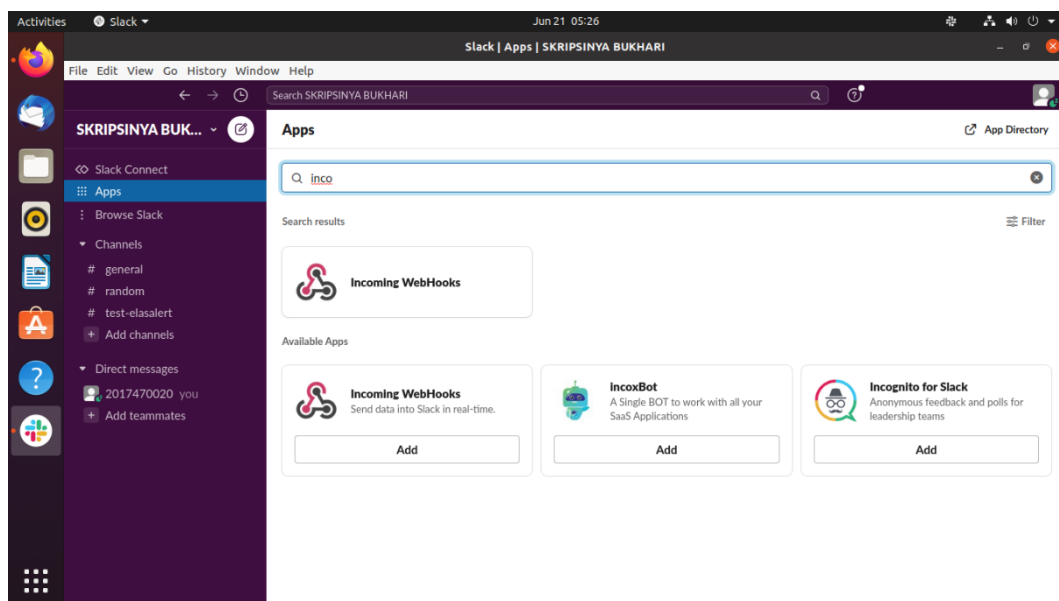
“snap install slack --classic”

Kedua perintah di atas berfungsi untuk melakukan instalasi *slack* melalui *snap store*. Adapun tampilan awal dari aplikasi *slack* tertera pada gambar 4.27.



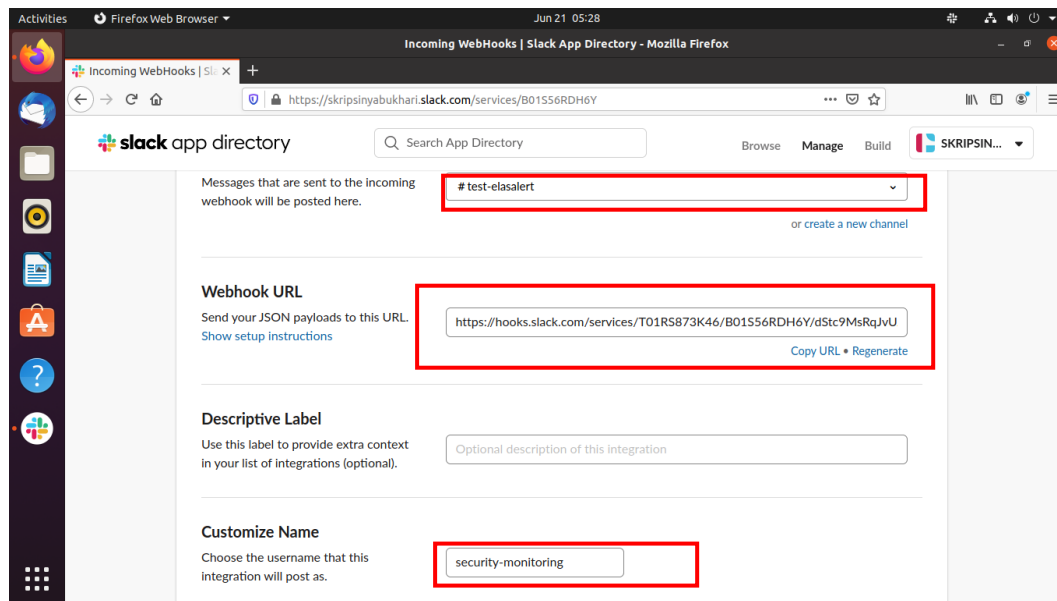
Gambar 4. 27 Tampilan Awal *Slack*

Gambar 4.27 di atas memperlihatkan tampilan awal dari aplikasi *slack* setelah berhasil di *install*. Selanjutnya buat akun pada *slack* dan buat *workspace* untuk menerima dan mengirim pesan. Setelah itu tambahkan *incoming webhooks* yang ada pada aplikasi *slack*, *incoming webhooks* berfungsi sebagai alamat untuk notifikasi *elastalert*. Adapun contoh tampilan dari proses penambahan *incoming webhooks* pada *slack* tertera pada gambar 4.28.



Gambar 4. 28 Penambahan *Incoming Webhooks* pada *Slack*

Gambar 4.28 di atas memperlihatkan penambahan *incoming webhooks* pada *slack*. Setelah ditambahkan, lakukan konfigurasi pada *incoming webhooks* sebagaimana yang tertera pada gambar 4.29.



Gambar 4. 29 Konfigurasi *Incoming Webhooks* pada *Slack*

Gambar 4.29 di atas memperlihatkan tampilan konfigurasi *incoming webhooks* pada *slack*. Pada tahap ini, konfigurasi *channel* untuk menerima pesan dan nama pengirim serta salin *webhook url* untuk digunakan pada konfigurasi *elastalert*.

4.6. Konfigurasi ELK Stack Server

Pada tahap ini dilakukan instalasi dan konfigurasi serta pembuatan filter *logstash* untuk menerima, memproses dan memvisualisasikan log yang dikirim web server.

4.6.1. Instalasi Perangkat Lunak pada ELK Stack Server

Pada tahap ini dilakukan instalasi perangkat lunak yang dibutuhkan ELK Stack server untuk membangun log *event management web application firewall* meliputi *Elasticsearch*, *Logstash*, *Kibana*, *Nginx* Web Server dan *Elasalert*. Untuk melakukan instalasi perangkat lunak yang dibutuhkan tersebut, penulis membuat bash script sebagai berikut :

```
#!/bin/bash

#Jalankan script ini dengan root privilege
#Import Elasticstack repository
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch --no-check-
certificate | sudo apt-key add -
echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee -a
/etc/apt/sources.list.d/elastic-7.x.list
apt update

#Install Elasticsearch
apt install elasticsearch -y

#Install Kibana
apt install kibana -y

#Install Logstash
apt install openjdk-11-jdk -y
apt install logstash -y

#Install Nginx
apt install nginx -y

#Install Elastalert
apt install python3-pip python3-dev libffi-dev libssl-dev -y
pip install elastalert
git clone https://github.com/Yelp/elastalert.git
cd elastalert
pip install "setuptools>=11.3"
python3 setup.py install
pip install "elasticsearch>=5.0.0"
cp config.yaml.example config.yaml

echo
"=====
echo "Seluruh Aplikasi ELK Stack Server berhasil di install"
echo
"=====
```

Bash script di atas berfungsi untuk melakukan pengunduhan dan instalasi perangkat lunak yang dibutuhkan pada *ELK Stack Server* meliputi *elasticsearch*, *logstash*, *kibana*, *nginx* dan *elastalert*. *Bash script* tersebut disimpan dengan nama *elk-server-app.sh*, selanjutnya buat *script* di atas *executable* dan jalankan *script* tersebut dengan perintah :

“`chmod +x elk-server-app.sh && ./elk-server-app.sh`”

4.6.2. Konfigurasi Elasticsearch

Setelah melakukan instalasi perangkat lunak, pada tahap ini dilakukan konfigurasi pada *elasticsearch*. *Elasticsearch* berfungsi untuk menyimpan log yang nantinya akan divisualisasikan oleh *kibana*. Agar dapat berfungsi, konfigurasi file `/etc/elasticsearch/elasticsearch.yml` sebagaimana yang tertera pada gambar 4.30.

```

GNU nano 4.8 /etc/elasticsearch/elasticsearch.yml
# Lock the memory on startup:
#
#bootstrap.memory_lock: true
#
# Make sure that the heap size is set to about half the memory available
# on the system and that the owner of the process is allowed to use this
# limit.
#
# Elasticsearch performs poorly when the system is swapping the memory.
#
# ----- Network -----
#
# By default Elasticsearch is only accessible on localhost. Set a different
# address here to expose this node on the network:
#
network.host: localhost
#
# By default Elasticsearch listens for HTTP traffic on the first free port it
# finds starting at 9200. Set a specific HTTP port here:
#
http.port: 9200
#
# For more information, consult the network module documentation.
#
# ----- Discovery -----
#
# Pass an initial list of hosts to perform discovery when this node is started:
# The default list of hosts is ["127.0.0.1", "[::1]"]
#
#discovery.seed_hosts: ["host1", "host2"]
#
# Bootstrap the cluster using an initial set of master-eligible nodes:
#
^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text    ^J Justify    ^C Cur Pos    M-U Undo
^X Exit       ^R Read File  ^_ Replace    ^U Paste Text ^T To Spell   ^G Go To Line  M-E Redo

```

Gambar 4. 30 Konfigurasi *Elasticsearch*

Gambar 4.30 di atas memperlihatkan konfigurasi elasticsearch yang dilakukan pada ELK *Stack* server. Konfigurasi pada bagian network pada gambar di atas berfungsi untuk mendefinisikan *host* dan *port* yang digunakan *elasticsearch*. Simpan konfigurasi tersebut kemudian jalankan *elasticsearch* dengan perintah :

“`systemctl start elasticsearch`”

4.6.3. Pembuatan Rules Logstash

Pada tahap ini dilakukan pembuatan *rules logstash* pada ELK *Stack* server. *Rules logstash* berfungsi melakukan filter terhadap log yang dikirimkan web server sebelum di simpan pada *elasticsearch*. Pada *rules Logstash* terdapat tahapan yakni *inputs*, *filters* dan *outputs* yang disebut *Logstash Pipeline*. Berikut ini merupakan *rules* yang akan digunakan pada tahap input :

```
input {
  beats {
    port => 5044
  }
}
```

Pada tahap *input*, *logstash* akan mencari sumber dari log yang akan difilter. Pada penelitian ini digunakan *tools filebeat* untuk mengirimkan log ke *logstash* melalui *port default logstash* yakni 5044. Setelah melalui tahap *input*, selanjutnya log akan melalui tahap filter untuk melakukan penguraian data. Pada penelitian ini digunakan *apache* sebagai web server, oleh karena itu serangan yang terdeteksi oleh *modsecurity* akan dicatat pada *apache error.log*. Berikut ini merupakan contoh dari log yang dihasilkan *ModSecurity* yang ada pada *file apache error.log* :

```
[Wed Apr 28 12:39:04.036621 2021] [:error] [pid 1134] [client
192.168.1.7:56844] ModSecurity: Warning. detected SQLi using libinjection.
[file "/etc/apache2/modsecurity.d/owasp-crs/rules/REQUEST-942-
APPLICATION-ATTACK-SQLI.conf"] [line "45"] [id "942100"] [rev "" ] [msg
"SQL Injection Attack Detected via libinjection"] [data "Matched Data: s&l
found within ARGS:id: 99' or 1=1"] [severity "2"] [ver "OWASP_CRS/3.2.0"]
[maturity "0"] [accuracy "0"] [hostname "192.168.1.101"] [uri "/" ] [unique_id
"161961354428.185428"] [ref "v9,10"]
```

Sebelum data disimpan pada *elasticsearch* dan divisualisasikan pada *kibana*, log yang diterima akan diurai menggunakan *grok* filter. *Grok* filter dapat melakukan *parsing* log data yang tidak terstruktur menjadi terstruktur dengan menggunakan *regular expression* (regex). Pada penelitian ini digunakan *Grok Debugger* yang ada pada *Kibana* untuk menguji *rules* yang dibuat pada *Logstash*.

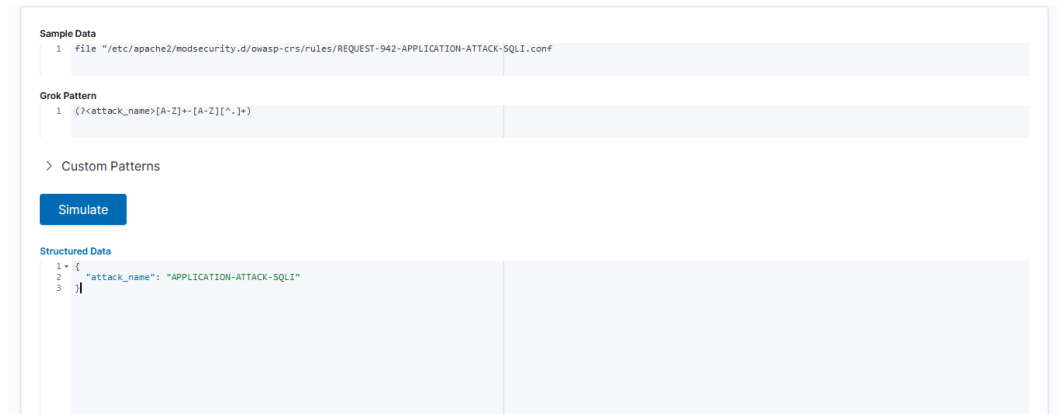
Adapun *grok* filter untuk mengurai waktu, jenis log, IP *attacker* dan pesan *ModSecurity* tertera pada gambar 4.31.

Gambar 4. 31 *Grok* Filter Waktu, Jenis Log, IP *Attacker* dan Pesan *ModSecurity*

Gambar 4.31 memperlihatkan *grok* filter yang digunakan untuk mengurai data berdasarkan waktu, jenis log, IP *attacker* dan pesan *ModSecurity*. Selanjutnya dibuat *grok* filter untuk mengurai *file rules modsecurity* yang mendeteksi serangan yang diterima. Adapun *grok* filter tersebut tertera pada gambar 4.32.

Gambar 4. 32 *Grok* Filter *File Rules ModSecurity*

Gambar 4.32 memperlihatkan *grok* filter yang digunakan untuk mengurai *file rules modsecurity* yang ada pada pesan *ModSecurity*. Selanjutnya dibuat filter untuk mengurai nama serangan agar data yang nantinya ditampilkan lebih spesifik. Adapun *grok* filter untuk mengurai nama serangan tertera pada gambar 4.33..



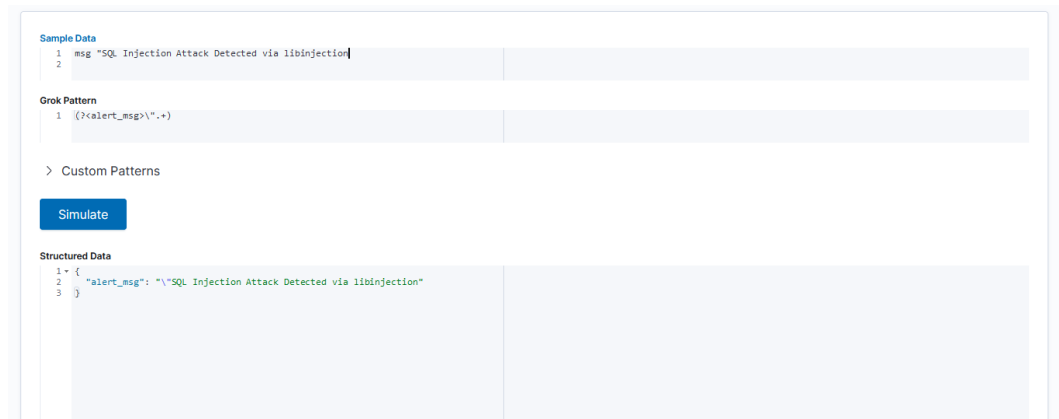
Gambar 4. 33 Grok Filter Nama Serangan

Gambar 4.33 di atas memperlihatkan *grok* filter yang digunakan untuk mengurai nama serangan. Dengan menggunakan *grok* filter di atas akan didapati hasil data berupa nama serangan yang dialami web server secara spesifik. Selanjutnya dibuat filter untuk mengurai pesan yang berisikan keterangan serangan yang diterima. Adapun *grok* filter untuk mengurai pesan tersebut tertera pada gambar 4.34.



Gambar 4. 34 Grok Filter Pesan Keterangan Serangan

Gambar 4.34 di atas memperlihatkan *grok* filter yang digunakan untuk mengurai pesan keterangan serangan. Selanjutnya dibuat *grok* filter untuk mengambil isi dari pesan keterangan serangan agar data yang ditampilkan lebih spesifik. Adapun filter untuk mengurai keterangan serangan tersebut tertera pada gambar 4.35.



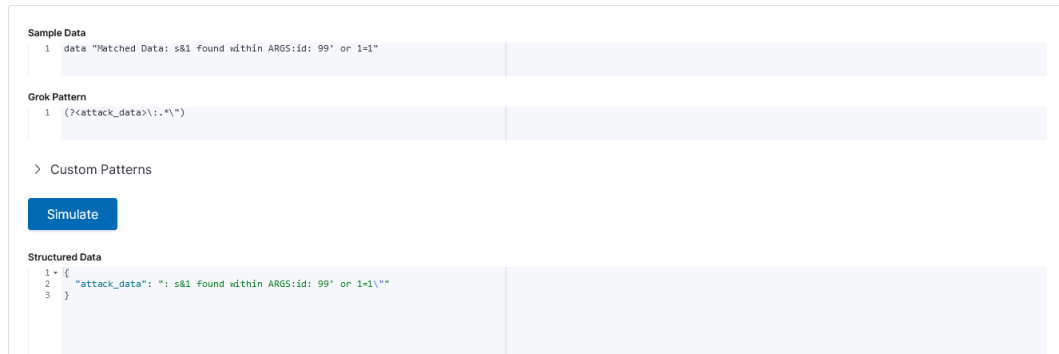
Gambar 4. 35 Grok Filter Keterangan Serangan

Gambar 4.35 di atas memperlihatkan *grok* filter yang digunakan untuk mengurai keterangan serangan yang diterima. Dengan *grok* filter di atas keterangan yang ditampilkan akan lebih spesifik. Selanjutnya dibuat filter yang digunakan untuk mengurai data pola serangan yang dihadapi. Adapun *grok* filter untuk mengurai data pola serangan yang dihadapi tertera pada gambar 4.36.



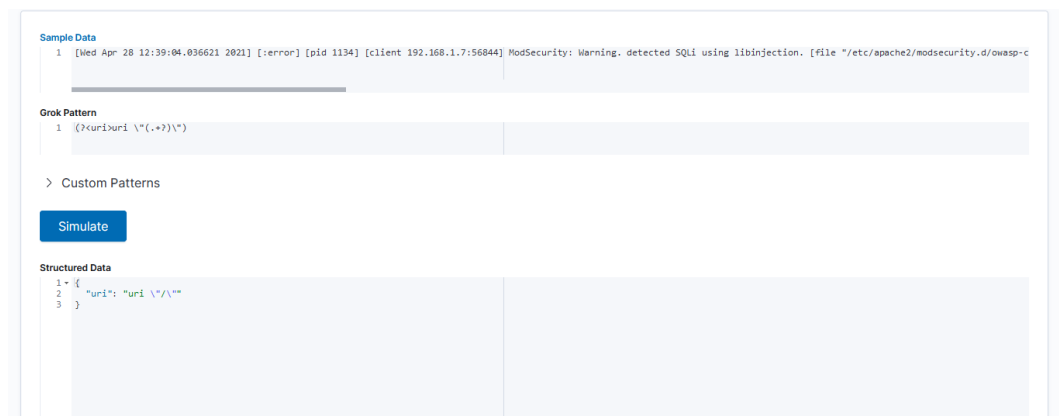
Gambar 4. 36 Grok Filter Data Pola Serangan

Gambar 4.36 memperlihatkan *grok* filter yang digunakan untuk mengurai data pola serangan dari pesan *modsecurity*. Selanjutnya dibuat filter untuk mengambil isi dari data pola serangan. Adapun *grok* filter yang digunakan untuk mengurai pola serangan tertera pada gambar 4.37.



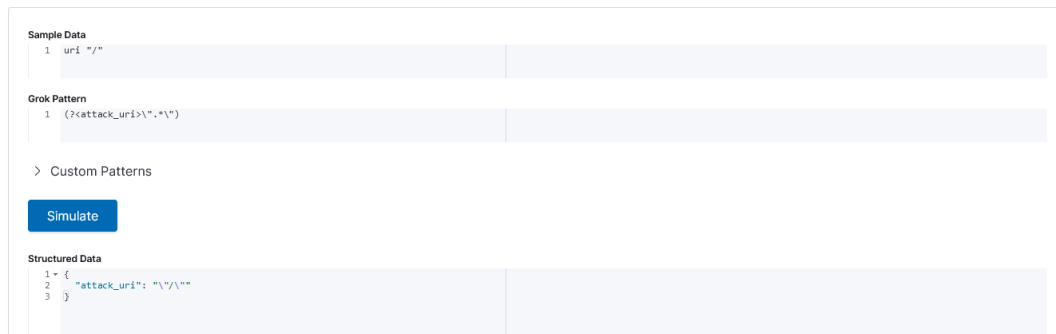
Gambar 4. 37 Grok Filter Pola Serangan

Gambar 4.37 memperlihatkan *grok* filter yang digunakan untuk mengurai pola serangan yang diterima. Dengan *grok* filter di atas data yang ditampilkan akan lebih spesifik dan membuang teks yang tidak diperlukan. Selanjutnya dibuat filter untuk mengurai halaman yang mengalami serangan. Adapun *grok* filter yang digunakan untuk mengurai halaman yang diserang tertera pada gambar 4.38.



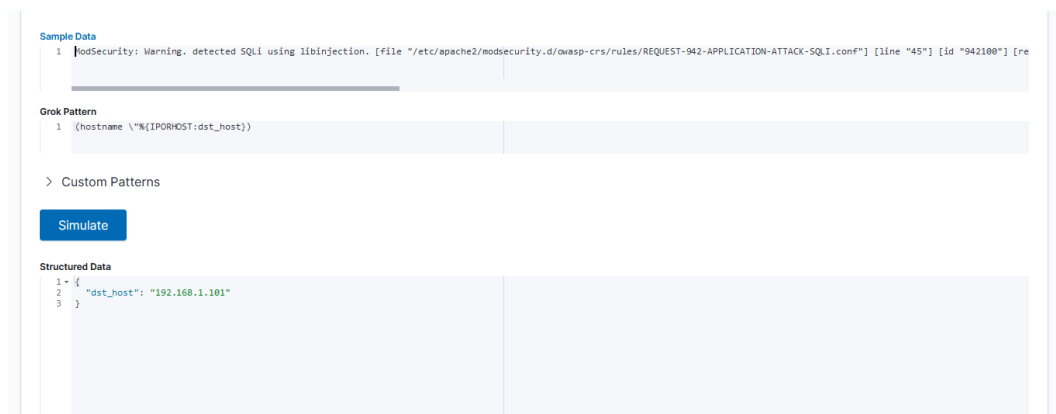
Gambar 4. 38 Grok Filter Data Halaman yang Diserang

Gambar 4.38 di atas memperlihatkan *grok* filter yang digunakan untuk mengurai data halaman yang diserang dari pesan *modsecurity*. Selanjutnya dibuat filter untuk mengambil isi dari data halaman yang diserang. Adapun *grok* filter yang digunakan untuk mengurai halaman yang diserang tertera pada gambar 4.39.



Gambar 4. 39 Grok Filter Halaman yang Diserang

Gambar 4.39 memperlihatkan *grok* filter yang digunakan untuk mengurai halaman yang diserang. Dengan *grok* filter di atas data yang ditampilkan akan lebih spesifik dan membuang teks yang tidak diperlukan. Selanjutnya dibuat filter untuk mengurai alamat IP dari web server yang mengalami serangan. Adapun *grok* filter yang digunakan untuk mengurai alamat IP web server tertera pada gambar 4.40.



Gambar 4. 40 Grok Filter IP Address Web Server

Gambar 4.40 di atas memperlihatkan *grok* filter yang digunakan untuk mengurai IP *address* web server. Langkah selanjutnya yang dilakukan yaitu dengan menggabungkan setiap *rules* yang dibuat pada tahap filter. Berikut ini merupakan seluruh *rules* yang akan digunakan pada tahap filter:

```
filter{
#Mengurai log lengkap ke dalam beberapa bagian yaitu time_stamp, log_level,
ip_attacker dan modsec_message
  grok{
    match => { "message" =>
"(?<time_stamp>%{MONTH:bulan})\s%{MONTHDAY:tanggal}\s%{TIME:wak
```

```

tu}\s%{YEAR:tahun})\]
\[\.%{LOGLEVEL:log_level}.*client\s%{IPORHOST:ip_attacker}:\d+\]\s%{GR
EEDYDATA:modsec_message}" }
}

#Ekstrak attack_file dari modsec_message
grok{
  match => { "modsec_message" => "(?<attack_file>file \"(.+).conf)" }
}

#Ekstrak attack_name dari attack_file
grok{
  match => { "attack_file" => "(?<attack_name>[A-Z]+-[A-Z][^.]*)" }
}

#Ekstrak msg dari modsec_message
grok{
  match => { "modsec_message" => "(?<msg>msg \"(.+?)\"" }
}

#Ekstrak alert_msg dari msg
grok{
  match => { "msg" => "(?<alert_msg>\".+\"" }
}

#Ekstrak data dari modsec_message
grok{
  match => { "modsec_message" => "(?<data>data \"(.+?)\"" }
}

#Ekstrak attack_data dari data
grok{
  match => { "data" => "(?<attack_data>\".*\"" }
}

#Ekstrak uri dari modsec_message
grok{
  match => { "modsec_message" => "(?<uri>uri \"(.+?)\"" }
}

#Ekstrak uri_attack dari uri
grok{
  match => { "uri" => "(?<attack_uri>\".*\"" }
}

#Ekstrak hostname dari modsec_message
grok{
  match => { "modsec_message" => "(hostname \"%{IPORHOST:dst_host})" }
}

```

```

}

#Hapus karakter yang tidak diperlukan
mutate{
  gsub => [
    "attack_uri", "[\\]", "",
    "attack_data", "[\\:]", "",
    "alert_msg", "[\\]", ""
  ]
  remove_field => ["modsec_message", "attack_file", "msg", "data", "uri",
"message"]
}
}

```

Tahap terakhir pada *logstash pipeline* adalah tahap *output*, pada tahap ini hasil dari data yang telah diurai sebelumnya akan disimpan pada *elasticsearch*. Berikut ini merupakan rules yang akan digunakan pada tahap *output* :

```

output {
  elasticsearch {
    hosts => ["localhost:9200"]
    manage_template => false
    index => "logstash-waf-%{+YYYY.MM}"
  }
}

```

Setelah membuat *input*, *filter* dan *output* dari *logstash pipeline*, langkah selanjutnya yaitu menggabungkan seluruh rules tersebut dalam satu file. Buat *rules logstash* dengan nama *modsec-filter.conf* pada folder */etc/logstash/conf.d* dengan nama *file* *modsec-filter.conf* dan isi *file* sebagai berikut :

```

input{
  beats{
    port => 5044
  }
}
filter{
  #Mengurai log lengkap ke dalam beberapa bagian yaitu time_stamp, log_level,
ip_attacker dan modsec_message
  grok{
    match          =>          {          "message"          =>
"(?<time_stamp>%{MONTH:bulan})\s%{MONTHDAY:tanggal}\s%{TIME:wak
tu}\s%{YEAR:tahun})\]"

```

```

\[\:%{LOGLEVEL:log_level}.*client\s%{IPORHOST:ip_attacker}:\d+\]\s%{GR
EEDYDATA:modsec_message}" }
}

#Ekstrak attack_file dari modsec_message
grok{
  match => { "modsec_message" => "(?<attack_file>file \"(.+).conf)" }
}

#Ekstrak attack_name dari attack_file
grok{
  match => { "attack_file" => "(?<attack_name>[A-Z]+-[A-Z][^.]*)" }
}

#Ekstrak msg dari modsec_message
grok{
  match => { "modsec_message" => "(?<msg>msg \"(.+?)\"" }
}

#Ekstrak alert_msg dari msg
grok{
  match => { "msg" => "(?<alert_msg>\".+)" }
}

#Ekstrak data dari modsec_message
grok{
  match => { "modsec_message" => "(?<data>data \"(.+?)\"" }
}

#Ekstrak attack_data dari data
grok{
  match => { "data" => "(?<attack_data>\".*\"" }
}

#Ekstrak uri dari modsec_message
grok{
  match => { "modsec_message" => "(?<uri>uri \"(.+?)\"" }
}

#Ekstrak uri_attack dari uri
grok{
  match => { "uri" => "(?<attack_uri>\".*\"" }
}

#Ekstrak hostname dari modsec_message
grok{
  match => { "modsec_message" => "(hostname \"%{IPORHOST:dst_host})" }
}

```

```

#Hapus karakter yang tidak diperlukan
mutate{
  gsub => [
    "attack_uri", "[\\]", "",
    "attack_data", "[\\:]", "",
    "alert_msg", "[\\]", ""
  ]
  remove_field => ["modsec_message", "attack_file", "msg", "data", "uri",
"message"]
}
}

output {
  elasticsearch {
    hosts => ["localhost:9200"]
    manage_template => false
    index => "logstash-waf-%{+YYYY.MM}"
  }
}

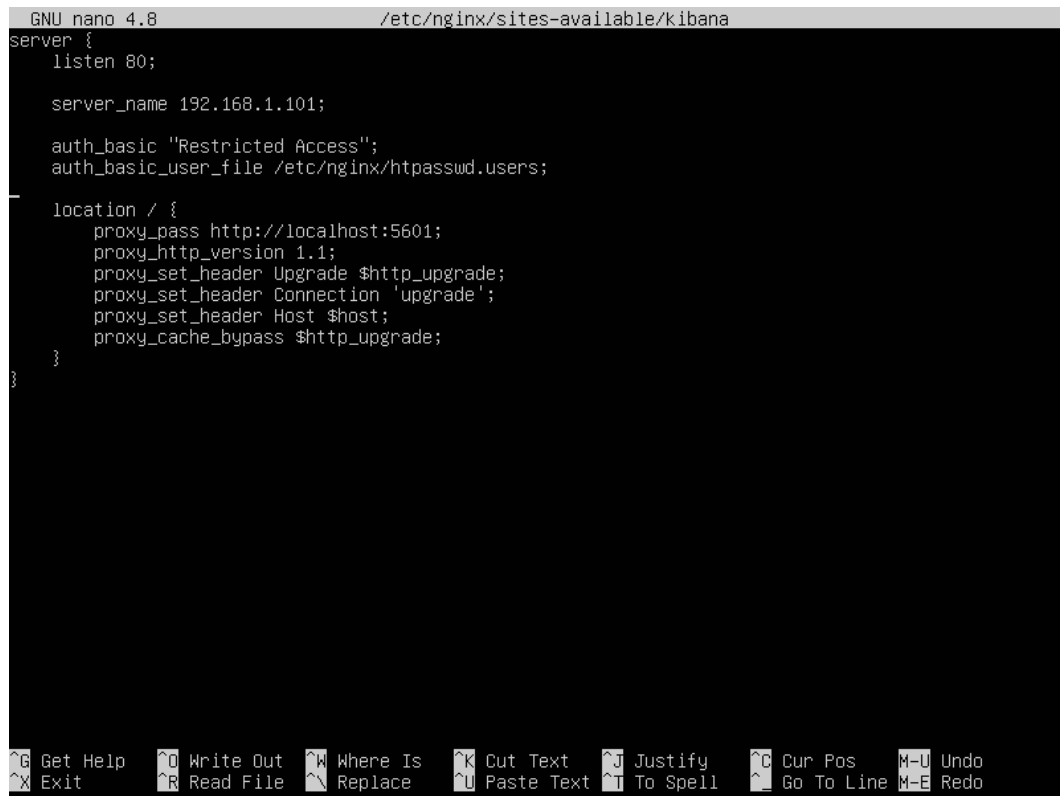
```

Setelah membuat dan menyimpan *file* tersebut, jalankan *logstash* dengan perintah :

“systemctl start logstash”.

4.6.4. Konfigurasi Nginx

Sebelum melakukan konfigurasi pada kibana, pada tahap ini dilakukan konfigurasi pada nginx. Konfigurasi *nginx* pada tahap ini berfungsi untuk mengarahkan server HTTP *traffic* pada *kibana* yang berada pada *port* 5601. Selain itu konfigurasi *nginx* ini juga berfungsi untuk menghadirkan autentikasi untuk mengakses kibana. Adapun konfigurasi pada nginx tersebut tertera pada gambar 4.41.



```

GNU nano 4.8 /etc/nginx/sites-available/kibana
server {
    listen 80;

    server_name 192.168.1.101;

    auth_basic "Restricted Access";
    auth_basic_user_file /etc/nginx/htpasswd.users;

    location / {
        proxy_pass http://localhost:5601;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }
}

```

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos M-U Undo
 ^X Exit ^R Read File ^N Replace ^U Paste Text ^T To Spell ^G Go To Line M-E Redo

Gambar 4. 41 Konfigurasi *Nginx*

Gambar 4.41 di atas memperlihatkan konfigurasi yang dilakukan pada *nginx*. Konfigurasi di atas disimpan dengan nama *kibana* pada *directory* */etc/nginx/sites-available*. Selanjutnya masukan perintah berikut untuk menambahkan kredensial untuk mengakses *kibana* :

```
“echo "admin:`openssl passwd -apr1`" | sudo tee -a /etc/nginx/htpasswd.users”
```

Setelah itu buat *link file kibana* yang telah dibuat sebelumnya pada direktori *sites-available* ke *sites-enabled* agar dapat diakses dan kemudian *restart nginx* dengan perintah :

```
“ln -s /etc/nginx/sites-available/kibana /etc/nginx/sites-enabled/”
```

```
“systemctl restart nginx”
```

4.6.5. Konfigurasi Kibana

Kibana berfungsi untuk melakukan visualisasi terhadap data-data yang telah disimpan pada *elasticsearch*. Untuk melakukan visualisasi tersebut perlu

dilakukan konfigurasi pada `/etc/kibana/kibana.yml` sebagaimana yang tertera pada gambar 4.42.

```

GNU nano 4.8 /etc/kibana/kibana.yml Modified
# Kibana is served by a back end server. This setting specifies the port to use.
server.port: 5601

# Specifies the address to which the Kibana server will bind. IP addresses and host names are both
# The default is 'localhost', which usually means remote machines will not be able to connect.
# To allow connections from remote users, set this parameter to a non-loopback address.
server.host: "localhost"

# Enables you to specify a path to mount Kibana at if you are running behind a proxy.
# Use the `server.rewriteBasePath` setting to tell Kibana if it should remove the basePath
# from requests it receives, and to prevent a deprecation warning at startup.
# This setting cannot end in a slash.
#server.basePath: ""

# Specifies whether Kibana should rewrite requests that are prefixed with
# `server.basePath` or require that they are rewritten by your reverse proxy.
# This setting was effectively always `false` before Kibana 6.3 and will
# default to `true` starting in Kibana 7.0.
#server.rewriteBasePath: false

# Specifies the public URL at which Kibana is available for end users. If
# `server.basePath` is configured this URL should end with the same basePath.
#server.publicBaseUrl: ""

# The maximum payload size in bytes for incoming server requests.
#server.maxPayload: 1048576

# The Kibana server's name. This is used for display purposes.
#server.name: "your-hostname"

# The URLs of the Elasticsearch instances to use for all your queries.
elasticsearch.hosts: ["http://localhost:9200"]

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos M-U Undo
^X Exit ^R Read File ^N Replace ^U Paste Text ^T To Spell ^G Go To Line M-E Redo

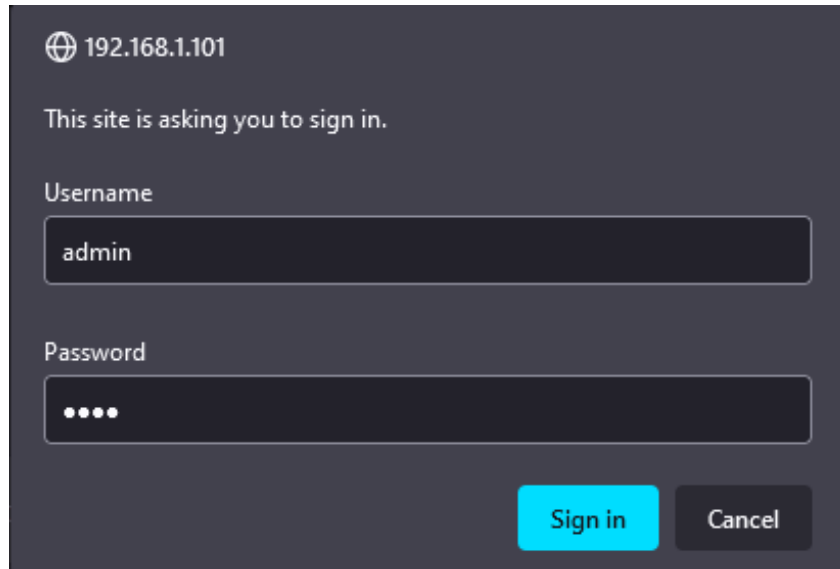
```

Gambar 4. 42 Konfigurasi Kibana

Gambar 4.42 di atas memperlihatkan konfigurasi *kibana* pada *ELK Stack* server. Konfigurasi pada gambar di atas berfungsi untuk mendefinisikan *port*, *host* dan *elasticsearch host* agar dapat terhubung dengan *kibana*. Kemudian jalankan *kibana* dengan perintah :

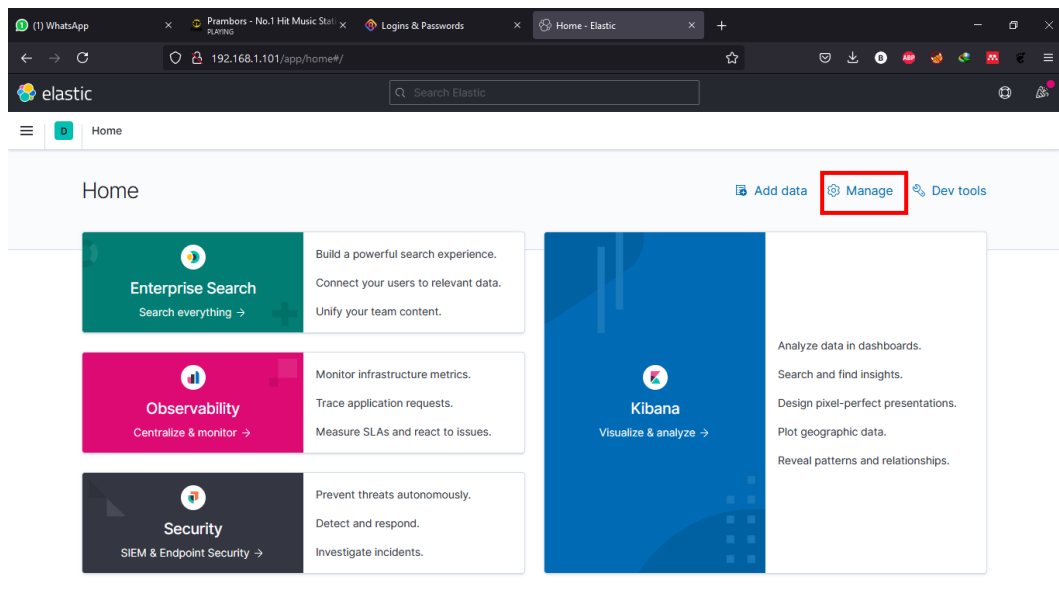
“systemctl start kibana”

Selanjutnya akses *kibana dashboard* melalui web browser dengan memasukkan IP *address* dari *ELK Stack* server pada *url*. Untuk dapat mengakses *kibana dashboard*, pengguna akan diminta memasukkan *username* dan *password*. Adapun contoh tampilan autentikasi pada *kibana dashboard* tertera pada gambar 4.43.



Gambar 4. 43 Autentikasi *Kibana Dashboard*

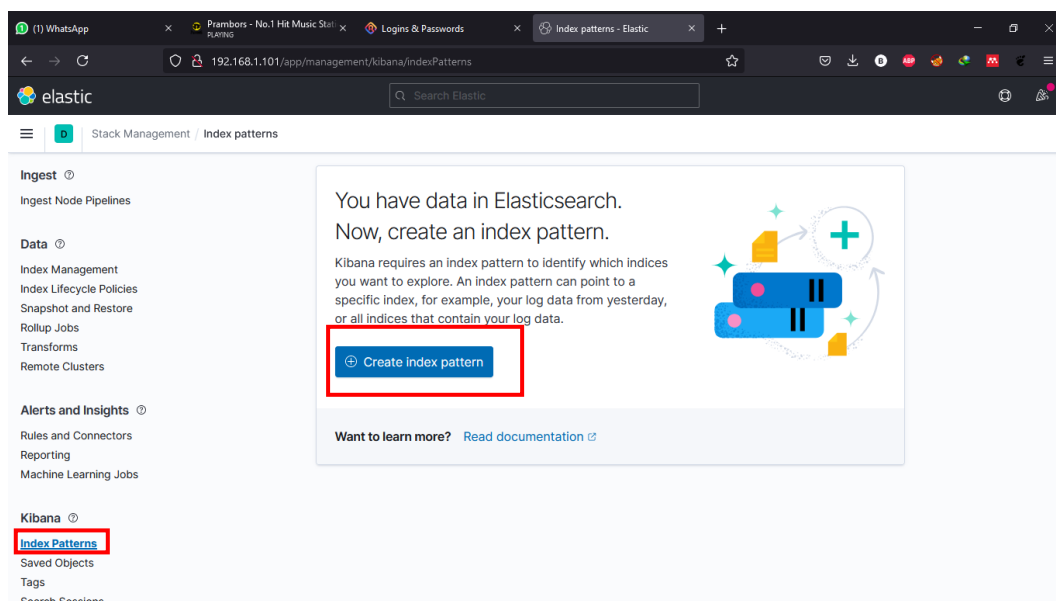
Gambar 4.43 memperlihatkan tampilan autentikasi untuk dapat mengakses *kibana dashboard*. Pengguna diharuskan memasukkan *username* dan *password* yang valid agar dapat mengakses *kibana dashboard*. Setelah berhasil melakukan *login*, selanjutnya akan tampil *kibana dashboard* pada web browser. Adapun contoh tampilan *kibana dashboard* tertera pada gambar 4.44.



Gambar 4. 44 Tampilan *Kibana Dashboard*

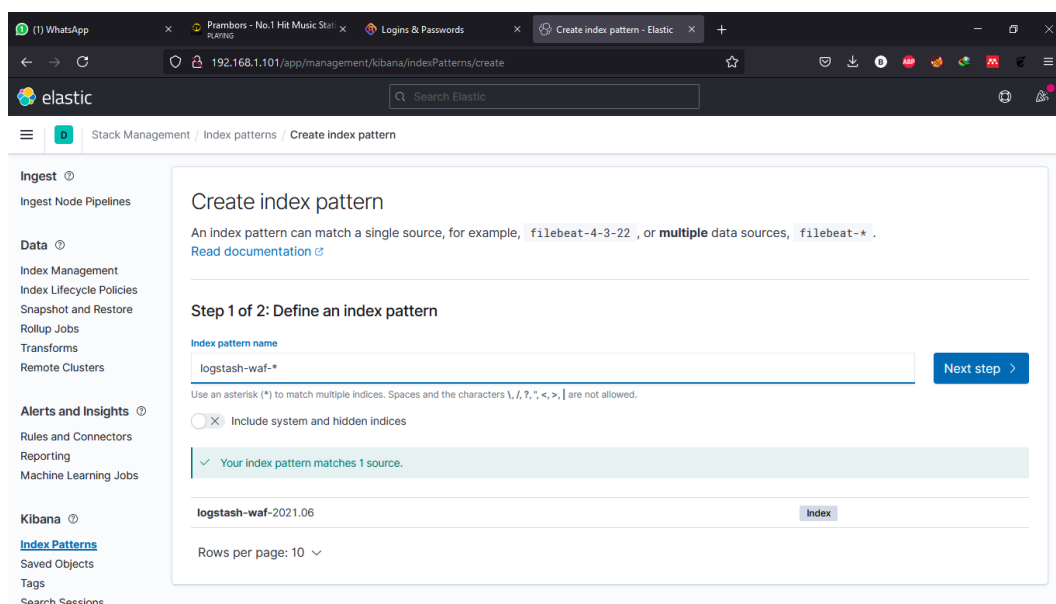
Gambar 4.44 di atas memperlihatkan tampilan awal dari *kibana dashboard*. Untuk melakukan visualisasi log, klik *manage* pada kanan atas untuk

membuat *index pattern*. Selanjutnya klik *index pattern* dan *create index pattern* sebagaimana yang tertera pada gambar 4.45.



Gambar 4. 45 Pembuatan *Index Pattern* pada *Kibana*

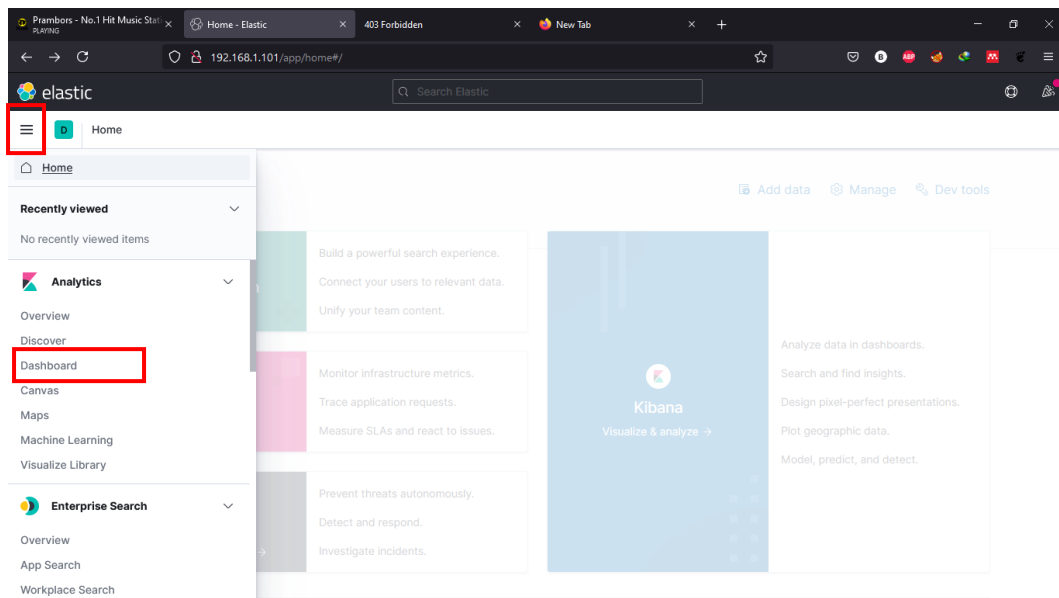
Gambar 4.45 di atas memperlihatkan pembuatan *index pattern* pada *kibana*. Selanjutnya masukkan nama *index pattern* sesuai dengan nama *output* yang telah ditetapkan pada *rules logstash*. Adapun contoh tampilan dari pembuatan nama *index pattern* tertera pada gambar 4.46.



Gambar 4. 46 Pembuatan Nama *Index Pattern* pada *Kibana*

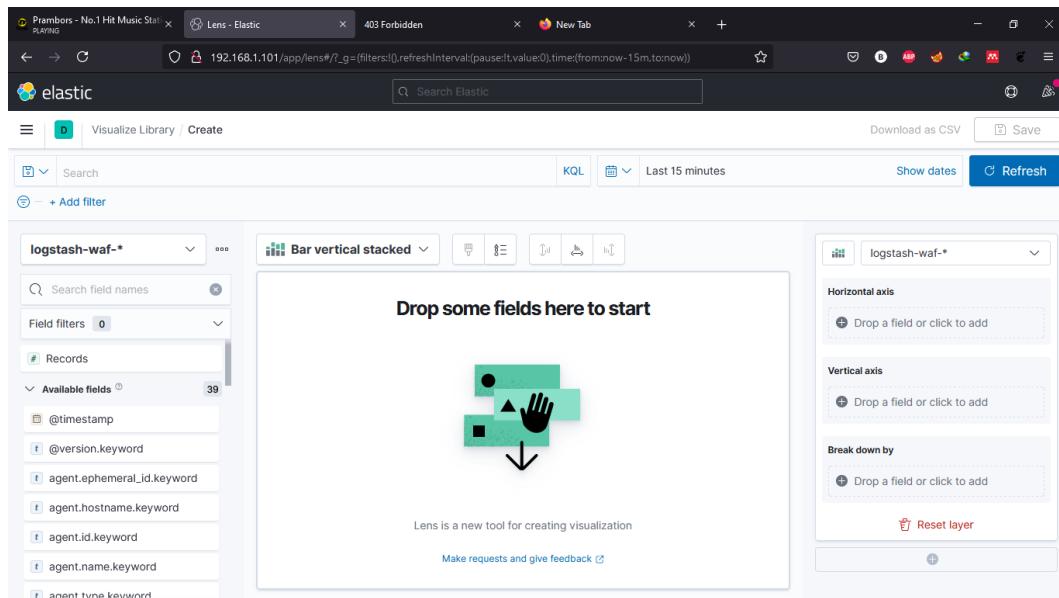
Gambar 4.46 di atas memperlihatkan tampilan pembuatan nama *index pattern* pada *kibana*. Setelah membuat *index pattern*, langkah selanjutnya yaitu membuat visualisasi log ke dalam berbagai bentuk. Pada penelitian ini, penulis membuat 3 jenis visualisasi yaitu *pie chart*, *bar chart* dan *table*.

Untuk melakukan visualisasi langkah pertama yaitu dengan membuka sidebar yang ada pada *kibana*, kemudian pilih submenu *dashboard*. Adapun contoh dari pemilihan submenu tersebut tertera pada gambar 4.47.



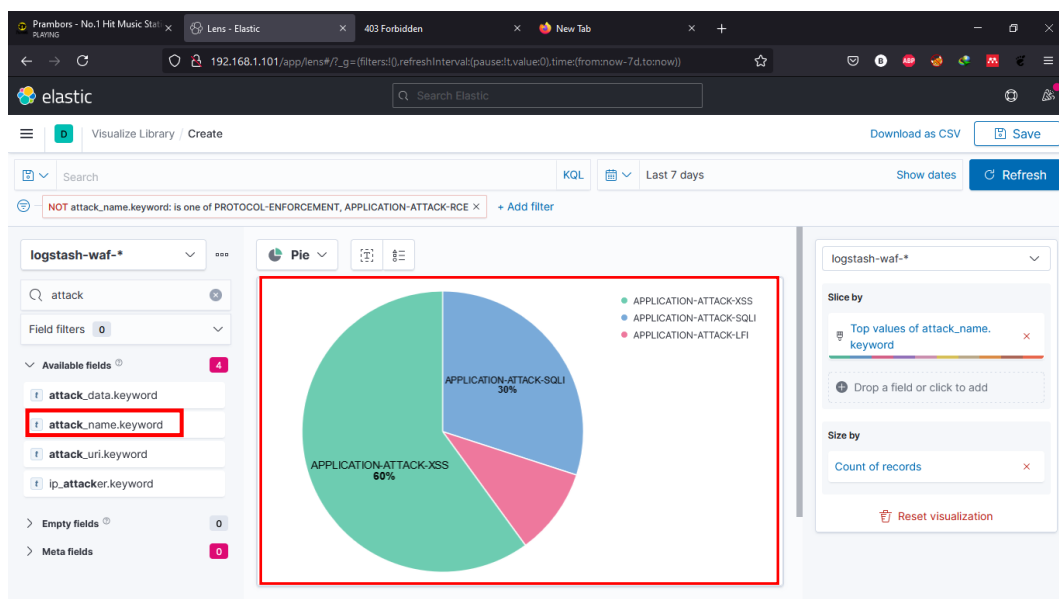
Gambar 4. 47 Tampilan Pilih Sub Menu *Dashboard* pada *Kibana*

Gambar 4.47 memperlihatkan tampilan saat memilih sub menu *dashboard* pada *kibana*. Setelah memilih menu tersebut, pilih *Create New Dashboard* kemudian pilih *Create Visualization*. Adapun tampilan untuk membuat visualisasi tertera pada gambar 4.48.



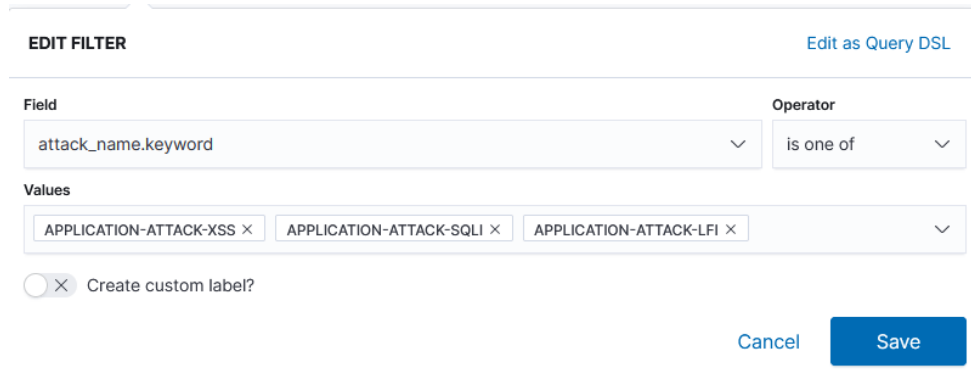
Gambar 4. 48 Tampilan Pembuatan Visualisasi pada *Kibana*

Gambar 4.48 di atas memperlihatkan tampilan awal untuk membuat visualisasi pada *kibana*. Visualisasi pertama yang akan dilakukan yaitu *pie chart*, pada visualisasi ini akan menampilkan persentase nama serangan yang ditujukan kepada web server. Untuk melakukan visualisasi tersebut pilih `attack_name.keyword` kemudian drag and drop pada Lens. Adapun tampilan dari visualisasi *pie chart* tersebut tertera pada gambar 4.49.



Gambar 4. 49 Tampilan Visualisasi *Pie Chart* pada *Kibana*

Gambar 4.49 di atas memperlihatkan tampilan visualisasi *pie chart* pada *kibana*. Agar data yang tampil pada *pie chart* spesifik kepada 3 jenis serangan yang diuji cobakan, pengguna dapat menambahkan filter pada visualisasi *kibana*. Adapun penambahan filter pada visualisasi *pie chart* tertera pada gambar 4.50.

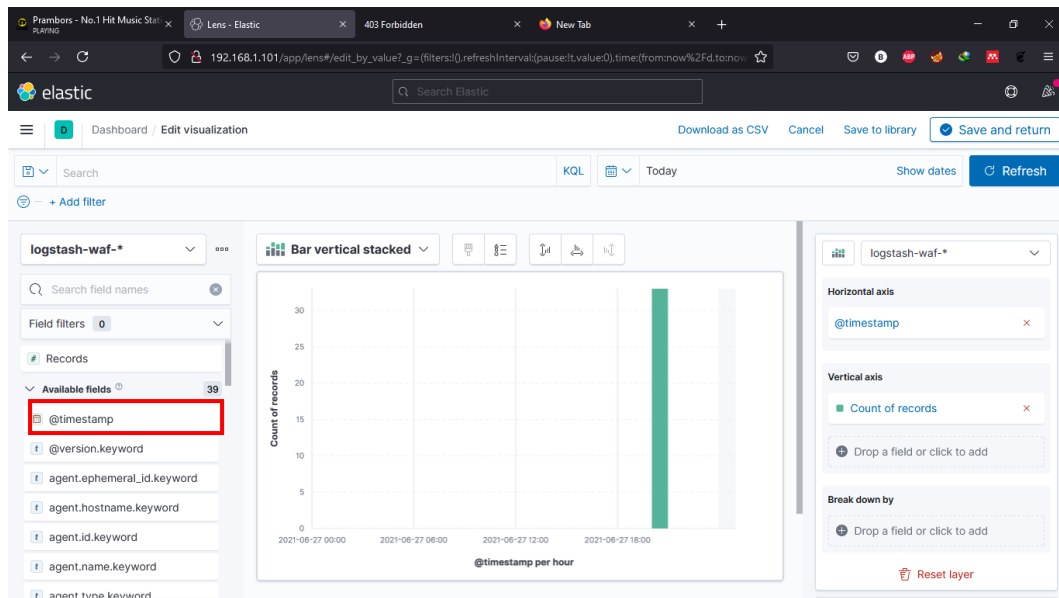


The image shows a 'Kibana' filter configuration window titled 'EDIT FILTER'. At the top right, there is a link 'Edit as Query DSL'. The main configuration area has two columns: 'Field' and 'Operator'. The 'Field' dropdown is set to 'attack_name.keyword'. The 'Operator' dropdown is set to 'is one of'. Below these, the 'Values' section contains three tags: 'APPLICATION-ATTACK-XSS', 'APPLICATION-ATTACK-SQLI', and 'APPLICATION-ATTACK-LFI'. At the bottom left, there is a radio button labeled 'Create custom label?'. At the bottom right, there are two buttons: 'Cancel' and 'Save'.

Gambar 4. 50 Filter Visualisasi *Pie Chart* pada *Kibana*

Gambar 4.50 memperlihatkan filter yang digunakan pada visualisasi *pie chart*. Pada filter di atas mengatur data yang ditampilkan agar spesifik kepada 3 jenis serangan yang diuji cobakan yaitu *Cross Site Scripting* (XSS), *SQL Injection* (SQLI) dan *Local File Inclusion* (LFI). Setelah menetapkan filter, langkah selanjutnya adalah tekan *Save and Return* untuk menyimpan visualisasi.

Visualisasi yang ke dua yaitu membuat visualisasi *bar chart* untuk menampilkan log yang diterima. Adapun tampilan dari visualisasi *bar chart* pada *kibana* tertera pada gambar 4.51.



Gambar 4. 51 Tampilan Visualisasi Bar *Chart* pada *Kibana*

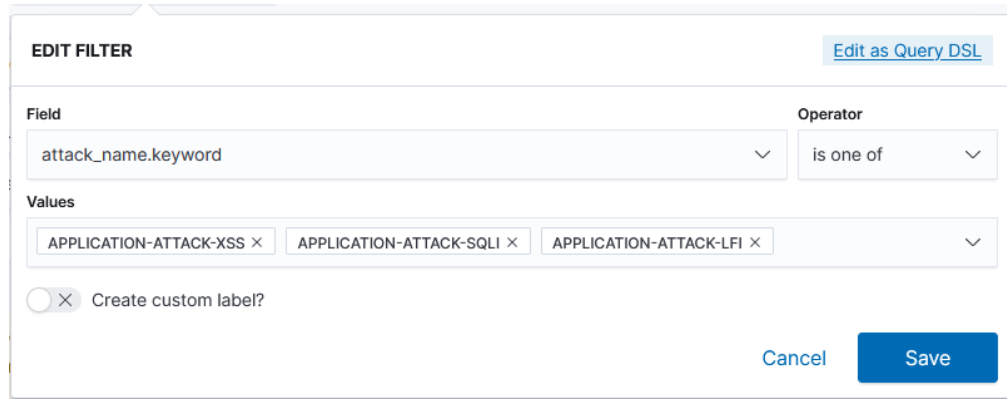
Gambar 4.51 memperlihatkan tampilan visualisasi bar *chart* pada *kibana*. Pada visualisasi ini *field* yang digunakan adalah *timestamp* untuk menampilkan seluruh log *modsecurity* yang dikirim dari web server. Visualisasi terakhir yaitu visualisasi dalam bentuk tabel. Adapun tampilan dari visualisasi tabel tertera pada gambar 4.52.

Waktu Serangan	IP Address	Keterangan	Halaman	Pola Serangan
Jun 27 13:50:24.497072 2021	192.168.1.6	SQL Injecti...	/sql/postin...	1UE1 foun...
Jun 27 13:50:24.497482 2021	192.168.1.6	Detects M...	/sql/postin...	UNION SEL...
Jun 27 13:50:24.497894 2021	192.168.1.6	Detects co...	/sql/postin...	999 UNION...
Jun 27 13:53:58.900443 2021	192.168.1.6	OS File Acc...	/index.php	etc/passw...

Gambar 4. 52 Tampilan Visualisasi Tabel pada *Kibana*

Gambar 4.52 di atas memperlihatkan tampilan untuk melakukan visualisasi tabel pada *kibana*. Pada visualisasi tabel, digunakan beberapa *field*

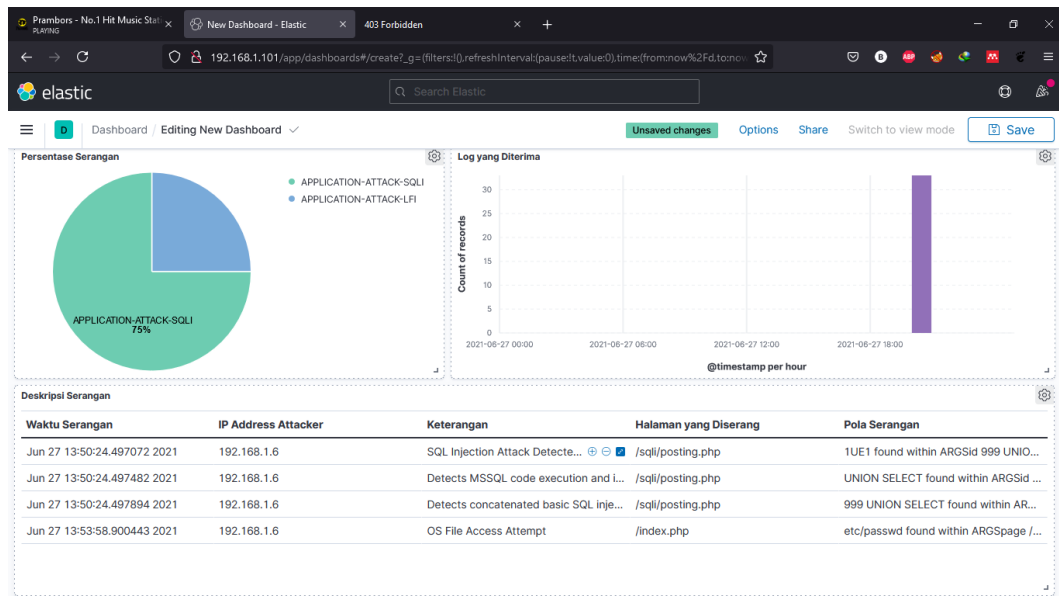
antara lain *time_stamp.keyword*, *ip_attacker.keyword*, *alert_msg.keyword*, *attack_uri.keyword* dan *attack_data.keyword*. Sama halnya dengan pie chart, data yang ditampilkan pada tabel juga perlu difilter agar lebih spesifik. Adapun filter yang digunakan pada visualisasi tabel tertera pada gambar 4.53.



The image shows a 'Kibana Filter Editor' window. At the top left is the title 'EDIT FILTER' and at the top right is a link 'Edit as Query DSL'. Below this, there are two dropdown menus: 'Field' with the value 'attack_name.keyword' and 'Operator' with the value 'is one of'. Underneath is a 'Values' section containing three tags: 'APPLICATION-ATTACK-XSS', 'APPLICATION-ATTACK-SQLI', and 'APPLICATION-ATTACK-LFI'. At the bottom left, there is a radio button labeled 'Create custom label?'. At the bottom right, there are two buttons: 'Cancel' and 'Save'.

Gambar 4. 53 Filter Visualisasi Tabel pada *Kibana*

Gambar 4.53 di atas memperlihatkan filter yang digunakan pada visualisasi tabel. Filter yang digunakan sama dengan filter pada *pie chart* karena data yang ingin ditampilkan adalah ketiga jenis serangan yang akan diuji cobakan. Setelah memberikan filter di atas, simpan dengan menekan *Save and Return*. Dengan demikian seluruh visualisasi tersebut telah akan ditampilkan pada *Dashboard Kibana*. Adapun tampilan *dari dashboard kibana* tertera pada gambar 4.54.



Gambar 4. 54 Tampilan *Dashboard Kibana*

Gambar 4.54 di atas menampilkan tampilan dari *dashboard* pada *kibana*. Langkah selanjutnya yaitu simpan *dashboard* tersebut dengan menekan *save* dan berikan *title* dan deskripsi dari *dashboard* tersebut.

4.6.6. Konfigurasi Elastalert

Konfigurasi terakhir yang dilakukan pada *ELK Stack* server yaitu konfigurasi *Elastalert*. *Elastalert* berfungsi untuk memberikan notifikasi dari serangan yang diterima web server melalui *slack* dalam waktu yang mendekati *realtime*. Untuk melakukan konfigurasi pada *elastalert*, langkah pertama yang dilakukan yaitu konfigurasi *file* pada *direktori elastalert/config.yaml*. Adapun konfigurasi yang dilakukan pada *elastalert* tertera pada gambar 4.55.

```

GNU nano 4.8                                elastalert/config.yaml
# This is the folder that contains the rule yaml files
# Any .yaml file will be loaded as a rule
rules_folder: example_rules

# How often ElastAlert will query Elasticsearch
# The unit can be anything from weeks to seconds
run_every:
  seconds: 5

# ElastAlert will buffer results from the most recent
# period of time, in case some log sources are not in real time
buffer_time:
  minutes: 15

# The Elasticsearch hostname for metadata writeback
# Note that every rule can have its own Elasticsearch host
es_host: localhost

# The Elasticsearch port
es_port: 9200

# The AWS region to use. Set this when using AWS-managed elasticsearch
#aws_region: us-east-1

# The AWS profile to use. Use this if you are using an aws-cli profile.
# See http://docs.aws.amazon.com/cli/latest/userguide/cli-chap-getting-started.html
# for details
#profile: test

# Optional URL prefix for Elasticsearch
#es_url_prefix: elasticsearch

# Connect with TLS to Elasticsearch

```

Read 115 lines

Get Help Write Out Where Is Cut Text Justify Cur Pos M-U Undo
Exit Read File Replace Paste Text To Spell Go To Line M-E Redo

Gambar 4. 55 Konfigurasi *Elastalert*

Gambar 4.55 di atas memperlihatkan konfigurasi yang dilakukan pada *elastalert*. Konfigurasi di atas bertujuan untuk menetapkan server yang memiliki *elasticsearch* dan menentukan jeda waktu notifikasi dikirimkan ke Slack. Langkah selanjutnya lakukan konfigurasi pada *file* *elastalert/example_rules/example_frequency.yaml* sebagaimana yang tertera pada gambar 4.56.


```

name: slack-demo
type: frequency
index: logstash-waf-*
num_events: 1
timeframe:
  hours: 1
filter:
- terms:
  attack_name.keyword: ["APPLICATION-ATTACK-SQLI", "APPLICATION-ATTACK-XSS", "APPLICATION-ATTACK-LFI"]
alert:
- "slack"
slack:
slack_webhook_url: "https://hooks.slack.com/services/T01RS873K46/B01S56RDH6Y/dStc9MsRqJvUAPbqjN1yf1c
"elastalert/example_rules/example_frequency.yaml" 20L, 347C 1,1 All

```

Gambar 4. 56 Konfigurasi *Rules Elastalert*

Gambar 4.56 memperlihatkan konfigurasi *rules elastalert* agar memberikan notifikasi pada *channel slack*. Langkah terakhir yang dilakukan yaitu dengan mengaktifkan *elastalert* dengan perintah :

```
“python3 -m elastalert.elastalert --verbose --rule example_frequency.yaml”
```

4.7. Pengujian Log Event Management Web Application Firewall

Setelah melakukan instalasi, konfigurasi dan pembuatan *rules logstash*, tahap selanjutnya yaitu pengujian log *event management web application firewall*. Pada tahap pengujian ini akan dilakukan 3 percobaan jenis serangan yaitu *SQL Injection*, *Local File Inclusion* dan *Cross Site Scripting*. Adapun bentuk serangan yang dilakukan untuk uji coba tertera pada tabel 4.1.

Tabel 4. 1 Uji Coba Serangan

No.	IP Attacker	Nama Serangan	Bentuk Serangan
1	192.168.1.8	<i>Local File Inclusion</i>	http://targeturl?page=/etc/passwd
2	192.168.1.6	<i>Cross Site Scripting</i>	http://targeturl?cari=<script>alert(1)</script>

Lanjutan Tabel 4.1 Uji Coba Serangan

3	192.168.1.7	<i>SQL Injection</i>	http://targeturl?id=999 UNION SELECT 1,2,3,4
---	-------------	----------------------	---

Tabel 4.1 di atas menampilkan pola serangan yang diuji cobakan pada aplikasi web. Ketiga uji coba serangan tersebut dilakukan pada aplikasi web yang memiliki celah keamanan. Pengujian *Local File Inclusion* (LFI) dilakukan dengan memasukkan perintah “/etc/passwd” pada kolom *url website*. Dengan perintah tersebut web server diminta menampilkan seluruh *user* yang ada pada server. Celah keamanan ini terjadi karena perintah *include* yang tidak difilter pada pemrograman web. Hasil pengujian dari serangan LFI dapat dilihat pada gambar 4.57.

Pengujian yang kedua yaitu *Cross Site Scripting* (XSS) dilakukan dengan memasukkan *script JavaScript* “<script>alert(1)</script>” pada kolom pencarian yang ada di *website*. Penggunaan *script* ini dilakukan untuk mengetahui apakah *website* memfilter *input* yang diberikan *user* terhadap web *application*. Celah keamanan ini dapat berakibat perubahan tampilan *website* hingga pencurian *session* pengguna. Hasil pengujian dari serangan XSS dapat dilihat pada gambar 4.58.

Pengujian yang terakhir yaitu *SQL Injection* yang dilakukan dengan memasukkan kode SQL “999 UNION SELECT 1,2,3,4” pada tab *url website*. Celah keamanan ini terjadi karena tidak ada filter terhadap *input* pengguna pada web *application*. Dengan demikian pengguna dapat memanfaatkan celah tersebut untuk mencari tahu informasi yang tersimpan pada *database* hingga melakukan *bypass* pada halaman *login*. Hasil pengujian dari serangan SQLI dapat dilihat pada gambar 4.59.

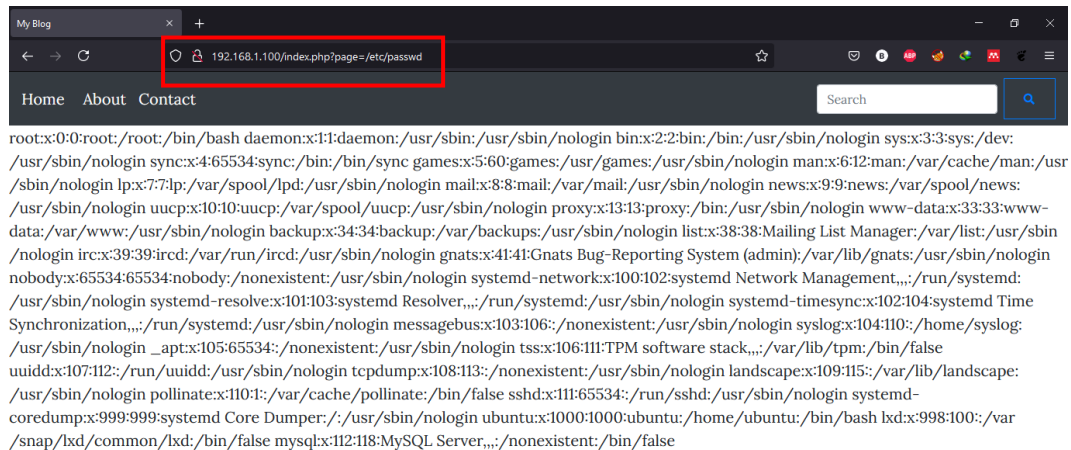
4.7.1. Uji Coba Serangan tanpa Log Event Management Web

Application Firewall

Berikut ini merupakan uji coba terhadap aplikasi web sebelum di terapkannya *log event management web application firewall*.

a. Local File Inclusion (LFI)

Uji coba yang pertama yaitu uji coba serangan *Local File Inclusion*. Adapun hasil uji coba serangan *local file inclusion* pada aplikasi web tertera pada gambar 4.57.

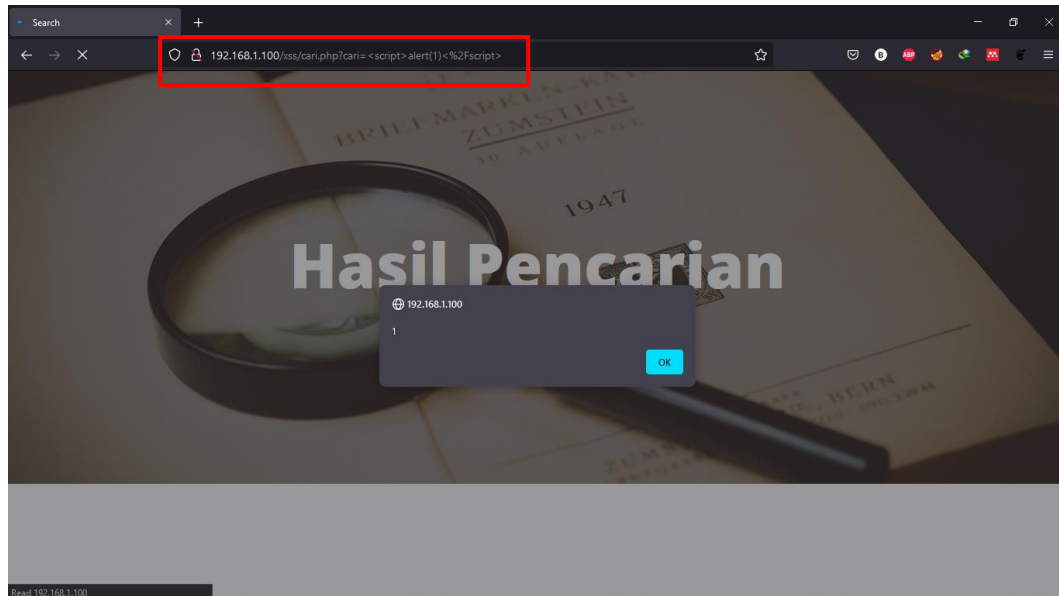


Gambar 4. 57 Uji Coba Serangan *Local File Inclusion* tanpa Log Event *Management Web Application Firewall*

Gambar 4.57 memperlihatkan hasil uji coba *local file inclusion* pada aplikasi berbasis web tanpa log *event management web application firewall*. Pada gambar di atas dapat dilihat bahwa web server merespons dengan menampilkan data *user* yang ada pada server web.

b. Cross Site Scripting (XSS)

Uji coba yang kedua dilakukan yaitu *cross site scripting* (XSS). Adapun hasil dari uji coba serangan *cross site scripting* tertera pada gambar 4.58.

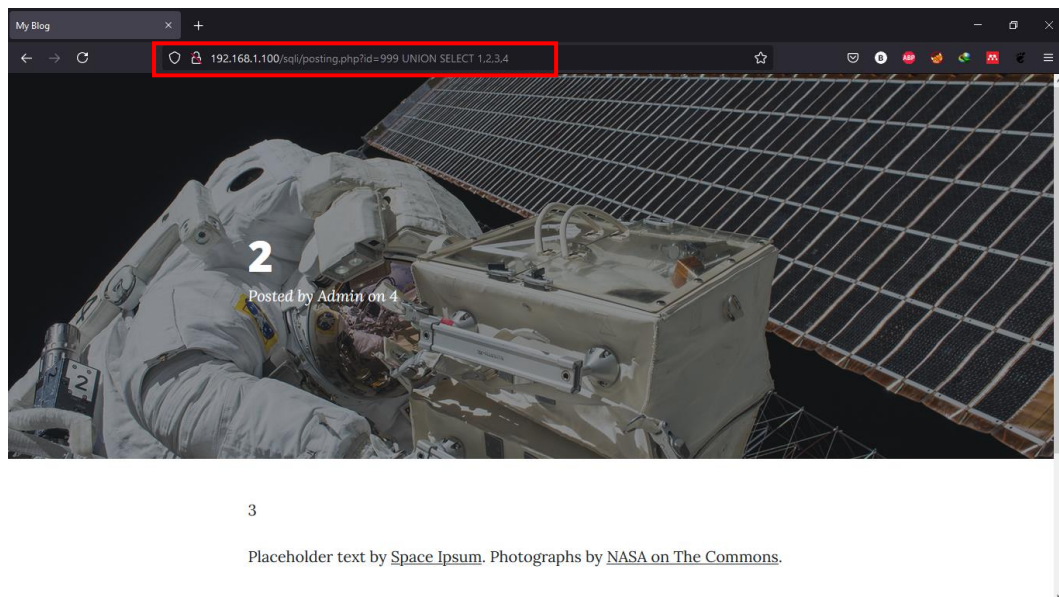


Gambar 4. 58 Uji Coba Serangan *Cross Site Scripting* tanpa Log Event *Management Web Application Firewall*

Gambar 4.58 memperlihatkan hasil uji coba cross site scripting pada aplikasi berbasis web tanpa *log event management web application firewall*. Pada gambar di atas dapat dilihat bahwa web server merespons dengan menampilkan *alert* pada web browser atas *request* yang diberikan.

c. *SQL Injection (SQLI)*

Uji coba ketiga yang dilakukan yaitu *SQL Injection*. Adapun hasil dari uji coba serangan *sql injection* tertera pada gambar 4.59.

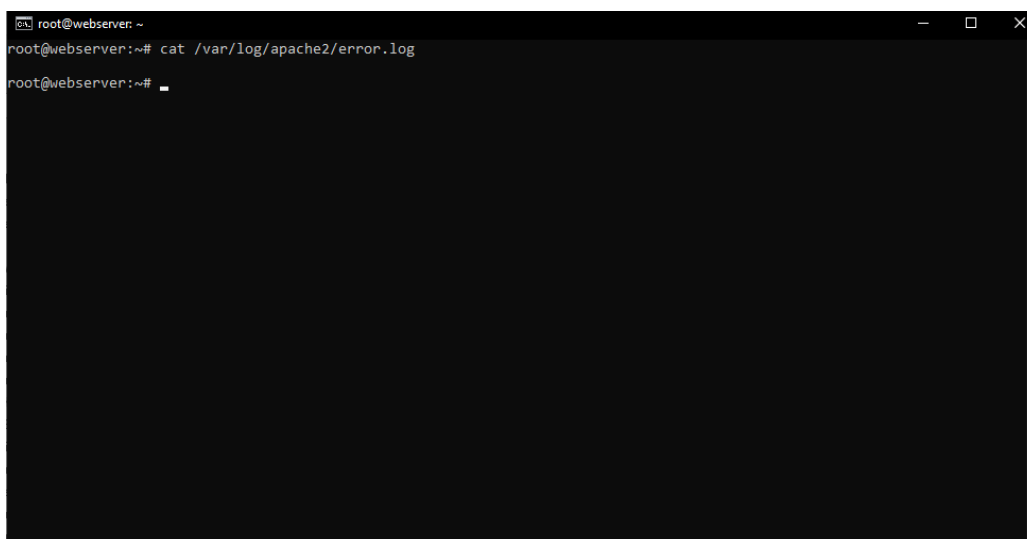


Gambar 4. 59 Uji Coba Serangan *SQL Injecton* tanpa Log Event *Management Web Application Firewall*

Gambar 4.59 memperlihatkan hasil uji coba serangan *SQL Injection* pada aplikasi berbasis web tanpa *log event management web application firewall*. Pada gambar di atas dapat dilihat bahwa terjadi perubahan isi dari halaman web akibat dari *request* yang dilakukan.

d. Pengecekan Log

Setelah dilakukan 3 uji coba serangan di atas web server tidak mendeteksi serangan tersebut. Adapun isi dari *file apache error.log* tertera pada gambar 4.60.



Gambar 4. 60 Log *Error Apache*

Gambar 4.60 memperlihatkan isi dari *file apache error.log*, di mana pada gambar di atas dapat dilihat bahwa *file* tersebut masih kosong yang menandakan bahwa web server tidak mendeteksi ketiga uji coba serangan yang dilakukan. Dengan tidak terdeteksinya uji coba serangan-serangan di atas, *attacker* dapat melakukan eskalasi serangan untuk mencuri informasi ataupun masuk ke dalam server.

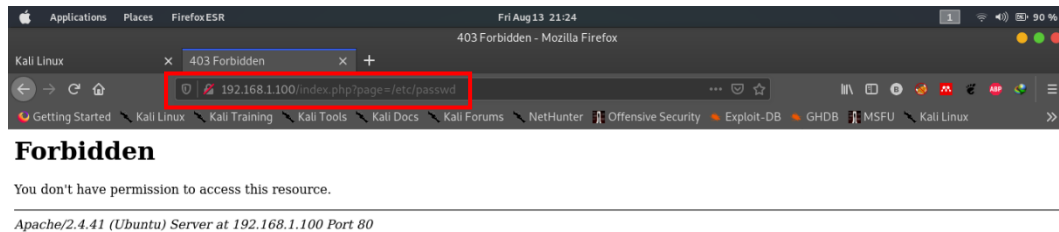
4.7.2. Uji Coba Serangan dengan Log *Event Management Web*

Application Firewall

Pada tahap ini akan dilakukan uji coba serangan pada web server setelah diterapkannya *log event management web application firewall*.

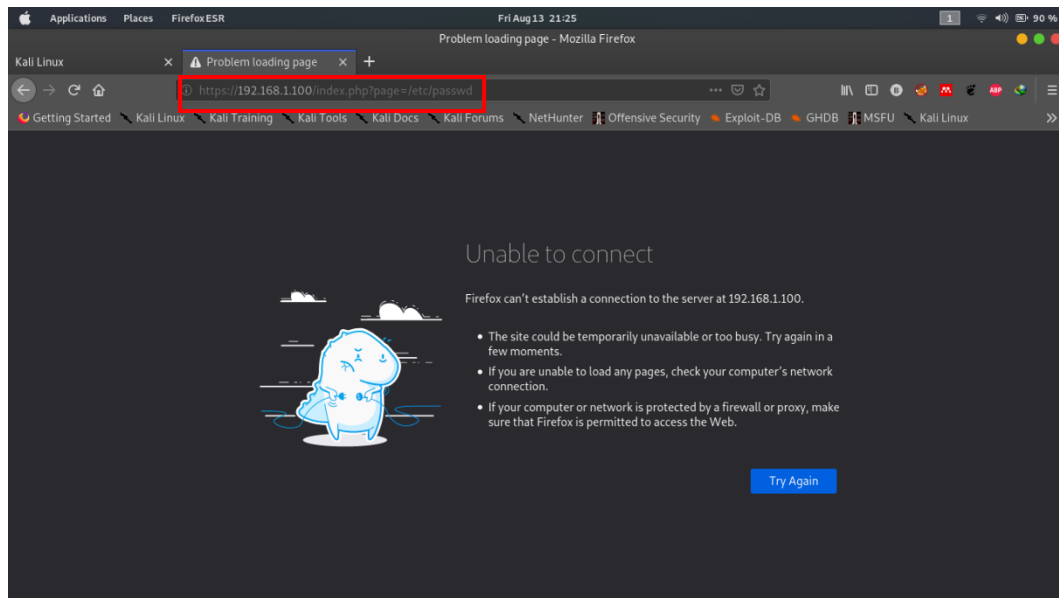
a. *Local File Inclusion (LFI)*

Uji coba serangan terhadap web server yang pertama yaitu *local file inclusion*. Adapun hasil dari uji coba serangan *local file inclusion* tertera pada gambar 4.61.



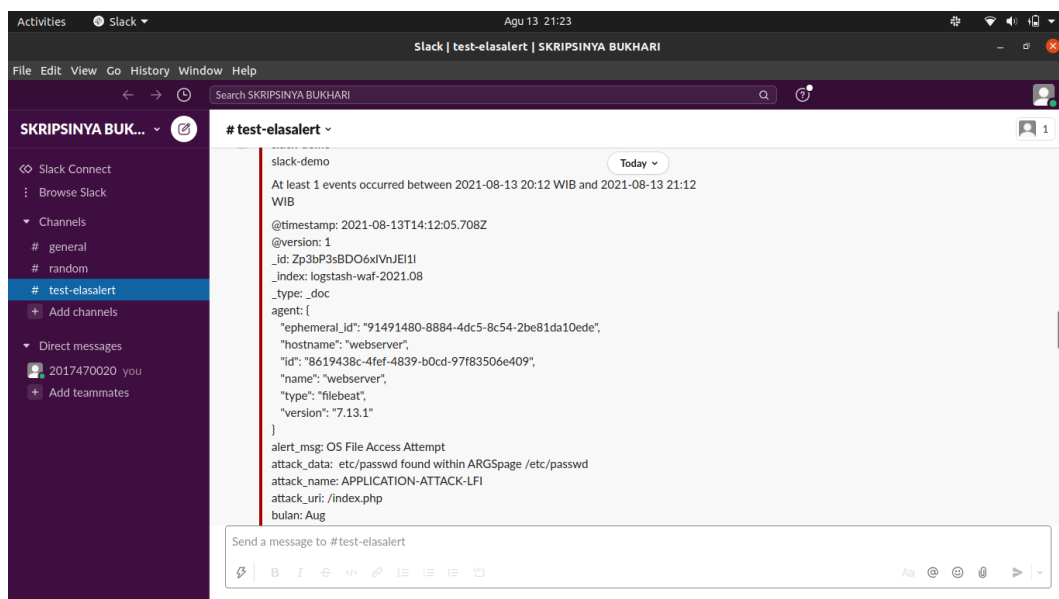
Gambar 4. 61 Uji Coba Serangan *Local File Inclusion* dengan Log Event *Management Web Application Firewall*

Gambar 4.61 memperlihatkan hasil uji coba serangan *local file inclusion* pada web server setelah diterapkannya *log event management web application firewall*. Pada gambar di atas dapat dilihat bahwa *request* yang dilakukan terdeteksi sebagai bentuk serangan, sehingga *modsecurity* memblokir *request* tersebut. Setelah melakukan uji coba serangan *Local File Inclusion*, *IP address attacker* akan diblokir sebagaimana yang tertera pada gambar 4.62.



Gambar 4. 62 Pemblokiran IP Address Attacker atas Serangan LFI

Gambar 4.62 di atas memperlihatkan pemblokiran IP *address attacker* setelah terindikasi adanya upaya peretasan menggunakan *Local File Inclusion* (LFI). Web server tidak memberikan respons terhadap *request* yang diminta oleh *attacker*, sehingga *attacker* tidak dapat mengakses halaman web untuk sementara waktu. Selain itu terdapat notifikasi dari serangan siber yang dilakukan sebagaimana yang tertera pada gambar 4.63.

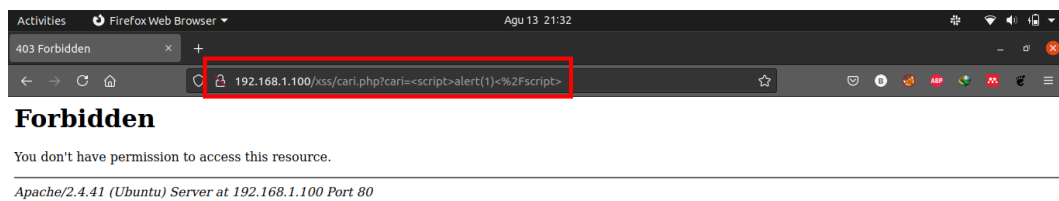


Gambar 4. 63 Notifikasi Serangan LFI pada Slack

Gambar 4.63 di atas memperlihatkan notifikasi serangan LFI yang dihadirkan melalui *Slack*. Pada notifikasi tersebut terdapat beberapa informasi seperti waktu serangan, *IP address attacker*, jenis serangan, pola serangan dan halaman yang diserang.

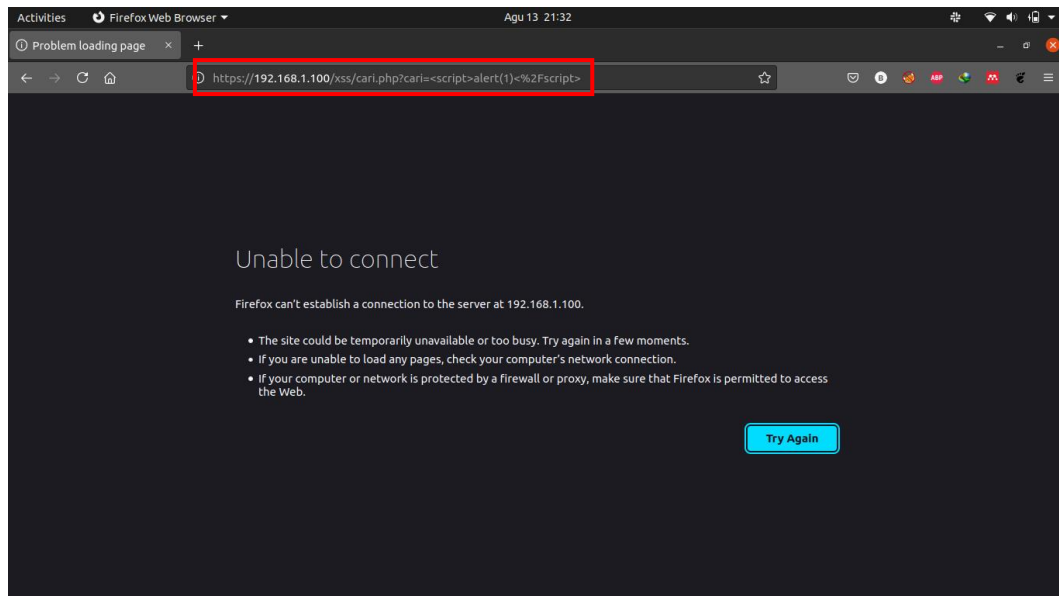
b. *Cross Site Scripting (XSS)*

Uji coba serangan terhadap web *application* selanjutnya yaitu *cross site scripting*. Adapun hasil uji coba serangan *cross site scripting* tersebut tertera pada gambar 4.64.



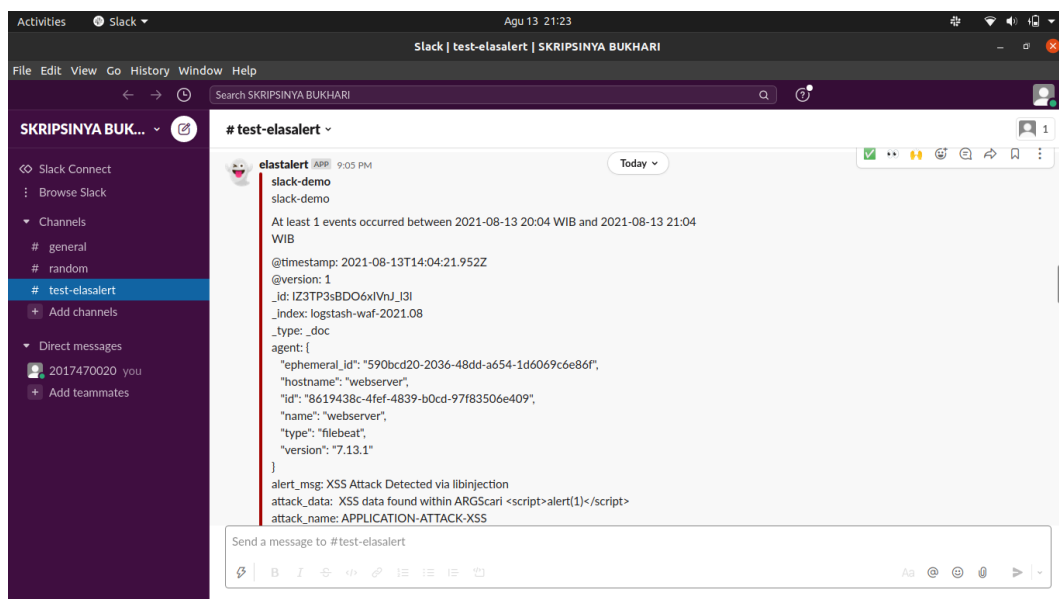
Gambar 4. 64 Uji Coba Serangan *Cross Site Scripting* dengan Log Event Management Web Application Firewall

Gambar 4.64 di atas memperlihatkan hasil uji coba serangan *cross site scripting* pada web *application* setelah diterapkannya log *event management web application firewall*. Pada uji coba serangan *cross site scripting* ini, *modsecurity* kembali memblokir *request* yang dilakukan. Setelah melakukan uji coba serangan *Cross Site Scripting*, *IP address attacker* akan diblokir sebagaimana yang tertera pada gambar 4.65.



Gambar 4. 65 Pemblokiran IP Address Attacker atas Serangan XSS

Gambar 4.65 di atas memperlihatkan pemblokiran IP *address attacker* setelah terindikasi adanya upaya peretasan menggunakan *Cross Site Scripting* (XSS). Web server tidak memberikan respons terhadap *request* yang diminta oleh *attacker*, sehingga *attacker* tidak dapat mengakses halaman web untuk sementara waktu. Selain itu terdapat notifikasi dari serangan siber yang dilakukan sebagaimana yang tertera pada gambar 4.66.

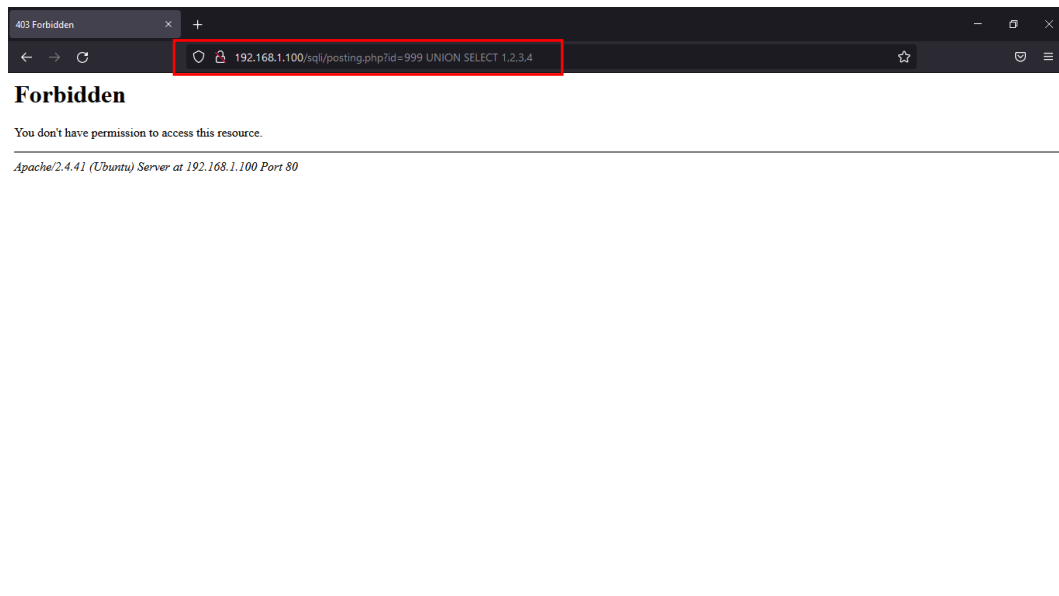


Gambar 4. 66 Notifikasi Serangan XSS pada Slack

Gambar 4.66 di atas memperlihatkan notifikasi serangan XSS yang dihadirkan melalui *Slack*. Pada notifikasi tersebut terdapat beberapa informasi seperti waktu serangan, *IP address attacker*, jenis serangan, pola serangan dan halaman yang diserang.

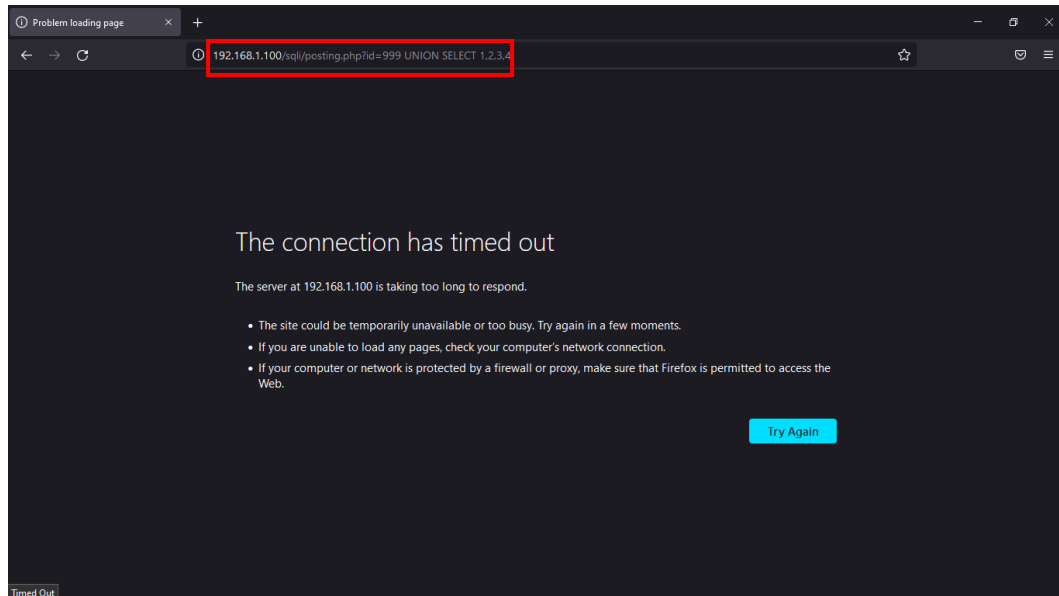
c. *SQL Injection (SQLI)*

Selanjutnya uji coba serangan yang terakhir yaitu *SQL Injection*. Adapun hasil dari uji coba serangan *SQL Injection* tertera pada gambar 4.63.



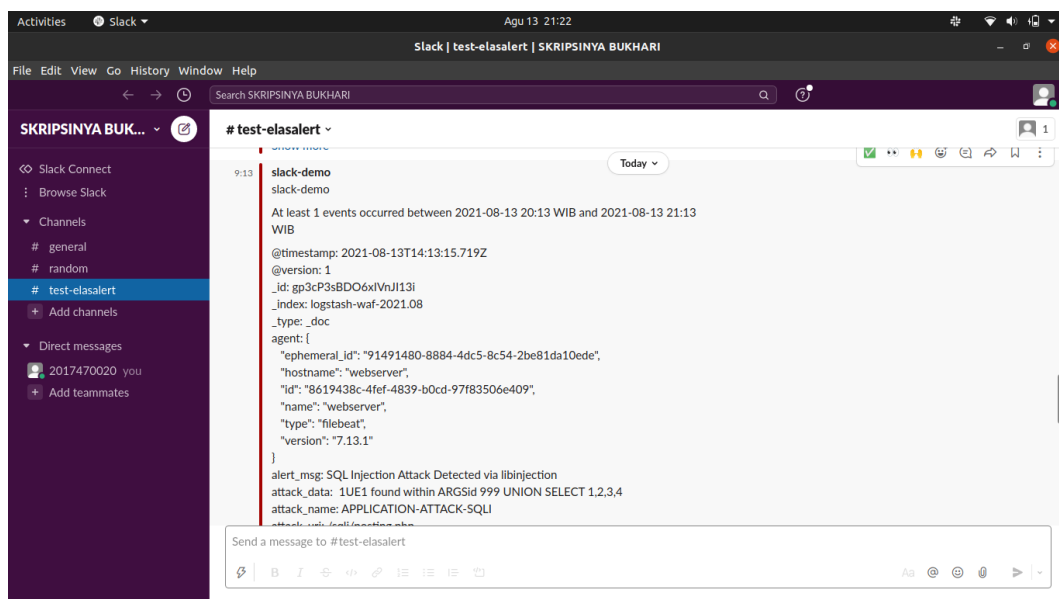
Gambar 4. 67 Uji Coba Serangan *SQL Injetcion* dengan *Log Event Management Web Application Firewall*

Gambar 4.67 memperlihatkan hasil dari uji coba serangan *SQL Injection* pada web *application* setelah diterapkannya *log event management web application firewall*. Pada gambar di atas dapat dilihat bahwa *modsecurity* memblokir serangan *SQL Injection* yang dilakukan pada web *application*. Setelah melakukan uji coba serangan *SQL Injection*, *IP address attacker* akan diblokir sebagaimana yang tertera pada gambar 4.68.



Gambar 4. 68 Pemblokiran IP Address Attacker atas Serangan SQLI

Gambar 4.68 di atas memperlihatkan pemblokiran IP address attacker setelah terindikasi adanya upaya peretasan menggunakan SQL Injection. Web server tidak memberikan respons terhadap request yang diminta oleh attacker, sehingga attacker tidak dapat mengakses halaman web untuk sementara waktu. Selain itu terdapat notifikasi dari serangan siber yang dilakukan sebagaimana yang tertera pada gambar 4.69.



Gambar 4. 69 Notifikasi Serangan SQLI pada Slack

Gambar 4.69 di atas memperlihatkan notifikasi serangan SQLI yang dihadirkan melalui *Slack*. Pada notifikasi tersebut terdapat beberapa informasi seperti waktu serangan, *IP address attacker*, jenis serangan, pola serangan dan halaman yang diserang.

d. Visualisasi Log

Selain memblokir serangan, *modsecurity* juga mencatat serangan tersebut pada *apache error.log*. Adapun log yang dicatat *modsecurity* tertera pada gambar 4.70.

```

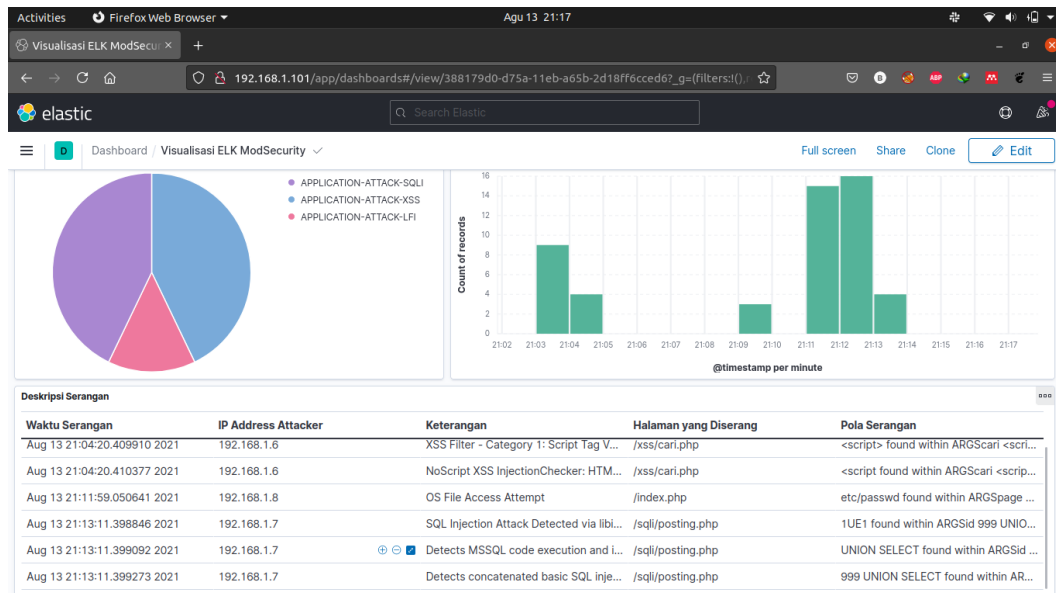
root@webserv: ~
root@elkserver: /home/ubuntu

+${ against variable 'REQUEST_HEADERS:Host' (Value: '192.168.1.100') [file "/etc/apache2/modsecurity.d/owasp-crs/rules/REQUEST-920-PROTOCOL-ENFORCEME
NT.conf"] [line "722"] [id "920350"] [rev "" ] [msg "Host header is a numeric IP address"] [data "192.168.1.100"] [severity "4"] [ver "OWASP_CRS/3.2.0"
] [maturity "0"] [accuracy "0"] [tag "application-multi"] [tag "language-multi"] [tag "platform-multi"] [tag "attack-protocol"] [tag "paranoia-level/1
"] [tag "OWASP_CRS"] [tag "OWASP_CRS/PROTOCOL_VIOLATION/IP_HOST"] [tag "WASCCTC/WASC-21"] [tag "OWASP_TOP_10/A7"] [tag "PCI/6.5.10"] [hostname "127.0.1
.1"] [uri "/assets/home/img/contact-bg.jpg"] [unique_id "16266552843.934951"] [ref "00,13v51,13"], referer: http://192.168.1.100/index.php?page=lf/c
ontact.php
[Mon Jul 19 10:32:08.271475 2021] [error] [pid 1858] [client 192.168.1.6:56498] ModSecurity: Warning: Matched 'Operator 'Rx' with parameter '^[\d.:]
+${ against variable 'REQUEST_HEADERS:Host' (Value: '192.168.1.100') [file "/etc/apache2/modsecurity.d/owasp-crs/rules/REQUEST-920-PROTOCOL-ENFORCEME
NT.conf"] [line "722"] [id "920350"] [rev "" ] [msg "Host header is a numeric IP address"] [data "192.168.1.100"] [severity "4"] [ver "OWASP_CRS/3.2.0"
] [maturity "0"] [accuracy "0"] [tag "application-multi"] [tag "language-multi"] [tag "platform-multi"] [tag "attack-protocol"] [tag "paranoia-level/1
"] [tag "OWASP_CRS"] [tag "OWASP_CRS/PROTOCOL_VIOLATION/IP_HOST"] [tag "WASCCTC/WASC-21"] [tag "OWASP_TOP_10/A7"] [tag "PCI/6.5.10"] [hostname "127.0.1
.1"] [uri "/assets/home/vendor/fontawesome-free/webfonts/fa-regular-400.woff2"] [unique_id "16266552883.222780"] [ref "00,13v85,13"], referer: http://
192.168.1.100/assets/home/vendor/fontawesome-free/css/all.min.css
[Mon Jul 19 10:32:08.278728 2021] [error] [pid 1850] [client 192.168.1.6:56500] ModSecurity: Warning: Matched 'Operator 'Rx' with parameter '^[\d.:]
+${ against variable 'REQUEST_HEADERS:Host' (Value: '192.168.1.100') [file "/etc/apache2/modsecurity.d/owasp-crs/rules/REQUEST-920-PROTOCOL-ENFORCEME
NT.conf"] [line "722"] [id "920350"] [rev "" ] [msg "Host header is a numeric IP address"] [data "192.168.1.100"] [severity "4"] [ver "OWASP_CRS/3.2.0"
] [maturity "0"] [accuracy "0"] [tag "application-multi"] [tag "language-multi"] [tag "platform-multi"] [tag "attack-protocol"] [tag "paranoia-level/1
"] [tag "OWASP_CRS"] [tag "OWASP_CRS/PROTOCOL_VIOLATION/IP_HOST"] [tag "WASCCTC/WASC-21"] [tag "OWASP_TOP_10/A7"] [tag "PCI/6.5.10"] [hostname "127.0.1
.1"] [uri "/favicon.ico"] [unique_id "16266552920.805626"] [ref "00,13v32,13"], referer: http://192.168.1.100/index.php?page=lf/contact.php
[Mon Jul 19 10:32:10.155110 2021] [error] [pid 1857] [client 192.168.1.6:56496] ModSecurity: Warning: Matched 'Operator 'Rx' with parameter '^[\d.:]
+${ against variable 'REQUEST_HEADERS:Host' (Value: '192.168.1.100') [file "/etc/apache2/modsecurity.d/owasp-crs/rules/REQUEST-920-PROTOCOL-ENFORCEME
NT.conf"] [line "722"] [id "920350"] [rev "" ] [msg "Host header is a numeric IP address"] [data "192.168.1.100"] [severity "4"] [ver "OWASP_CRS/3.2.0"
] [maturity "0"] [accuracy "0"] [tag "application-multi"] [tag "language-multi"] [tag "platform-multi"] [tag "attack-protocol"] [tag "paranoia-level/1
"] [tag "OWASP_CRS"] [tag "OWASP_CRS/PROTOCOL_VIOLATION/IP_HOST"] [tag "WASCCTC/WASC-21"] [tag "OWASP_TOP_10/A7"] [tag "PCI/6.5.10"] [hostname "127.0.1
.1"] [uri "/index.php"] [unique_id "16266553963.634907"] [ref "00,13v49,13"], referer: http://192.168.1.100/index.php?page=lf/contact.php
[Mon Jul 19 10:32:10.881690 2021] [error] [pid 1857] [client 192.168.1.6:56496] ModSecurity: Warning: Matched 'Operator 'Rx' with parameter '^[\d.:]
+${ against variable 'REQUEST_HEADERS:Host' (Value: '192.168.1.100') [file "/etc/apache2/modsecurity.d/owasp-crs/rules/REQUEST-920-PROTOCOL-ENFORCEME
NT.conf"] [line "722"] [id "920350"] [rev "" ] [msg "Host header is a numeric IP address"] [data "192.168.1.100"] [severity "4"] [ver "OWASP_CRS/3.2.0"
] [maturity "0"] [accuracy "0"] [tag "application-multi"] [tag "language-multi"] [tag "platform-multi"] [tag "attack-protocol"] [tag "paranoia-level/1

```

Gambar 4. 70 Log *ModSecurity* pada Web Server

Gambar 4.70 di atas memperlihatkan log yang dicatat oleh *modsecurity* pada web server. Log tersebut akan dikirim oleh *filebeat* menuju *ELK Stack* server untuk diurai, disimpan dan divisualisasikan. Seluruh log yang telah divisualisasikan dapat dilihat pada *dashboard* yang telah dibuat sebelumnya. Adapun tampilan *dashboard* pada kibana tertera pada gambar 4.71.



Gambar 4. 71 Tampilan Hasil Uji Coba Serangan pada *Dashboard Kibana*

Gambar 4.71 di atas memperlihatkan tampilan dari *dashboard kibana* setelah dilakukan uji coba serangan. *Kibana* melakukan visualisasi terhadap log yang sebelumnya telah diurai oleh *logstash* dan disimpan pada *elasticsearch*. Visualisasi dihadirkan dalam bentuk *pie chart*, *bar chart* dan tabel.

4.8. Pembahasan Hasil

Setelah dilakukan uji coba serangan siber *SQL Injection*, *Local File Inclusion* dan *Cross Site Scripting* untuk menguji penerapan *Log Event Management* dan *Web Application Firewall*, didapati hasil yang tertera pada tabel 4.2.

Tabel 4. 2 Hasil Pengujian *Log Event Management Web Application Firewall*

No.	Jenis Serangan	Hasil	Kesimpulan
1	<i>Local File Inclusion (LFI)</i>	Serangan LFI berhasil dideteksi (Gambar 4.61), diblokir (Gambar 4.62) dan log berhasil divisualisasikan (Gambar 4.71) serta terdapat notifikasi melalui <i>Slack</i> (Gambar 4.63)	Berhasil
2	<i>Cross Site Scripting (XSS)</i>	Serangan XSS berhasil dideteksi (Gambar 4.64), diblokir (Gambar 4.65) dan log berhasil divisualisasikan (Gambar 4.71) serta terdapat notifikasi melalui <i>Slack</i> (Gambar 4.66)	Berhasil

Lanjutan Tabel 4.2 Hasil Pegujian *Log Event Management Web Application Firewall*

3	SQL <i>Injection</i> (SQLI)	Serangan SQLI berhasil dideteksi (Gambar 4.67), diblokir (Gambar 4.68) dan log berhasil divisualisasikan (Gambar 4.71) serta terdapat notifikasi melalui <i>Slack</i> (Gambar 4.69)	Berhasil
---	-----------------------------------	---	----------

Tabel 4.2 di atas memperlihatkan hasil dari pengujian serangan yang dilakukan terhadap web server setelah diterapkannya *Log Event Management Web Application Firewall*. Ketiga jenis serangan yang diuji cobakan yaitu *Local File Inclusion* (LFI), *Cross Site Scripting* (XSS) dan *SQL Injection* (SQLI) berhasil diblokir oleh *ModSecurity* dan log divisualisasikan pada *ELK Stack Server*. Selain itu notifikasi yang berisikan detail serangan berhasil dihadirkan melalui *Slack*.

Dari hasil pengujian di atas penerapan *Log Event Management Web Application Firewall* terbukti dapat meningkatkan keamanan web server dari ketiga jenis serangan siber yang dilakukan. *ModSecurity* berhasil mendeteksi, memblokir dan mencatat serangan siber pada *log file*. Log tersebut kemudian berhasil dikirimkan pada *ELK Stack server* sehingga log yang sebelumnya menumpuk pada web server dapat diurai, disimpan dan divisualisasikan dalam bentuk *pie chart*, *bar chart* dan tabel. Notifikasi terhadap serangan siber pun berhasil dihadirkan melalui *Slack*, sehingga serangan siber yang terjadi pada web server dapat diketahui dalam jangka waktu yang singkat.

BAB V

KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan penelitian yang telah dipaparkan pada bab-bab sebelumnya, maka dapat ditarik kesimpulan sebagai berikut :

1. Berdasarkan hasil uji coba yang dilakukan, penerapan log *event management* web *application firewall* terbukti berhasil meningkatkan keamanan web *application* dari serangan siber. Dengan mengintegrasikan web *application firewall* dengan ELK *Stack* serangan siber akan diblokir dan dicatat pada log *file*. Selanjutnya log tersebut akan dikirim dan divisualisasikan pada ELK *Stack* Server.
2. *Modsecurity* berhasil mendeteksi dan memblokir serangan SQL *Injection*, *Local File Inclusion* dan *Cross Site Scripting*. Ketiga jenis serangan tersebut pun berhasil tercatat pada log *file*. Selanjutnya IP *Address* dari *attacker* yang tercatat pada log *modsecurity* akan diblokir oleh fail2ban dalam kurun waktu yang telah ditentukan, sehingga untuk sementara waktu IP tersebut tidak dapat mengakses web *application*.
3. Log *event management* terhadap log yang dihasilkan oleh web *application firewall modsecurity* berhasil dilakukan. Log yang dihasilkan *modsecurity* berhasil diurai, disimpan dan divisualisasikan dengan menggunakan ELK *Stack*. Data-data dari visualisasi log tersebut dapat digunakan sebagai dasar perbaikan celah keamanan yang ada pada web *application*.
4. Notifikasi serangan dihadirkan *elastalert* melalui *channel Slack* dalam waktu yang mendekati *real time*. Dengan demikian *system administrator* dapat memonitor serangan yang dialami web *application* melalui notifikasi tersebut.

5.2. Saran

Saran yang dapat diberikan untuk pengembangan lebih lanjut terhadap Log Event *Management Web Applicaiton Firewall* antara lain :

1. Melakukan uji coba serangan siber lain terhadap *Web Application Firewall ModSecurity*.
2. Membangun *Security Information and Event Management (SIEM)* dengan menghimpun, mengurai, menyimpan dan memvisualisasi berbagai jenis log yang dihasilkan dari berbagai macam perangkat seperti *router*, server, *firewall* dan *Intrusion Detection System (IDS)*.

DAFTAR PUSTAKA

- Admi, A., Hakim, A., & Maulana, N. (2020). Penerapan Elastic Stack sebagai Tools Alternatif Pemantauan Traffic Jaringan dan Host pada Instansi Pemerintah untuk Memperkuat Keamanan dan Ketahanan Siber Indonesia. *JUSTINDO (Jurnal Sistem & Teknologi Informasi Indonesia)*, 2017, 69–77.
- Al-Mahbashi, I. Y. M., Potdar, D. M. B., & Chauhan, M. P. (2017). Security Enhancement through Effective Log Analysis Using ELK. *Proceedings of the IEEE 2017 International Conference on Computing Methodologies and Communication*, (Iccmc), 566–570.
- Alwan, Z. S., & Younis, M. F. (2017). Detection and Prevention of SQL Injection Attack : A Survey. *International Journal of Computer Science and Mobile Computing*, 6(8), 5–17.
- Anshori, I. F. (2019). Implementasi Socket Tcp / Ip Untuk Mengirim Dan Memasukan File Text Kedalam Database. *Jurnal Responsif : Riset Sains Dan Informatika*, 1(1), 1–5.
- Arifin, M. N., Susilowati, E., & Sugiartowo. (2018). Desain Dan Implementasi Log Event Management Server Menggunakan Elasticsearch Logstash Kibana (Elk. *Seminar Nasional Sains Dan Teknologi*, 1–7. Retrieved from jurnal.umj.ac.id/index.php/semnastek
- Armanto. (2017). Implementasi Jaringan Tunnel Berbasis Eoip (Ethernet Over Ip) Dengan Mikrotik Router Rb 2011 Il-Rm Di Silampari Tv Lubuklinggau. *JURNAL ILMIAH BETRIK : Besemah Teknologi Informasi Dan Komputer*, 08(01), 42–52. Retrieved from <https://ejournal.lppmsttpagaralam.ac.id/index.php/betrik/article/view/65>
- Atmaja, A. P., & Yulianto, S. V. (2019). Pemanfaatan Elasticsearch untuk Temu Kembali Informasi Tugas Akhir. *Jurnal Nasional Teknologi Dan Sistem Informasi*, 4(3), 160–167. <https://doi.org/10.25077/teknosi.v4i3.2018.160-167>
- Babu, J. B., Prasad, S., & Prasad, G. S. (2019). Detecting and Analyzing the Malicious Linux Events using Filebeat and ELK Stack. *International Journal of Recent Technology and Engineering*, 8(6), 156–160.
- Bajer, M. (2017). Building an IoT Data Hub with Elasticsearch , Logstash and Kibana. *International Conference on Future Internet of Things and Cloud Workshops*. <https://doi.org/10.1109/FiCloudW.2017.101>
- Betarte, G., Gimenez, E., Martinez, R., & Pardo, A. (2019). Improving Web Application Firewalls through Anomaly Detection. *Proceedings - 17th IEEE International Conference on Machine Learning and Applications, ICMLA 2018*, 779–784. <https://doi.org/10.1109/ICMLA.2018.00124>
- Chandra, A. Y. (2019). Analisis Performansi Antara Apache & Nginx Web Server Dalam Menangani Client Request. *Jurnal Sistem Dan Informatika (JSI)*,

- 14(1), 48–56. <https://doi.org/10.30864/jsi.v14i1.248>
- Chhajed, S. (2015). *Learning ELK Stack*. Birmingham: Packt Publishing Ltd.
- Chinprutthiwong, P., Vardhan, R., Yang, G., & Gu, G. (2020). Security Study of Service Worker Cross-Site Scripting. *ACSAC '20: Annual Computer Security Applications Conference*, 643–654.
- Fauzi, R., & Desmulyati. (2020). IMPLEMENTASI NETWORK MONITORING SYSTEM MENGGUNAKAN NAGIOS DAN NAGVIS PADA PT . PELNI (PERSERO). *Journal of Information System , Informatics and Computing*, 4(1), 92–98.
- GÜL, E., & YILMAZ, E. N. (2019). LOG MANAGEMENT WITH OPEN SOURCE TOOLS. *International Symposium on Innovative Approaches in Scientific Studies*.
- Haerudin, D. I., Aksara, L. B., & Yamin, M. (2017). IMPLEMENTASI WIRELESS DISTRIBUTION SYSTEM (WDS) PADA HOTSPOT (STUDI KASUS : SMK NEGERI 1 KENDARI). *SemanTIK*, 3(2), 105–112.
- Hamilton, J., Gonzalez Berges, M., Tournier, J.-C., & Schofield, B. (2018). SCADA Statistics monitoring using the elastic stack (Elasticsearch, Logstash, Kibana). *16th Int. Conf. on Accelerator and Large Experimental Control Systems*, 451–455. <https://doi.org/10.18429/JACoW-ICALEPCS2017-TUPHA034>
- Hasan, A., & Meva, D. (2018). Web Application Safety by Penetration Testing. *Special Issue Based on Proceedings of 4TH International Conference on Cyber Security (ICCS) 2018*, 159–163.
- Hasrul, & Lawani, A. M. (2017). PENGEMBANGAN JARINGAN WIRELESS MENGGUNAKAN MIKROTIK ROUTER OS RB750 PADA PT . AMANAH FINANCE PALU. *Jurnal Elektronik Sistem Informasi Dan Komputer*, 3(1), 11–19.
- Isky. (2018). Pengertian Jaringan PAN, LAN, MAN, dan WAN. Retrieved March 27, 2021, from <https://www.isky.web.id/2018/09/pengertian-jaringan-pan-lan-man-dan-wan.html>
- Jankovic, D. Z. (2012). Key security measures for personal data protection in IT systems. *2012 20th Telecommunications Forum, TELFOR 2012 - Proceedings*, 79–82. <https://doi.org/10.1109/TELFOR.2012.6419152>
- Kementerian Pertahanan. *Peraturan Menteri Pertahanan Republik Indonesia Nomor 82 Tahun 2014 Tentang Pedoman Pertahanan Siber.* , (2014).
- Koprawi, M. (2020). Dampak dan Pencegahan Serangan File Inclusion: Perspektif Developer. *InfoTekJar : Jurnal Nasional Informatika Dan Teknologi Jaringan*, 5(1), 40–43. Retrieved from <https://doi.org/10.30743/infotekjar.v5i1.1997>
- Kurniawan, A. (2020). 3 Fungsi Router, Pahami Cara Kerja dan Perbedaannya dengan Modem. Retrieved April 4, 2021, from

<https://www.merdeka.com/jabar/3-fungsi-router-pahami-cara-kerja-dan-perbedaannya-dengan-modem-klm.html>

- Liu, G., Xu, J., Wang, C., & Zhang, J. (2018). A Performance Comparison of HTTP Servers in a 10G/40G Network. *ICBDC '18: Proceedings of the 2018 International Conference on Big Data and Computing*. Retrieved from https://dl.acm.org/doi/abs/10.1145/3220199.3220216?casa_token=W1iEfAFb0EEAAAAA:GS6kjdqaxOEZhAzGFX0RxQUTKvijZwnskIT_dWBu8WFJctB596Uf6W86g0-I7U0KGUuC544Cr9nl
- Mail. (2020). Pengertian Kabel Stp, Fungsi Kabel Stp dan Jenis Kabel Stp. Retrieved April 4, 2021, from <https://anaktik.com/pengertian-kabel-stp-fungsi-kabel-stp-dan-jenis-kabel-stp/>
- Muzawi, R., Efendi, Y., & Agustin, W. (2018). SATIN – Sains dan Teknologi Informasi Sistem Pengendalian Lampu Berbasis Web dan Mobile Rometdo Muzawi. *Sains Dan Teknologi Informasi*, 4(1), 29–35.
- Nugroho, K., & Kurniawan, A. Y. (2018). Uji Performansi Jaringan menggunakan Kabel UTP dan STP. *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika*, 5(1), 48. <https://doi.org/10.26760/elkomika.v5i1.48>
- Peniarsih. (2020). SISTEM JARINGAN INTERNET DATA UNTUK PENDISTRIBUSIAN VLAN. *Jurnal Mitra Manajemen*, 92–108. Retrieved from <https://journal.universitassuryadarma.ac.id/index.php/jmm/article/viewFile/547/513>
- Prasetyo, K. A., Idhom, M., & Wahanani, H. E. (2020). *SISTEM PENCEGAHAN SERANGAN BRUTEFORCE PADA MULTIPLE SERVER DENGAN MENGGUNAKAN FAIL2BAN*. 1(3), 789–796.
- Prasetyo, K. T. (n.d.). What is LAN (Local Area Network). Retrieved March 27, 2021, from <https://students.warsidi.com/2018/07/what-is-lan-local-area-network.html>
- Rajagukguk, P., Hardani, S., & Haryono, B. (2020). TINJAUAN PELAKSANAAN ADMINISTRASI PERSEDIAAN BARANG PADA PT MAXINDO MITRA SOLUSI JAKARTA. *Jurnal Akrab Juara*, 5(1), 55.
- Riska, & Alamsyah, H. (2021). Penerapan Sistem Keamanan WEB Menggunakan Metode WEB Application Firewall. *Jurnal Amplifier*, 11(1).
- Riska, P., Sugiartawan, P., & Wiratama, I. (2018). Sistem Keamanan Jaringan Komputer dan Data Dengan Menggunakan Metode Port Knocking. *Jurnal Sistem Informasi Dan Komputer Terapan Indonesia*, 1(2), 53–64.
- Rizal, R. (2019). Implementasi Jaringan Local Area Network (LAN) dengan Menggunakan Router Mikrotik pada SMA Kosgoro. *Jurnal Teknik Informatika STIMIK Antar Bangsa*, V(2), 103–107.
- Robinson, Akbar, M., & Ridha, M. A. F. (2018). Injection and Cross Site

- Scripting Prevention Using OWASP Web Application Firewall. *INTERNATIONAL JOURNAL ON INFORMATICS VISUALIZATION*, 2, 286–292.
- Sahren. (2021). Implementasi teknologi firewall sebagai keamanan server dari syn flood attack. *JURTEKSI (Jurnal Teknologi Dan Sistem Informasi)*, VII(2), 159–164.
- Sholihah, W., Pripambudi, S., & Mardiyono, A. (2020). Log Event Management Server Menggunakan Elastic Search Logstash Kibana (ELK Stack). *JTIM: Jurnal Teknologi Informasi Dan Multimedia*, 2(1), 12–20. <https://doi.org/10.35746/jtim.v2i1.79>
- Simanullang, A., Napitupulu, J., Jamaluddin, & Purba, M. J. (2018). SIMULASI PEMANFAATAN IPCOP SEBAGAI PC ROUTER DALAM JARINGAN LOCAL (LAN) DI LABORATORIUM FE-UMI. *Jurnal Manajemen Informatika & Komputerisasi Akuntansi*, 2(1), 22–29.
- Sora. (2015). Pengertian Kabel UTP Dan Fungsinya Secara Lengkap. Retrieved April 4, 2021, from <http://www.pengertianku.net/2015/01/pengertian-kabel-utp-dan-fungsinya-secara-lengkap.html>
- Tampubolon, K. E. A. (2019). Perbedaan Cyber Attack, Cyber Crime dan Cyber Warframe. *Juris-Diction*, 2.
- Teckchandani, A. (2018). Slack: A Unified Communications Platform Improve Team Collaboration, Available at [https:// Slack.com/](https://Slack.com/). *Academy of Management Learning & Education*, 17(2), 226–228.
- Widodo, D. A., Mushansyah, A., & Ambarsari, N. (2019). IMPLEMENTASI SISTEM PICTURE ARCHIVING AND COMMUNICATION SYSTEM PADA SISTEM OPERASI UBUNTU. *EProceedings of Engineering*, 11(1), 1–14. Retrieved from <https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/8991>