MONITORING AKTIVITAS *USER* PADA SYSTEM DENGAN MENGGUNAKAN EFK (ELASTICSEARCH, FLUENTD, KIBANA) *STACK*



Disusun Oleh:

N a m a : Danur Lintang

NIM : 15523075

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2020

HALAMAN PENGESAHAN DOSEN PEMBIMBING

MONITORING AKTIVITAS *USER* PADA SYSTEM DENGAN MENGGUNAKAN EFK (ELASTICSEARCH, FLUENTD, KIBANA) *STACK*

TUGAS AKHIR

Disusun Oleh:

N a m a : Danur Lintang
NIM : 15523075

Yogyakarta, 1 Januari 2020

Pembimbing,

(Fieyata Yudha S.Kom)

ISLAM
UNIVERSITAS
INDONESIA

HALAMAN PENGESAHAN DOSEN PENGUJI

MONITORING AKTIVITAS *USER* PADA SYSTEM DENGAN MENGGUNAKAN EFK (ELASTICSEARCH, FLUENTD, KIBANA) *STACK*

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Teknik Informatika di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 1 November 2020

Tim Penguji

Fietyata Yudha, S.Kom., M.Kom.

Anggota 1

Moh. Idris, S.Kom., M.Kom.

Anggota 2

Erika Ramadhani, S.T., M.Eng.

Mengetahui,
Ketua Program Studi Informatika – Program Sarjana
Fakultas Teknologi Industri
Universitas Islam Indonesia

(Dr. Raden Teduh Dirgahayu, S.T., M.Sc.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Danur Lintang

NIM : 15523075

Tugas akhir dengan judul:

MONITORING AKTIVITAS *USER* PADA SYSTEM DENGAN MENGGUNAKAN EFK (ELASTICSEARCH, FLUENTD, KIBANA) STACK

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila dikemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung resiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 1 November 2019



(Danur Lintang)

HALAMAN PERSEMBAHAN

Alhamdulillah, segala puji bagi Allah S.W.T berkat rahmat dan hidayah-Nya penulis dapat menyelesaikan Tugas Akhirnya diberikan kelancaran dalam proses penelitian beserta proses penulisan. Tugas Akhir ini penulis persembahkan kepada:

1. Ayah Agus Haryono dan Ibunda Sri Rejeki.
2. Bapak Fietyata Yudha S.Kom., M.Kom.
3. Kakakku Pranan Jiwo, Erika Afiana Putri, Rido Widi dan Aulianita Rizka fitri
4. Keponakanku Adrisakha Malik Prananda, Devandra Keanu Makarim, Khansania Githa Prananda
5. Teman – teman Teknik Informatika UII angkatan 2015.
6. Teman – teman OY! Indonesia

Beserta semua yang telah memberikan dorongan berupa moril maupun materiil dan menjadi alasan untuk penulis menyelesaikan Tugas Akhir ini.

HALAMAN MOTO

Pelaut handal tidak lahir di laut yang tenang

Work hard ! Everybody is Smart

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh

Puji syukur senantiasa dipanjatkan kehadirat Allah SWT, yang telah memberikan rahmat dan hidayah-Nya sehinga penulis dapat menyelesaikan Tugas Akhir dengan judul **"*MONITORING* AKTIVITAS *USER* PADA SISTEM DENGAN MENGGUNAKAN EFK (ELASTICSEARCH,FLUENTD,KIBANA)*STACK*"**

Dalam rangka menyelesaikan pendidikan Strata Satu (S1) dan memperoleh gelar Sarjana Komputer pada Program Studi Informatika, Jurusan Informatika, Fakultas Tekhnologi Industri, Universitas Islam Indonesia. Tak lupa shalawat serta salam kami haturkan kepada junjungan kita Nabi Muhammad SAW, yang telah membawa kita dari zaman jahiliyah menuju zaman terang benderang.

Kesuksesan dalam menyelesaikan penelitian dan penulisan Tugas Akhir ini tidak lepas dari banyak pihak yang telah memberikan dukungan, bimbingan, saran, serta bantuan baik itu berupa moril maupun materiil dalam menyelesaikan penelitian dan penulisan laporan ini. Penghargaan dan terima kasih dengan tulus penulis berikan kepada:

1. Allah SWT.
2. Kedua orang tua penulis, Agus Haryono dan Sri Rejeki yang selalu memberikan semangat, serta dorongan untuk menyelesaikan Tugas Akhir ini.
3. Bapak Fathul Wahid, S.T., M.Sc., Ph.D. sebagai Rektor Universitas Islam Indonesia.
4. Bapak Hari Purnomo, Prof., Dr., Ir., M.T sebagai Dekan Fakultas Teknologi Industri Universitas Islam Indonesia.
5. Bapak Hendrik, S.T., M.Eng. sebagai Ketua Jurusan Teknik Informatika Fakultas Teknologi Industri Universitas Islam Indonesia.
6. Bapak Dr. Raden Teduh Dirgahayu, S.T., M.Sc. sebagai Ketua Program Studi S1 Teknik Informatika Fakultas Teknologi Industri Universitas Islam Indonesia.

7. Bapak Fietyata Yudha S. Kom., M. Kom. sebagai dosen pembimbing Tugas Akhir yang telah memberikan ilmu, waktu, dan bimbingan.
8. Rido Widi selaku kakak dan mentor yang telah memberikan pengetahuan dibidang infrastruktur serta waktu yang diberikan.
9. Kakak saya Pranan Jiwo, Erika Afiana Putri dan Aulianita Rizka Fitri yang selalu setia mendengarkan keluh kesah dalam menyelesaikan Tugas Akhir ini.
10. Sahabat – sahabatku, Agung, Alvin, Ari, Asdita, Dandy, Bima, Madi, Handika, Hanif, Harry, Hilman L, Ivan, Alim, Ryan, Savana, Evan, dan Tavif yang telah memberikan semangat baik itu secara langsung maupun tidak langsung.
11. Kawan senasib seperjuangan, Aditama Faqih, Kinanti laras, Cahyani Tri Windari yang memberi semangat dari proses awal hingga akhir Tugas Akhir ini.
12. Teman, sahabat yang selalu mensupport secara langsung, Fatimah Umi Mardiyah.
13. Keluarga besar Teknik Informatika UII angkatan 2015, yang telah memberikan pembelajaran semasa kuliah
14. Semua pihak yang tidak dapat penulis sebutkan satu – persatu yang telah membantu dalam proses pengerjaan Tugas Akhir ini hingga selesai.

Semoga segala bentuk bantuan yang telah diberikan kepada penulis mendapat ganjaran berupa amal kebaikan di hadapan Allah SWT. Penulis menyadari bahwa masih terdapat kekurangan serta kesalahan dikarenakan keterbatasan yang penulis miliki.

Semoga laporan Tugas Akhir ini bermanfaat bagi pembaca dan dapat dikembangkan dikemudian hari.

Wassalamu'alaikum Warahmatullahi Wabarakatuh.

Yogyakarta, 1 November 2019

(Danur Lintang)

SARI

Log adalah catatan tentang peristiwa yang terjadi dalam *system* dan jaringan, log berisi tentang detail informasi yang berkaitan dengan peristiwa tertentu yang terjadi pada *system* atau jaringan. Banyak sekali log yang di tangkap system yang dapat memberitahu kita terkait hal- hal yang dapat membahayakan maupun merusak kinerja dari *system* tersebut. Perubahan yang terjadi di sytem bisa menjadi suatu hal yang harus kita awasi atau pantau agar tidak mengganggu proses bisnis yang dijalankan. Salah satu solusi untuk mengontrol seluruh log yang sangat banyak adalah dengan *log management. Log management* ini berangkat dari masalah banyaknya volume log yang dihasilkan dan dikirim oleh mesin. *Log management* pada umumnya mencangkup pengumpulan log agregasi, log yang terpusat, penyimpanan log jangka panjang, log pencarian dan pelaporan log.

Pengumpulan log terpusat (*centralized logging)* merupakan salah satu tujuan dari log *management*. Pengumpulan log secara terpusat ini berfungsi untuk memudahkan seseorang untuk melakukan *debuging* ketika suatu sistem terjadi malfungsi atau terjadi sebuah insiden. Selain itu membuat log secara terpusat juga sangat penting bagi strategi keamanan yang kuat. Dengan menempatkan seluruh log pada satu tempat yang terpusat, ini dapat membantu menyederhanakan seluruh analisis dari log dan dapat memberikan gambaran yang lebih terperinci tentang situasi saat ini. Penggunaan EFK (Elastichsearch, Fluentd dan Kibana) yang juga digunkan untuk membuat tampilan dan log – log yang terjadi di dalam suatu sistem kita dapat ditelusuri dengan mudah.

Tujuan dari pengimplementasian *tools* ini adalah untuk tersedianya data data yang dapat dipantau dengan mudah serta dapat dipahami dengan jelas. Pengujian dengan cara melakukan aktivitas di *local computer* kita lalu dipantau dengan mengambil *log-log* yang ada lalu di implementasikan dan divisualisasikan menjadi bentuk grafik yang tersedia di Kibana.

Kata kunci: log, monitoring, fluentd,kibana,elasticsearch.

GLOSARIUM

Glosarium memuat daftar kata tertentu yang digunakan dalam laporan dan membutuhkan penjelasan, misalnya kata serapan yang belum lazim digunakan. Contoh penulisannya seperti di bawah ini:

Elasticsearch	adalah salah satu database yang masuk ke dunia NoSQL dengan fokus di *search engine database*
Kibana	sebuah *frontend* visualisasi data yang kuat dan *dashboard* untuk Elasticsearch
Fluentd	adalah pengumpul data *open-source*, untuk mengumpulkan, mengubah, dan mengirimkan data log ke *backend* Elasticsearch

DAFTAR ISI

HALAMAN JUDUL i
HALAMAN PENGESAHAN DOSEN PEMBIMBING ii
HALAMAN PENGESAHAN DOSEN PENGUJI iv
HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR v
MONITORING AKTIVITAS USER PADA *SYSTEM* DENGAN MENGGUNAKAN EFK (ELASTICSEARCH,FLUENTD,KIBANA) STACK**Error! Bookmark**
HALAMAN PERSEMBAHAN vi
HALAMAN MOTO vii
KATA PENGANTAR viii
SARI x
GLOSARIUM xi
DAFTAR ISI xii
DAFTAR GAMBAR xiii
BAB I PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Batasan Masalah 2
1.4 Tujuan Penelitian 3
1.5 Manfaat Penelitian 3
1.6 Metode Penelitian 3
1.6.1 Metode Pengumpulan Data 3
1.6.2 Metode Pengembangan Sistem 4
Metode pengembangan sistem yang akan dilakukan berdasarkan data – data yang telah diperoleh dan akan di pergunakan, yaitu: 4
1.7 Sistematika Penulisan 5
BAB II LANDASAN TEORI 6
2.1 Logging 6
2.2 Monitoring 7
2.3 Elasticsearch 8
2.4 Fluentd 9
2.5. Kibana 9
2.6 Auditbeat 10
BAB III METODOLOGI PENELITIAN 15
BAB IV Hasil dan Pembahasan 23
4.2 Pembuatan dan Provisioning server 23
4.1.1 Pembuatan server 23
4.1.2 Provisioning Server 29
4.2 Hasil 42
BAB V Kesimpulan dan Saran 45
a. Kesimpulan 45
b. Saran 45
1. DAFTAR PUSTAKA 46
LAMPIRAN 48

DAFTAR GAMBAR

Gambar 3.1 Skema Alir Penelitian……..14
Gambar 3.2 Skema Perancangan arsitektur sistem Authentifikasi. 16
Gambar 3.3 Skema Perancangan arsitektur sistem File integrity. 17
Gambar 4.1 Dashboard Awal AWS console 20
Gambar 4.2 AWS Instances 20
Gambar 4.3 Amazon Machine Image 21
Gambar 4.4 Choose an instance type.....21
Gambar 4.5 Configure Instance Details.....22
Gambar 4.6Add Storage.....23
Gambar 4.7 Add tags.....23
Gambar 4.8 Hasil tags.....24
Gambar 4.9 Configure Security Group..... 24
Gambar 4.10 Playbook EFK stack.....25
Gambar 4.11 Playbook Elasticsearch.....27
Gambar 4.12 Playbook Fluentd.....27
Gambar 4.13 Playbook Kibana.....28
Gambar 4.14 Playbook Fluentd..... 29
Gambar 4.15 Role Fluentd.....30
Gambar 4.16 Fluentd task1..... 30
Gambar 4.17 Fluentd task2.....31
Gambar 4.18 Fluentd task3..... 32
Gambar 4.19 Role Elasticsearch.....32
Gambar 4.20 task Elasticsearch 1 33
Gambar 4.21 task Elasticsearch 2 34
Gambar 4.22 task Elasticsearch 3 35
Gambar 4.23 Role kibana..... 36

Gambar 4.24 task kibana 1..... 36
Gambar 4.25 task kibana 2 37
Gambar 4.26 Hasil alerting kibana 38
Gambar 4.27 Hasil alerting auditd 39

BAB I
PENDAHULUAN

1.1 Latar Belakang

Ilmu pengetahuan dan teknologi semakin berkembang menjadi latar belakang seseorang untuk memiliki peluang yang lebih banyak untuk mendalami dan menciptakan hal-hal yang baru yang berguna untuk mempermudah tugas manusia di masa kini maupun masa yang akan datang. Salah satunya adalah teknologi yang memanfaatkan komputer sebagai salah satu penunjang perkerjaan mereka. Pemanfaatan komputer sudah banyak sekali kita dijumpai mulai dari pemanfaatan sederhana seperti pengolahan data maupun untuk hal yang lebih kompleks seperti untuk pembangunan sebuah *system*. Banyak perusahaan pada saat ini yang memanfaatkan komputer untuk pengembangan *system* yang mereka buat sebagai alat penunjang keberlangsungan perusahaan tersebut. Pengembangan system biasanya mereka implementasikan pada server-server yang dibuat sebagai tempat *system* di bentuk.

Untuk menunjang kinerja pada komputer server maka dibutuhkan system operasi khusus yang dibuat untuk server tersebut, ada beberapa contoh system operasi pada server seperti *windows, novel netware, mac os, linux*. Selain itu pada server-server tersebut juga terkadang dibuat untuk menjalankan tugas-tugas yang berberda. Dalam pengembangan sebuah system tentu kita perlu melakukan monitoring terhadap *system* yang kita buat untuk menjaga dan memelihara *system* agar tetap dalam kondisi yang prima. Dalam hal ini kita bias monitoring server kita dengan melihat metrics dan log yang dikirimkan server tersebut.

Log adalah catatan tentang peristiwa yang terjadi dalam system dan jaringan, log berisi tentang detail informasi yang berkaitan dengan peristiwa tertentu yang terjadi pada system atau jaringan. Banyak sekali log yang di tangkap system yang dapat memberitahu kita terkait hal- hal yang dapat membahayakan maupun merusak kinerja dari *system* tersebut. Perubahan yang terjadi di sytem bisa menjadi suatu hal yang harus kita awasi atau pantau agar tidak mengganggu proses bisnis yang dijalankan.

File Integrity Monitoring (FIM) merupakan salah satu solusi untuk memonitoring sebuah *system*. *File Integrity Monitoring* (FIM) adalah proses internal yang melakukan tindakan memvalidasi integritas sistem operasi dan file perangkat lunak aplikasi

menggunakan metode verifikasi antara status *file* saat ini dan *baseline* yang diketahui baik. Selain itu pemanfaatan *File Integrity Monitoring* (FIM) dapat juga untuk memproteksi IT infrastruktur dikarenakan dapat memantau perubahan yang terjadi dalam *system, database, network devices*, aplikasi, server maupun *cloud*. Kemudian bagaimana kita dapat memantau secara detail peristiwa-peristiwa tersebut? Pada tulisan ini, saya bermaksud untuk melakukan penelitian serta implementasi penggunakan EFK Stack (*elasticsearch, fluentd, dan kibana*) yang mendukung *File Integrity Monitoring* (FIM). Mengapa kita membutuhkan *FIM* ini? Jawabannya karena sudah banyak sekali contoh kasus yang *cyber* , salah satunya salah satu perusahaan besar "*Adobe"* menjadi korban atas tercurinya jutaan akun termasuk nama, password, dan bahkan nomor kartu kredit. Implementasi FIM bisa menjadi salah satu solusi yang tepat untuk melakukan pencegahan suatu serangan dari internet.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan maka rumusan masalah dalam penelitian ini adalah :

1. Bagaimana memonitor siapa saja yang mencoba akses terhadap *system* yang kita buat?
2. Bagaimana mendeteksi *sudo command* yang dieksekusi oleh *user*?
3. Bagaimana mendeteksi apa saja yang dilakukan user pada *system* kita?
4. Bagaimana *auditbeat* dan *elasalert* mendeteksi dan mengirimkan *alerting*?

1.3 Batasan Masalah

Untuk menjaga fokus penelitian dalam TA ini, beberapa batasan berikut diperhatikan:

1. Pada tugas akhir ini menggunakan *system* operasi *Linux.*
2. Dalam penelitian ini, implementasi dengan menggunakan EFK stack.
3. Dalam penelitian ini menggunakan *log shipper* yaitu Auditbeat.
4. Untuk *command* yang di monitor adalah *sudo command*
5. *Auditbeat* hanya memonitoring pada path /etc/

1.4 Tujuan Penelitian

Tujuan penelitian ini akan merancang *system monitoring* untuk server untuk *memonitoring* aktivitas *user* yang dilakukan pada server dengan memanfaatkan *elasticsearch, fluentd, dan kibana* (EFK).

1.5 Manfaat Penelitian

Adapun manfaat dari penelitian ini adalah sebagai berikut:

1. Memudahkan *system* administrator untuk memantau server dalam hal keamanan sehingga dapat melakukan pencegahan maupun penanganan terhadap celah pada server.
2. Memudahkan melakukan *troubleshooting* dengan melihat detail log yang ada di web *dashboard* kibana.
3. Memproteksi infrastruktur IT dengan menerapkan implementasi *File Integrity Monitoring* (FIM).

1.6 Metode Penelitian

1.6.1 Metode Pengumpulan Data

Metode Pengumpulan data yang akan dilakukan dalam penelitian ini yaitu:

a. Studi Literatur

Metode ini digunakan dengan tujuan untuk memperoleh informasi terkait penerapan EFK Stack (elasticsearch, fluentd, dan kibana) pada sebuah sistem untuk menerapkan *File Integrity Monitoring* (FIM).

b. Wawancara

Wawancara digunakan untuk mendapatkan informasi serta mengumpulkan data dari seorang infrastruktur *engineer* maupun *sysadmin* yang telah menjalani pekerjaannya untuk memonitoring *system*. Data tersebut mengenai bagaimana pengembangan terkait *File Integrity Monitoring* (FIM)

1.6.2 Metode Pengembangan Sistem

Metode pengembangan sistem yang akan dilakukan berdasarkan data – data yang telah diperoleh dan akan di pergunakan, yaitu:

a. Identifikasi Masalah

Tahapan ini dilakukan dengan melakukan wawancara serta melakukan kuisioner dengan beberapa engineer secara mendalam terkait system monitoring yang akan di buat. Dari tahapan ini diharapkan dapat mengidentifikasi masalah yang ada pada sisi engineer serta dapat meningkatkan produktivitas kerja.

b. Studi Pustaka

Tahapan ini dilakukan dengan melakukan studi pustaka yang lebih mendalam terkait topik- topik yang mengangkat tema *monitoring tools, elasticsearch, kibana dan fluentd.*

c. Perancangan Sistem

Tahap perancangan sistem ini akan dilakukan untuk membentuk merancang kebutuhan sistem sehingga mempermudah tahapan implementasi sistem. Perancangan sistem ini terdiri dari beberapa perancangan, yaitu:

1. Perancangan Elasticsearch
2. Perancangan Fluentd
3. Perancangan Kibana

d. Implementasi sistem

Tahapan Implementasi sistem merupakan tahap dimana penerapan File Integrity Monitoring (FIM) yang diimplementasikan pada EFK stack berdasarkan pada tahap perancangan serta melakukan integraso amyar ramcamham sistem tersebut.

e. Pengujian sistem

Setelah system selesai dibuat, maka akan langsung dilakukan pengujian dengan mengecek bagaimana performa dan jalannya system tersebut, apakah log sudah berhasil di kirim dan ditampilkan pada kibana dengan cara melakukan testing dengan melakukan perubahan- perubahan pada file yang ada pada sebuah sistem.

f. Evaluasi Sistem

Hasil dari pengujian system akan ditinjau kembali serta di evaluasi apakah sudah sesuai dengan rancangan yang dikehendaki.

1.7 Sistematika Penulisan

Dalam proses penulisan tugas akhir ini, sistematika penulisan dibagi menjadi 6 bab, yaitu:

BAB I Pendahuluan, berisi tentang latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian, dan sistematika penulisan.

BAB II Landasan Teori, berisi tentang landasan teori yang bertujuan sebagai sumber atau alat dalam memahami konsep dasar dalam implementasi *efk stack*

BAB III Metodologi Penelitian, Mendeskripsikan uraian metodologi penelitian yang dilakukan. Berisi subab-subab bahasan, yaitu: skema alir penelitian, skema perancangan EFK stack, dan skenario pengujian.

BAB IV Hasil dan Pembahasan, Bagian ini memuat pembahasan hasil aktifitas yang diperoleh selama penelitian. Pembahasan ini dapat berupa uraian, penjelasan, dan tahap-tahap proses analisa data.

BAB V Kesimpulan dan saran, Bagian ini berisi kesimpulan serta saran dari penelitian yang telah dilakukan.

BAB II
LANDASAN TEORI

2.1 Logging

Logging merupakan sebuah proses pencatatan informasi pada suatu sistem mengenai peristiwa yang terjadi pada sebuah sistem yang ada atau bisa dikatakan secara mudah yaitu sebagai rekaman aktivitas yang terjadi sebelumnya pada sistem. Log itu sendiri berarti dapat dikatakan sebagai catatatan peristiwa. Log ini akan disimpan oleh sistem pada Log *file* yang dapat berisi informasi tentang perubahan perangkat, *driver* perangkat, perubahan sistem, peristiwa, operasi, dan lainnya. Log pada sistem memiliki kegunaan yang sangat penting yang dibutuhkan ketika kita membuat sebuah sistem, mendeteksi dan memecahkan masalah yang ditemukan di komputer, sebagai media penyimpanan dan pemulihan data tentang perubahan pada item data. Dan juga dapat digunakan sebagai peringatan dari sistem yang dimana dapat digunakan untuk memprediksi masalah sistem.

Pada saat ini salah satu solusi untuk mengontrol seluruh log yang sangat banyak adalah dengan *log management. Log management* ini berangkat dari masalah banyaknya volume log yang dihasilkan dan dikirim oleh mesin. *Log management* pada umumnya mencangkup pengumpulan log agregasi, log yang terpusat, penyimpanan log jangka panjang, log Pencarian dan pelaporan log.

Pengumpulan log terpusat (*centralized logging)* merupakan salah satu tujuan dari log *management*. Pengumpulan log secara terpusat ini berfungsi untuk memudahkan seseorang untuk melakukan *debuging* ketika suatu sistem terjadi malfungsi atau terjadi sebuah insiden. Selain itu membuat log secara terpusat juga sangat penting bagi strategi keamanan yang kuat. Dengan menempatkan seluruh log pada satu tempat yang terpusat, ini dapat membantu menyederhanakan seluruh analisis dari log dan dapat memberikan gambaran yang lebih terperinci tentang situasi saat ini. Dengan terpusatnya semua log pada satu tempat, maka dapat memudahkan dalam membuat suatu notifikasi (*alert*) untuk administrator sistem

2.2 Monitoring

2.2.1 Pengertian

Monitoring adalah suatu kegiatan yang dilakukan untuk mengamati, menganalisa suatu kondisi maupun keadaan dan mengambil informasi yang penting berdasarkan data yang ada. Melalui kegiatan monitoring diperoleh informasi terkait sukses tidaknya suatu sistem yang kita buat, apakah sudah berjalan sesuai rencana awal dan apakah sebuah sistem sudah berjalan benar. Kegiatan monitoring dapat dilakukan sebelum kegiatan itu berlangsung, biasanya monitoring sebelum kegiatan berlangsung dilakukan untuk mengukur kondisi awal suatu sistem sebelum kita mengembangkan sistem. Monitoring pada saat kegiatan berlangsung, kegiatan ini dilakukan untuk mengetahui apakah sistem yang sudah diimplementasi berjalan dengan baik dan benar dan monitoring setelah kegiatan berlangsung yang memiliki tujuan sebagai evaluasi dan sebagai bahan masukan untuk kegiatan yang akan dilakukan selanjutnya. Tujuan dari monitoring yaitu:

1. Untuk mengetahui sejumlah informasi yang penting yang dapat digunakan sebagai acuan kegiatan atau tindakan yang akan dilakukan.
2. Memantau proses dan perkembangan pelaksanaan aktiftas program dengan mengacu pada indikator yang telah di tetapkan.
3. Kegiatan ini dapat sebagai antisipasi insiden maupun pencegahan dari sesuatu yang terjadi tidak semestinya.
4. Mendapatkan gambaran atas ketercapaian suatu proses.

2.2.2 Monitoring Tools

Monitoring tools atau alat monitoring adalah suatu alat yang digunakan untuk memantau aktivitas yang terjadi pada mesin. Monitoring tools digunakan untuk mengetahui status dari suatu sistem untuk mendapatkan informasi mengenai kegagalan sistem, masalah dan bagaimana untuk memperbaiki masalah tersebut maupun untuk lebih mengoptimalkan suatu sistem. Dalam hal ini monitoring yang digunakan adalah monitoring untuk mendukung pengumpulan log yang terpusat (*centrelized logging*).

Dalam project ini penulis menggunkan *tools* sebagai berikut:

- Amazon Web Services (AWS)
- Ansible

- Slack
- Elasalert
- Elasticsearch
- Fluentd
- Kibana
- Auditbeat

2.3 Landasan Teori

2.3..1 Elasticsearch

Elasticsearch merupakan salah satu *open source* yang dapat kita gunakan secara gratis yang memiliki tujuan atau fungsi sebagai mesin pencari (*search engine*) maupun mesin analisis text yang dapat diskalakan. Elasticsearch akan memudahkan kita untuk melakukan penyimpanan, pencarian serta melakukan analisis data dengan jumlah data yang besar secara cepat dan *real-time*. Elasticsearch juga menjadi salah satu database yang terjun ke dalam NoSQL, dengan fokus pada *search engine database*. Pada EFK stack, peran elasticseacrh ini sebagai database untuk menampung data data yang nantinya akan dikirimkan dari tiap-tiap mesin yang berupa log yang dituliskan oleh fluentd. Elasticsearch sendiri memiliki fitur yang dibawanya yaitu:

a. Fast time to value

Elasticsearch menawarkan API berbasis REST sederhana, antarmuka HTTP sederhana, dan menggunakan dokumen JSON bebas skema, sehingga mudah untuk memulai dan membangun aplikasi dengan cepat untuk berbagai kasus penggunaan.

b. High performance

Elasticsearch memiliki sifat terdistribusi sehingga untuk memproses volume data yang besar dapat diproses pencariannya dengan cepat sesuai *query* yang kita inginkan.

c. Real-time operation

Elasticsearch dapat digunakan untuk mendeteksi anomali maupun pemantauan aplikasi karena dapat menampilkan data secara *real-time* (langsung).

d. Multiple application deployment

Elasticsearch tersedia untuk berbagai bahasa termasuk Java, Python, PHP, JavaScript, Node.js, Ruby dan banyak lagi.

e. Complimantary tooling and plugin

Elasticsearch pada saat ini dapat diintegrasikan dengan beberapa platform populer yang dapat membantu memaksimalkan kinerja elasticsearch.

2.3..2 Fluentd

Fluentd adalah salah satu perangkat lunak yang dapat digunakan secara gratis yang berfungsi sebagai *data collector* atau memiliki fungsi sebagai alat pengumpul data yang dimana memungkinkan kita para pengguna dapat menyatukan pengumpulan dan konsumsi data untuk penggunaan serta pemahaman yang lebih baik. Data yang dimaksud salah satunya adalah log yang ada pada setiap mesin, tugas fluentd adalah mengumpulkan log log tersebut sebelum log tersebut didistribusikan ke tempat lain, pada EFK stack, data yang telah dikumpulkan oleh fluentd akan di kirimkan ke elasticsearch atau untuk pemahaman lebih mudah, fluentd yang akan melakukan penulisan pada elasticsearch indexs. Banyak kasus-kasus yang memaksa kita untuk membaca log dalam menyelesaikan masalah. Kadang banyaknya aplikasi yang tersebar di beberapa server sangat menyulitkan ketika ingin mengaksesnya dan dengan hadirnya fluentd sangat membantu pekerjaan tersebut. Fluentd akan menuliskan data data yang dikumpulan tersebut kedalam format json dengan cara melakukan parsing yang dilakukan untuk memilah data tersebut sesuai dengan apa yang kita ingin outputkan. Format JSON ini yang membuat fluentd menjadi fleksibel dan terstruktur.

2.3..3 Kibana

Kibana merupakan salah satu *open source* yang dapat kita gunakan secara gratis yang memiliki tujuan atau fungsi sebagai alat visualisasi data serta dapat digunakan untuk eksplorasi data seperti analisis log dan time-series, pemantauan aplikasi, dan kasus penggunaan intelijen operasional. Dalam hal menampilkan datanya, kibana menawarkan fitur yang sangat baik dan serta memudahkan kita dalam analisis data tersebut yaitu seperti grafik, diagram, histogram, diagram lingkaran. Kibana untuk menampilkan visualisasi data, dia berintegrasi dengan elasticsearch selaku tempat dimana data disimpan atau secara mudahnya,

kibana mengambil data yang ditulis fluentd di elasticsearch. Kibana memiliki memiliki kemudahan sebagai berikut:

1. Intuitive chart

Salah satu fitur yang ditawarkan kibana adalah dengan menyediakan bagan-bagan dan laporan intuitif yang dapat kita gunakan sebagai salah satu pendukung kita dalam mempermudah pencarian data.

2. Pre-build aggregations and filters

Menggunakan agregasi dan filter buatan Kibana, Anda dapat menjalankan berbagai analitik seperti histogram, kueri Top-N, dan tren hanya dengan beberapa klik.

3. Easily accessable dashboard

Dashboard yang disediakan kibana sangat mudah diatur sesuai yang kita perlukan

2.3..4 Auditbeat

Auditbeat adalah agen pengirim yang ringan yang dapat kita instal di server untuk mengaudit aktivitas pengguna dan proses pada sistem. Misalnya, kita dapat menggunakan Auditbeat untuk mengumpulkan dan memusatkan acara audit dari Linux Audit Framework. Auditbeat juga dapat digunakan untuk mendeteksi perubahan pada file penting, seperti file biner dan konfigurasi, dan mengidentifikasi kemungkinan pelanggaran kebijakan keamanan. Auditbeat berkomunikasi langsung dengan kerangka kerja audit Linux, mengumpulkan data yang sama dengan auditd, dan mengirimkan acara ke Elastic Stack secara real time. Pada auditbeat, kita dapat memonitor directory apapun pada sistem operasi yang kita pakai dan dapat digunakan sebagai *File Integrity Monitoring* Perubahan file dikirim secara real time ke Elasticsearch, masing-masing pesan berisi metadata dan hash kriptografi dari isi file untuk analisis lebih lanjut. Kirim ke Elasticsearch. Visualisasikan dalam Kibana.Auditbeat adalah bagian dari Elastic Stack.

2.3..5 Slack

Slack adalah salah satu *platform* komunikasi dimana pada aplikasi ini bisa digunakan di perangkat ganda dimana bisa mempermudah kita untuk melakukan komunikasi dengan mendukung intergrasi dengan beberapa *platform* lain dengan menggunakan *webhook*

2.3..6 Amazon Web Services

AWS adalah sebuah cloud platform, dimana pada project ini digunakan untuk pembuatan server-server yang nanti akan di gunakan untuk melakukan instalasi pada server-server tersebut. AWS merupakan salah satu anak perusahaan dari raksasa e-commerce dan perusahaan terbesar di dunia yaitu Amazon. Di project ini kita menggunakan beberapa layanan AWS seperti *Instance, Security group* , dan juga *Aws Elastic Ip.*

2.3..7 Ansible

Ansible merupakan sebuah *tool* Automation yang dapat digunakan untuk mempermudah kita untuk melakukan instalasi,*deployment, provisioning server*. Ansible berguna untuk mempermudah pekerjaan mengatur dan mengelolah server yang memiliki jumlah yang sangat banyak, dengan beragam aplikasi atau sofware di dalamnya. Ansible terkenal dengan kemudahan membaca konfigurasinya, Cara kerja ansible itu hanya dengan menggunakan koneksi SSH pada tiap server yang akan di jalankan *script* ansible ini

2.3..8 Elasalert

Salah satu *tools* yang berfungsi untuk melakukan *alerting* dimana dia mengambil data langsung dari database elasticsearch

NO	JUDUL	LINK	FOKUS	SARAN
1	*Scada statistics monitoring using the elastic stack (elasticsearch, logstash, kibana)*	https://pdfs.semanticscholar.org/5db9/ca3742ac99dcd6d498b5a3553fb716d99ce1.pdf	Monitoring mesin SCADA. ELK (elasticsearch, logstash, kibana). Sistem dapat digunakan, misalnya untuk mendeteksi situasi abnormal dan kesalahan konfigurasi alarm. Selain statistik ini setiap aplikasi menghasilkan file log berbasis teks yang diuraikan, dikumpulkan dan ditampilkan menggunakan Elastic Stack untuk memberikan terpusat akses ke semua log aplikasi.	-Elasticsearch -Kibana -Logstash -Filebeat
2	*Using Kibana and ElasticSearch for the Recommendation of Job Offers to Students*	https://pdfs.semanticscholar.org/6831/87d93124b9231e9cf62eb8669f6024739b4c.pdf	Monitoring untuk menyeleksi/ menampilkan rekomendasi efisien dari tawaran pekerjaan kepada siswa, agar memungkinkan mereka untuk meningkatkan karir profesional mereka. Kedua alat *Kibana* dan *ElasticSearch* terintegrasi dalam sistem web yang dikembangkan sendiri di UNED. Dengan menggunakan Kibana, pengguna bias dapatkan representasi grafis dari situasi pekerjaan untuk jangka waktu tertentu, yang bias menunjukkan kepada mereka apa pengetahuan yang paling dicari oleh perusahaan.	-Elasticsearch -Kibana
3	*Building a real-world*	https://hdms.bsz-	Monitoring tools, penjelasan secara	-Elasticsearch

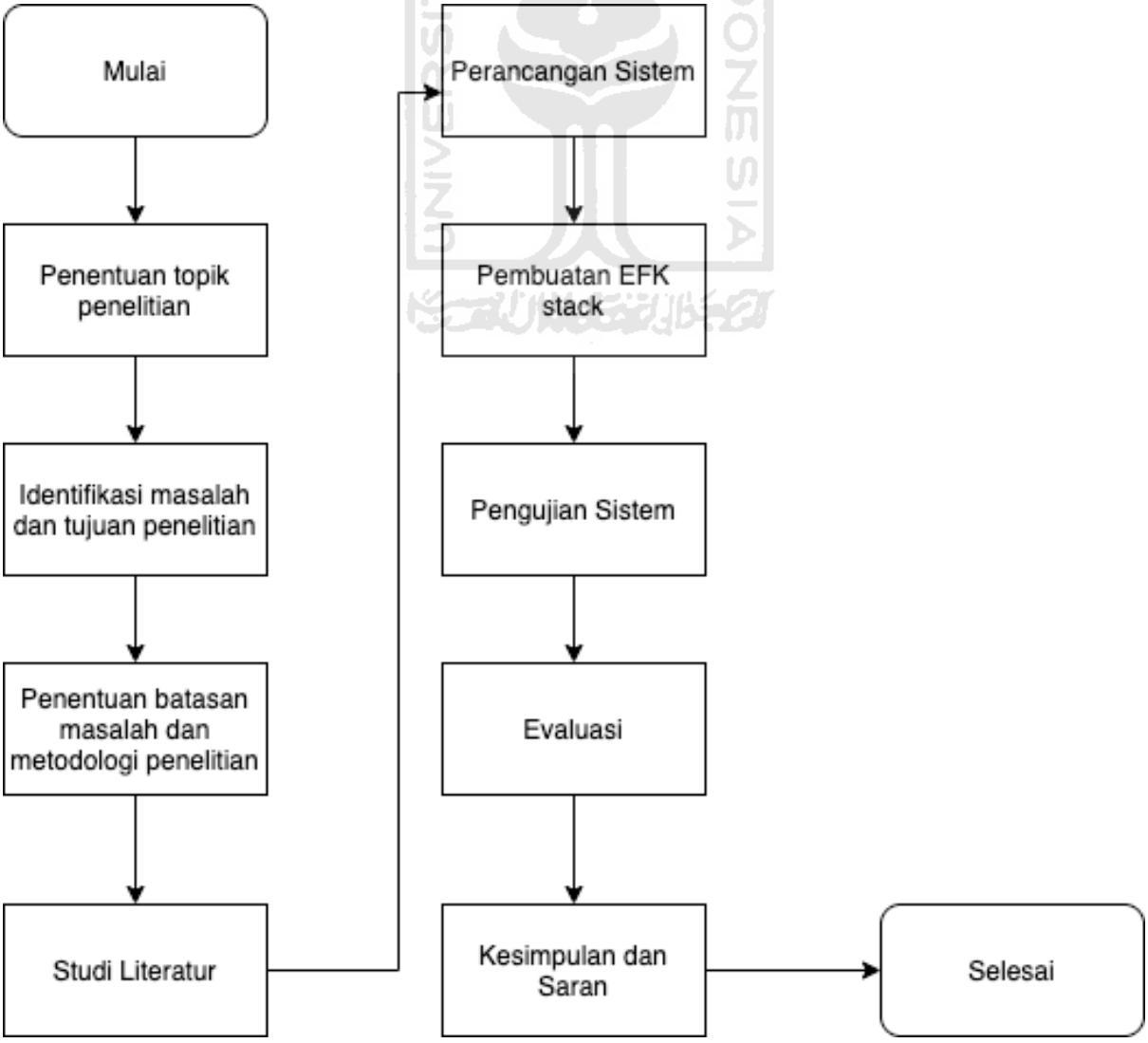
	logging infrastructure with Logstash, Elasticsearch and Kibana	bw.de/frontdoor/deliver/index/docId/5021/file/elk_paper_patrick_kleindienst.pdf	keseluruhan bagaimana membangun ELK stack, penggunaannya, dan alasan pemilihan	-Kibana -Logstash
4	*Log Real-time Management Scheme Based on LEK (2nd International Workshop on Materials Engineering and Computer Sciences (IWMECS 2015)*	https://download.atlantis-press.com/article/25840584.pdf		
5	*Monitoring of IaaS and scientific applications on the Cloud using the Elasticsearch ecosystem*	https://iopscience.iop.org/article/10.1088/1742-6596/608/1/012016/pdf	Monitoring tools untuk cloud base dengan menggunakan ELK stack yang terhubung dengan salah satu cloud infrastructure yang bernama OpenNebula.	-Elasticsearch -Kibana -Logstash
6	*The Data Management and Real-time Search Based on Elasticsearch (4th International Conference on Computer, Mechatronics, Control and Electronic Engineering (ICCMCEE 2015))*	https://pdfs.semanticscholar.org/f574/2f9b66f1e9a4a0fb9cf16492d986cbe40470.pdf?_ga=2.115299396.211430 8168.1567269789-981094237.1565768523	Mengulas tentang Elasticsearch yang memiliki 2 manfaat yang dapat digunakan yaitu sebagai "database" dan sebagai "Search engine" dengan menunjukan data secara real time.	-Elasticsearch -Kibana -Logstash
7	*E-commerce*	https://www.scienc	Fokus pada penggunaan	- Elasticsearch

	Personalization with Elasticsearch	edirect.com/science/article/pii/S1877050919306271	*elasticsearch* untuk mengimplementasi pendekatan baru untuk pencarian e-commerce hasil personalisasi yang mendekati waktu nyata yang memberikan hasil hasil personalisasi yang lebih baik dalam kerangka waktu terbatas yang diterima yang diperlukan untuk penelusuran online.	
8	*Elasticsearch tools: Beat (Lightweight Data Shippers*	https://www.elastic.co/products/beats	*Beats* adalah platform untuk pengirim data satu tujuan. Mereka mengirim data dari ratusan atau ribuan mesin dan sistem ke Logstash atau Elasticsearch.	-Filebeat -auditbeat -Metricsbeat -Packetbeat -heartbeat -functionbeat
9	*Elasticsearch tools: Auditbeat (Lightweight Data Shippers for Log*)	https://www.elastic.co/products/beats/filebeat	*Filebeat* adalah pengirim ringan untuk meneruskan dan memusatkan data log. Diinstal sebagai agen di server Anda, Filebeat memonitor file log atau lokasi yang Anda tentukan, mengumpulkan peristiwa log, dan meneruskannya ke Elasticsearch atau Logstash untuk pengindeksan.	

BAB III
METODOLOGI PENELITIAN

3.1 Skema alir Penelitian

Gambar 3.1 menunjukkan tahapan penelitian yang dilakukan oleh peneliti. Tahapan tersebut di antaranya adalah pendahuluan dan studi literatur, pengumpulan dan pengolahan data, dan kesimpulan dan saran.



Gambar 3.1 Skema Alir Penelitian

Berdasarkan Gambar 3.1, skema alir pada penelitian ini dibagi menjadi tiga tahapan, yaitu:

1. Tahapan Pendahuluan dan Studi Literatur

Tahap ini merupakan tahap awal pengerjaan sistem ini, tahap awalnya yaitu menentukan topik penelitian. Setelah topik ditentukan, langkah berikutnya adalah mengidentifikasi permasalahan dan melakukan batasan masalah. Kemudian peneliti menetapkan tujuan penelitian dan penggunaan metodologi penelitian. Studi literatur dilakukan setelah tahapan sebelumnya selesai dengan mengkaji beberapa penelitian serupa yang dapat dijadikan sebagai acuan dalam pengerjaan . Selain itu studi literatur juga digunakan untuk menghinari kesamaan topik dalam penelitian yang akan dilakukan

1. Tahapan Perancangan Sistem

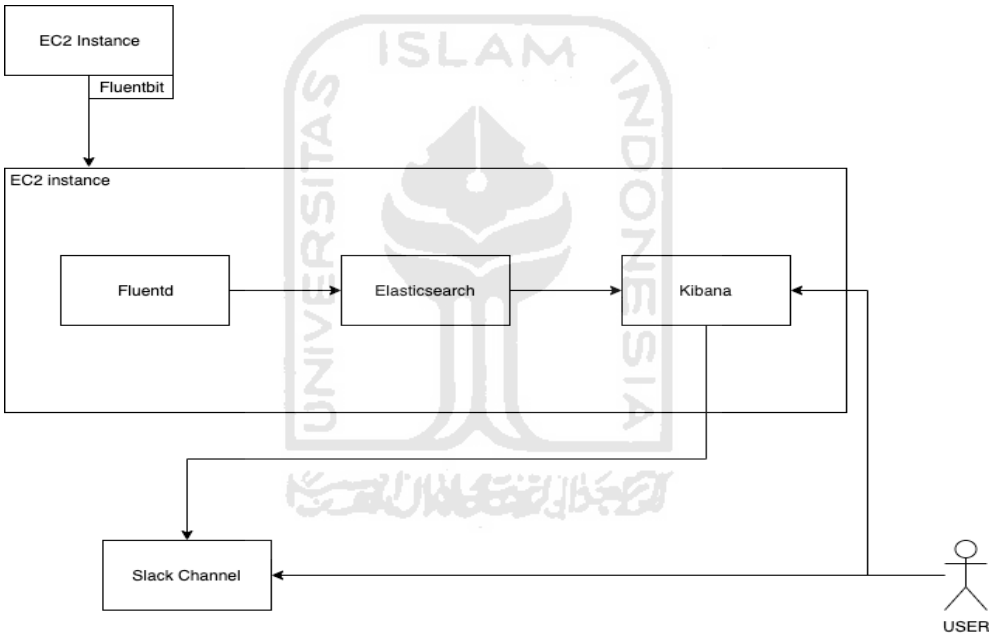
Tahapan ini berisi bagaimana penulis merancang sistem dengan membuat kerangka awal sebagai acuan pembuatan sistem dengan menggambarkan desain arsitektur sistemnya serta pemilihan alat-alat yang nantinya akan digunakan. Untuk alat-alat yang digunakan untuk mendukung sistem ini adalah sebagai berikut :

- AWS (amazon web services)
- Ansible
- Github
- Elasticsearch
- Fluentd
- Kibana
- Auditbeat
- Slack
- Elasalert

3.2 Skema Perancangan EFK Stack

Desain yang dibuat ada 2 desain untuk memudahkan pembaca dalam memahaminya, dibedakan sebagai :

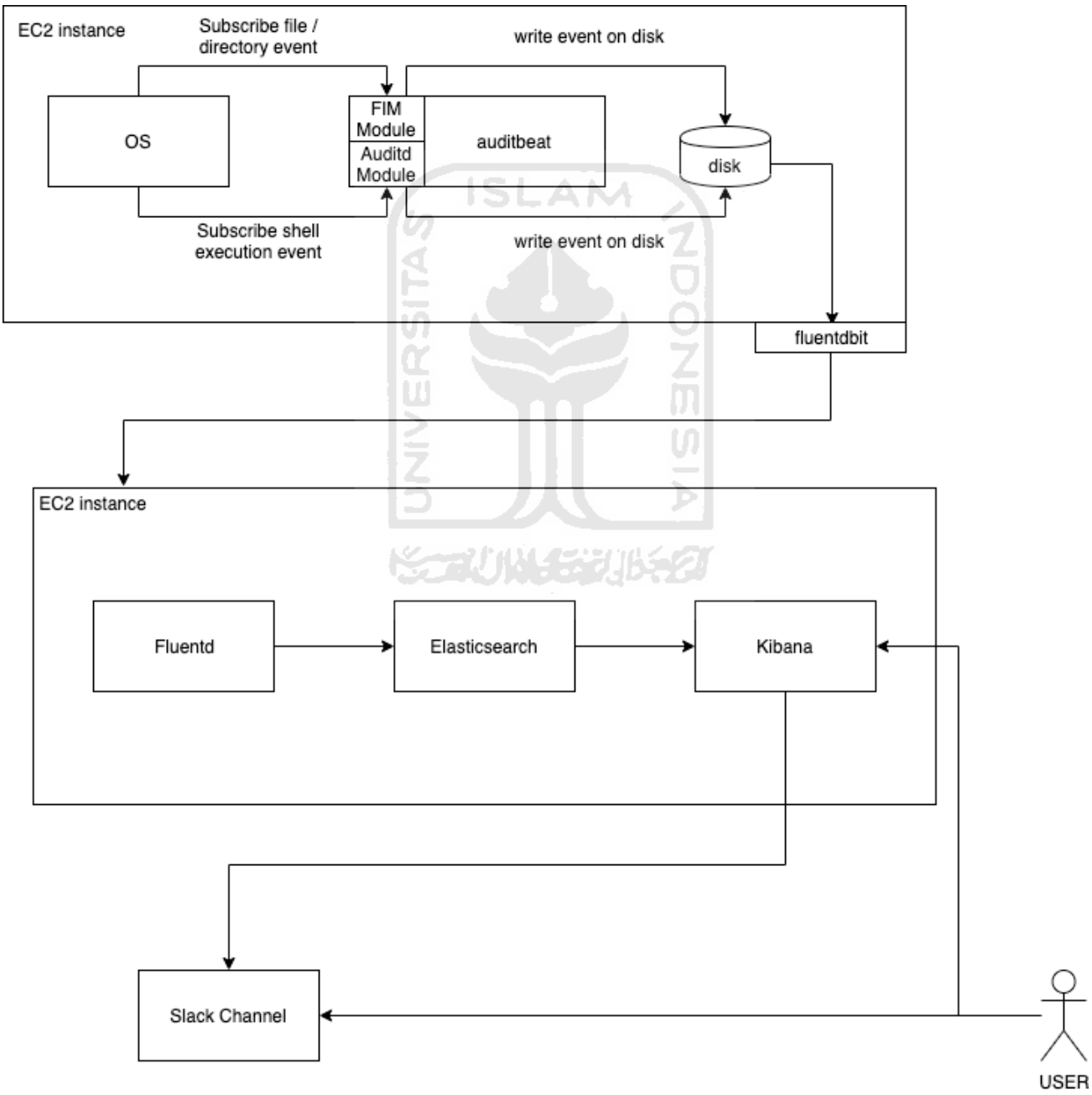
- Arsitektur sistem Authentifikasi



Gambar 3.2 Skema Perancangan arsitektur sistem Authentifikasi

Pada arsitektur sistem Authentifikasi, EC2 instance (server aws) yang dimana server ini yang akan kita monitoring. Pada desain sistem ini, agent fluentd yang ada pada server yaitu fluentbit akan mmengumpulkan data terkait authentifikasi yang ada pada server tersebut. Setelah terkumpul maka akan ditulis menuju server yang berisi EFK stack.

- Arsitektur sistem File integrity



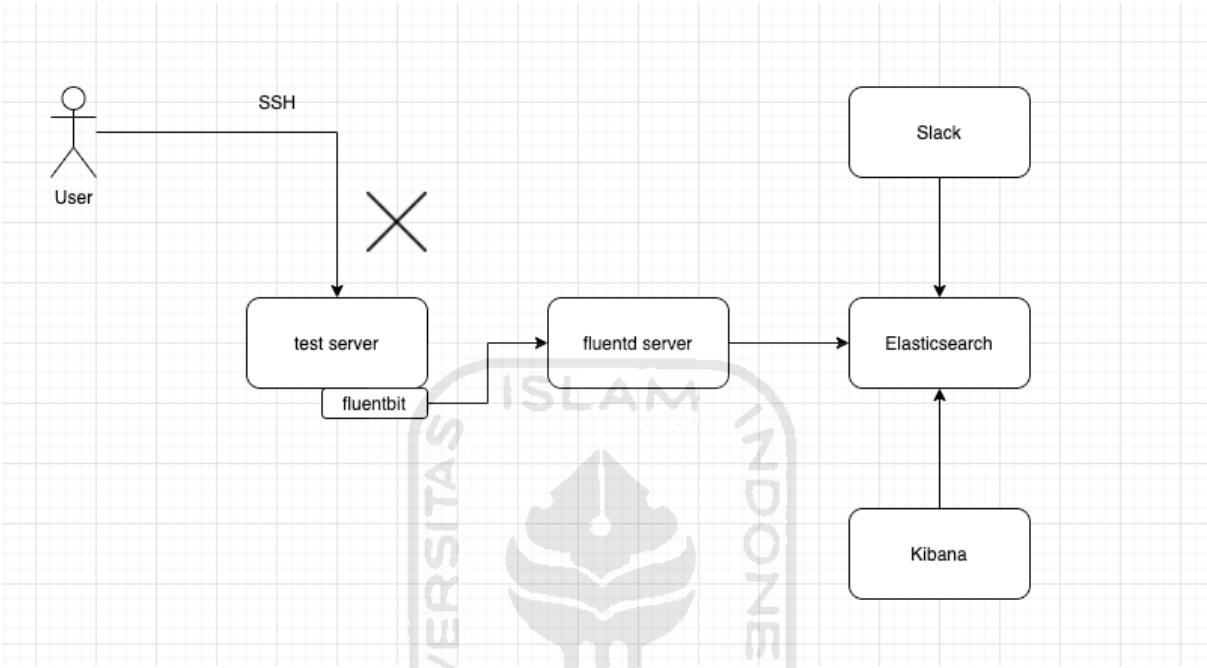
Gambar 3.3 Skema Perancangan arsitektur sistem File integrity

Pada arsitektur file integrity, EC2 instance (server aws) yang dimana server ini yang akan kita monitoring. Pada desain sistem ini,sebelum agent fluentd yang ada pada server yaitu fluentbit akan mengumpulkan data terkait authentifikasi yang ada pada server tersebut, ada proses terlebih dahulu yang dilakukan oleh agent Elastic yang bernama auditbeat , fungsi dari auditbeat itu sendiri adalah untuk memantau sistem yang berisi aktivitas pengguna dalam server tanpa menyentuh auditd. Auditbeat berkomunikasi langsung dengan kerangka kerja audit yang ada dilinux lalu mengumpulkan data yang sama dengan auditd, dan mengirimkan log log tersebut langsung ke EFK stack secara *real-time.* Setelah terkumpul maka akan ditulis menuju server yang berisi EFK stack.

4.4 Skema Pengujian

Setelah semua process yang dilakukan telah selesai tahapan berikutnya untuk peengujiannya adalah dengan cara memastikan log dari mesin yang dipantau dapat dikirimkan oleh fluentd menuju elasticsearch untuk disimpan dalam indexs elasticsearch, dan kiubana berhasil mendapatkan data dari elasticsearch yang nantinya kibana dapat memvisualisasikan log tersebut. Untuk pengujian ini kita bagi menjadi beberapa skema pengujian untuk mencapai ekspektasi yang diharapkan. Skema pengujian kita bagi menjadi 5 bagian yang akan dijelaskan dibawah ini :

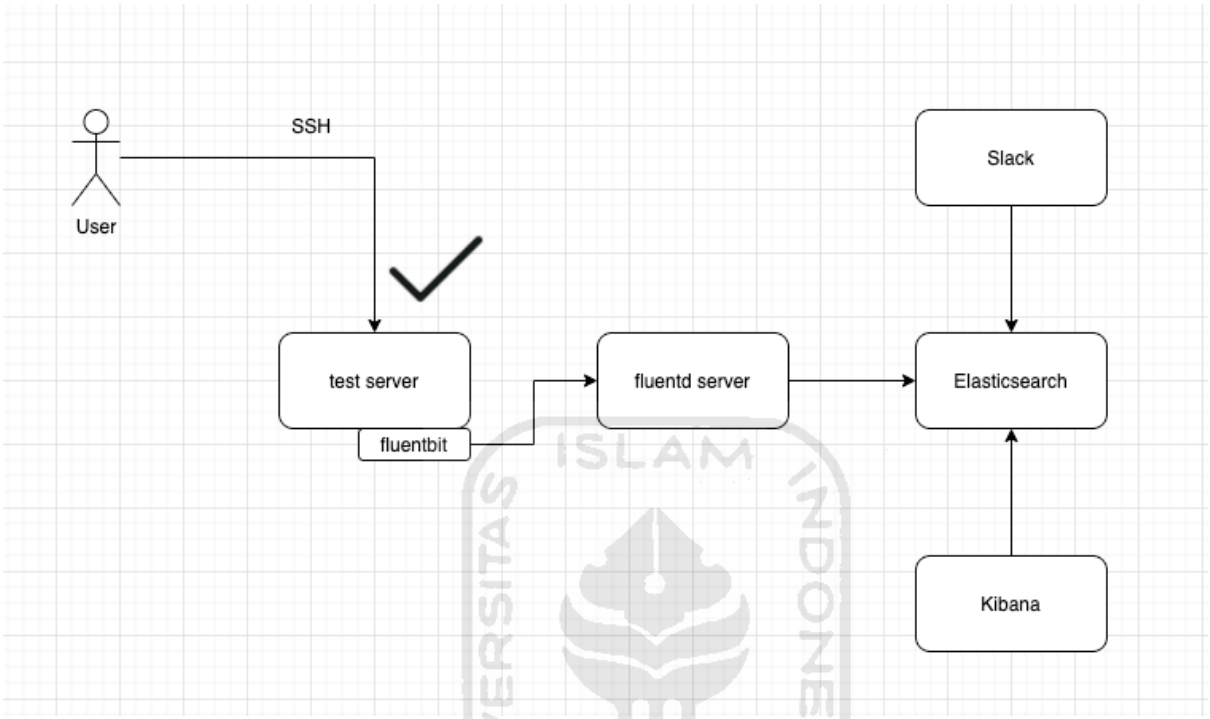
4.4.1 Skema Pengujian 1



Pada Skema Pengujian pertama , kita menggukan kasus pengguna yang tidak terotoriasi di server mencoba masuk kedalam server kita. Ekspektasi yang diharapkan adalah user gagal masuk kedalam server , lalu fluentbit mencatat log yang akan muncul pada log di *access.log* lalu mengirimkan log tersebut menuju kibana dan diteruskan menuju elasticsearch. Lalu Kibana akan mencoba menampilkan visualiasi yang lebih mudah dipahami

Selain itu kita membuat rule pada mesin kibana untuk membuat alert ketika ada *event* gagal login yang dilakukan dapat mengirimkan *alerting* dengan batasan rule yang kita buat, setelah itu pastikan notifikasi muncul pada slack channel untuk mempermudah visualisasi kita terhadap keamanan server

4.4.2 Skema Pengujian 2



Pada Skema Pengujian kedua , kita menggukan kasus pengguna yang terotoriasi di server mencoba masuk kedalam server kita. Ekspektasi yang diharapkan adalah user berhasil masuk kedalam server , lalu fluentbit mencatat log yang akan muncul pada log di *access.log* lalu mengirimkan log tersebut menuju kibana dan diteruskan menuju elasticsearch. Lalu Kibana akan mencoba menampilkan visualiasi yang lebih mudah dipahami

4.4.3 Skema Pengujian 3

Pada Skema Pengujian tiga , kita menggukan kasus pengguna yang terotoriasi di server mencoba masuk kedalam server kita lalu user akan melakukan eksekusi perintah pada *command line / terimal* dengan menggunakan sudo . Ekspektasi yang fluentbit mencatat log yang akan muncul pada log di *access.log* lalu mengirimkan log tersebut menuju kibana dan diteruskan menuju elasticsearch. Lalu Kibana akan mencoba menampilkan visualiasi yang lebih mudah dipahami

4.4.4 Skema Pengujian 4

Pada Skema Pengujian tiga , kita menggukan kasus pengguna yang terotoriasi di server mencoba masuk kedalam server kita lalu user akan melakukan perubahn dan penghapusan pada suatu file system yang berada pada *directory* /etc/. Ekspektasi yang fluentbit mencatat log yang akan muncul pada log di *audtibeat log* lalu mengirimkan log tersebut menuju kibana dan diteruskan menuju elasticsearch. Lalu Kibana akan mencoba menampilkan visualiasi yang lebih mudah dipahami

4.4.5 Skema Pengujian 5

Pada skema pengujian terakhir merupakan lanjutan dari skema pengujian 4 dengan kita membuat rule elasalert yang nantinya akan memberitau pada kita melalui slack notif yang dikirimkan ketika adanya perubahan yang terjadi pada filesystem

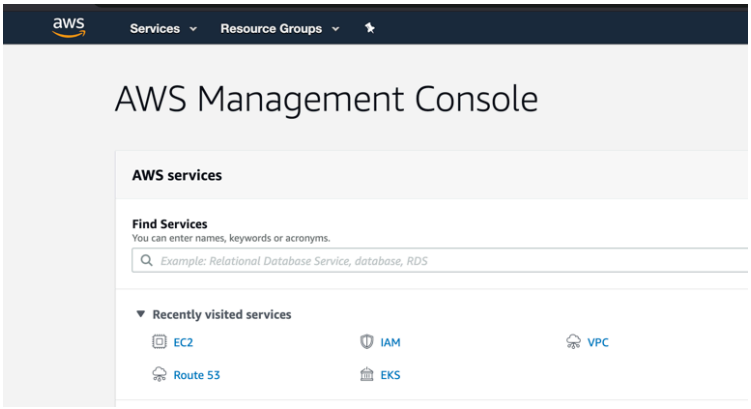
BAB IV
Hasil dan Pembahasan

Dalam penelitian ini, Penulis melakukan perancangan keseluruhan server dengan membagi beberapa server sesuai dengan peranan atau tugas yang telah ditentukan. Penulis membuat 4 buat server dimana server itu dibagi menjadi, 1 server untuk testing dan 3 server untuk pembuatan EFK stack. Lalu ke empat server itu akan dijalankan secara bersamaan hingga menciptakan keselarasan service yang bekerja secara beraturan.

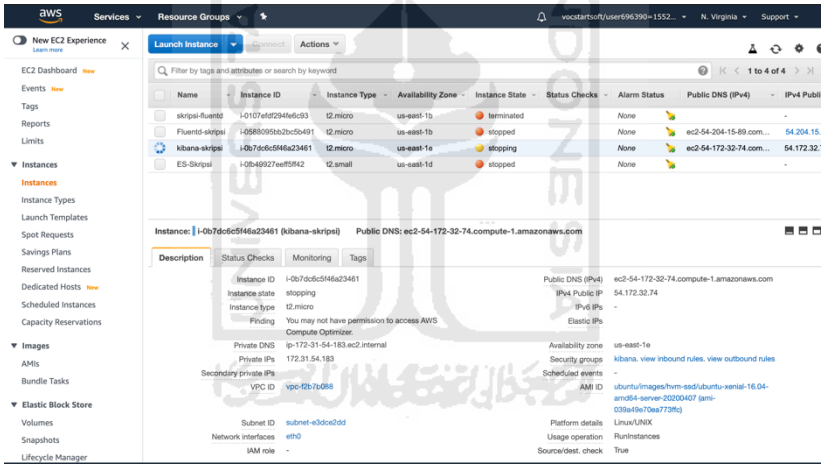
2.3 Pembuatan dan Provisioning server

5.1.1 Pembuatan server

Tahapan awal yang dilakukan oleh penulis adalah melakukan pembuatan server pada salah satu web service yaitu aws. Dimana di AWS ini akan dibuat beberapa mesin yang digunakan untuk tiap services yang dibuat. Mesin yang dibuat ada empat mesin, satu mesin untuk mesin yang akan dipatau, dan 3 mesin lainnya digunakan untuk EFK stack yang akan dibuat. Mesin yang dibuat berukuran t2.nano dan ada satu mesin berukuran t2.small . Perbedaan mesin ini ada pada kapasitas CPU dan memory yang berbeda. T2.mikro memiliki 1 cpu dan 1gb ram, sedangkan t2.small memiliki 1 cpu dengan ram yang lebih besar yaitu 2gb ram. Mengapa ada perbedaan size mesin yang digunakan, dikarenakan ada salah satu mesin yang berperan sebagai database untuk menyimpan data dan service tersebut membutuhkan ram yang cukup untuk menjalankan java yang memakan cukup memori. Pada gambar 4.1 dan 4.2 akan ditunjukan bagaimana tampilan awal dashboard amazon web service yang nantinya akan kita gunakan.

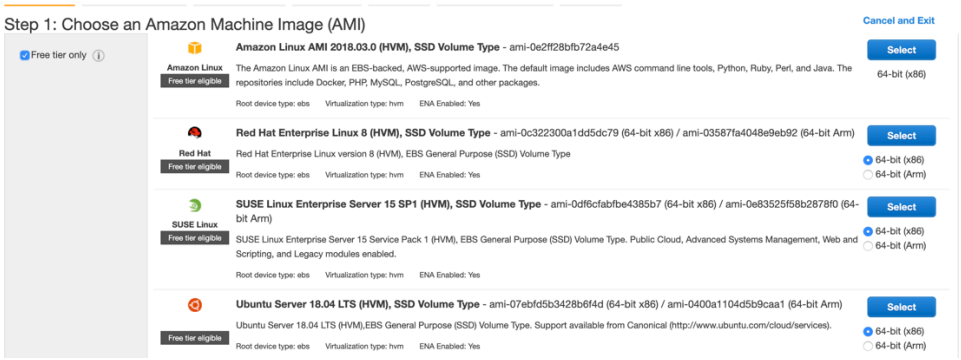


Gambar 4.1 Dashboard Awal AWS console



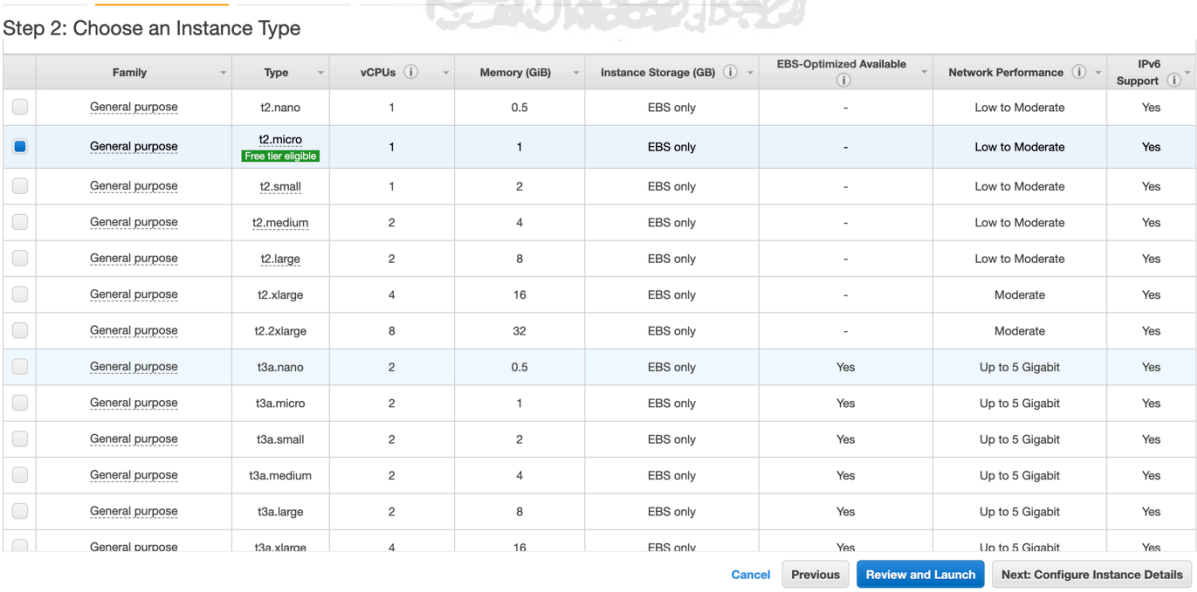
Gambar 4.2 AWS Instances

Semua mesin yang dibuat adalah mesin dengan menggunakan system operasi Linux distro Ubuntu dengan Ubuntu versi 16. Pada Gambar 4.3 ditunjukan pada saat kita membuat mesin, kita dapat memilih menggunakan system operasi yang mana. Ada beberapa mesin yang dapat digunakan secara gratis, ada sekitar 18 Amazon Machine image yang dapat digunakan secara gratis serta ada 1 instance juga dengan size t2.mkiro yang dapat digunakan secara gratis tanpa dipungut biaya dari aws. Pemilihan Ubuntu versi 16 ini didasarkan pada config code yang dibuat hanya dapat digunakan pada Ubuntu versi 16 .



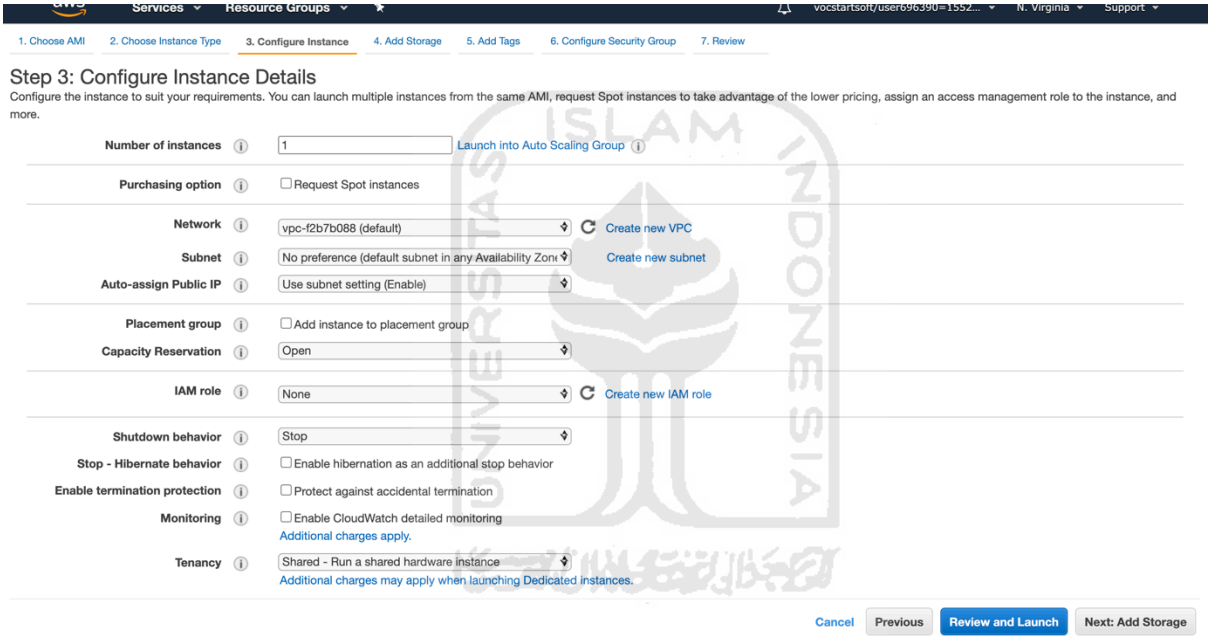
Gambar 4.3 Amazon Machine Image

Setelah kita menentukan untuk memeilih mesin mana yang akan kita gunakan , maka kita akan melakukan konfigurasi terkait besar mesin yang akan digunakan. Dalam project ini penulis menggunakan 3 mesin t2.micro dan 1 mesin berukuran t2.small. Penulis mempertimmbangkan efektifitas serta penggunaan service yang nanti akan berjalan diatas server ini. Mesin t2.small digunakan untuk service elasticsearch yang lumayan cukup memerlukan memory yang lebih besar untuk menjalankan servicenya. Perhatikan Gambar 4.4 dibawah untuk lebih jelas terkait ukuran instance yang tersedia pada aws ec2.



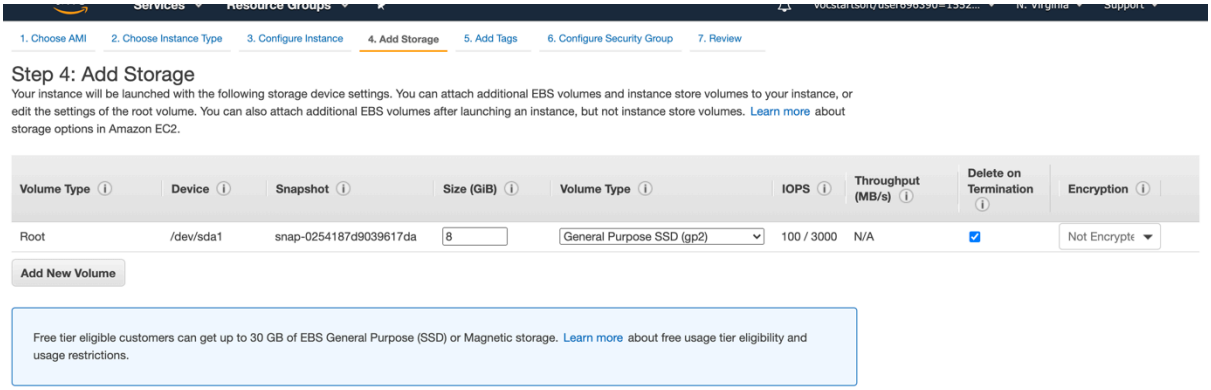
Gambar 4.4 Choose an instance type

Langkah selanjutnya dalam pembuatan instance ec2 aws ini adalah konfigurasi mesin secara detail. Maksud detail disini adalah seperti berapa jumlah mesin yang akan kita buat, lalu pemilihan detail network yang akan kita pilih seperti mesin ini menggunakan vpc apa , subneting untuk ip nya serta apakah dalam mesin perlu kita sertakan *public ip address*. Konfigurasi I AM role dimana digunakan untuk permission mesin ini, lalu konfigurasi lainnya yang dapat kita simak pada Gambar 4.5



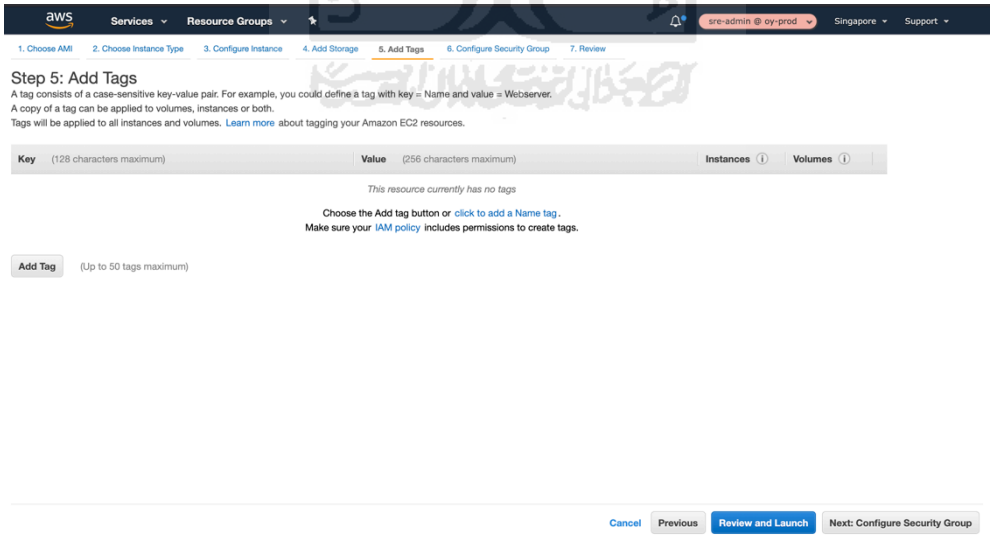
Gambar 4.5 Configure Instance Details

Konfigurasi terkait mesin secara detail telah dilakukan maka selanjutnya kita akan melakukan pengaturan penyimpanan (*storage)* yang dibutuhkan pada mesin kita, kita bisa menentukan besar ukuran storage yang kita butuhkan. Gambar 4.5 akan menjelaskan detail bagaimana kita akan mengkonfigurasi mesin yang dibuat

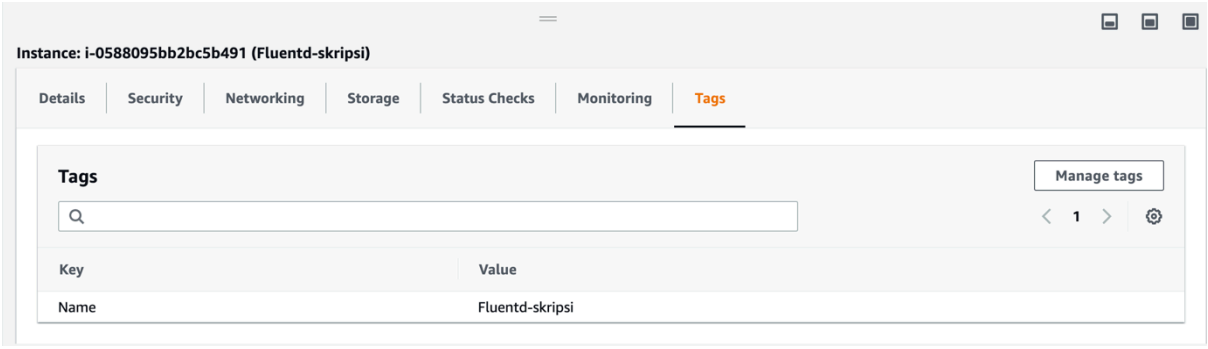


Gambar 4.6 Add Storage

Langkah kelima dalam pembuatan instance ini adalah menambahkan tags yang kita butuhkan. Tags ini sebagai penanda atau bisa dikatakan sebagai detail dari mesin yang kita buat, seperti kita akan menambahkan tags nama, atau tags yang lain sesuai apa yang kita butuhkan. Gambar dibawah ini akan menunjukkan bagaimana cara kita mengeset tags dan hasil yang akan terbentuk

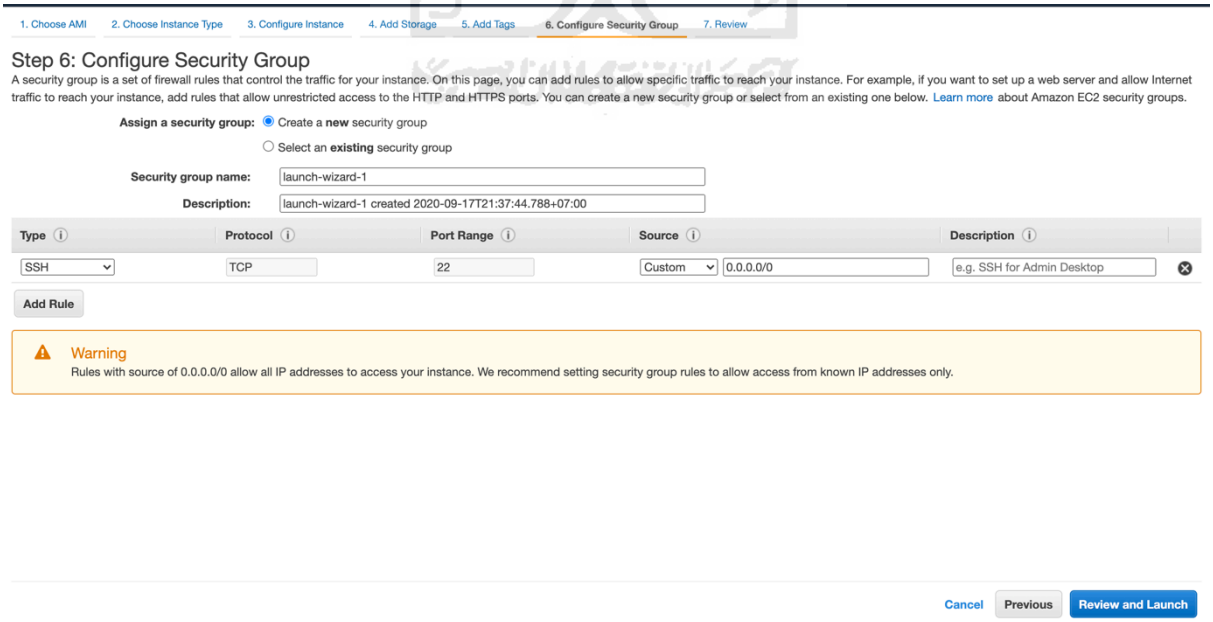


Gambar 4.7 Add tags



Gambar 4.8 Hasil tags

Langkah sebelum kita mereview semua yang sudah kita setup adalah melakukan konfigurasi terkait security groups. Security groups Grup bertindak sebagai firewall virtual untuk instans EC2 kita yang berfungsi untuk mengontrol lalu lintas masuk dan keluar. Dalam konfigurasi ini berisi aturan masuk mengontrol lalu lintas masuk ke instance , dan aturan keluar mengontrol lalu lintas keluar dari instance Jika kita tidak menentukan security group ini, Amazon EC2 menggunakan grup keamanan default.



Gambar 4.9 Configure Security Group

5.1.2 Provisioning Server

Provisioning server adalah menyiapkan dan mengkonfigurasi sebuah sistem dengan menyiapkan segala kebutuhannya sehingga sebuah s*service* atau *aplication* dapat berjalan pada mesin yang telah kita sediakan. Sebagai contoh semisal kita akan membuat aplikasi yang akan dibuat pada sebuah ec2 kita memerlukan beberapa hal yang mendukung aplikasi tersebut agar bisa berjalan , misal kita memerlukan java untuk mendukung aplikasi yang kita buat bisa berjalan. *Provisioning* adalah hal yang sangat penting ketika kita sedang melakukan setup sebuah mesin baru . Dalam projek ini penulis menggunakan salah satu tools automasi untuk memudahkan penulis dalam melakukan *provisioning* beberapa mesin yang telah dibuat. Tools yang digunakan adalah *Ansible* . Pemilihan ansible ini di dasari oleh kemudahan dalam pengaplikasian penyediaan server dalam artian instalasi bebagai kebutuhan dengan cukup mudah, selain itu ansible juga memudahkan seseorang untuk lebih mudah memahami sesuatu dari pada hanya dengan membaca code biasa . Selain itu ansible juga menjalankan task yang kita buat dengan berurutan satu persatu hingga task terakhir selesai. Pada ansible ada beberapa komponen yang perlu kita pahami yaitu :

- Playbook
- Task
- Variable
- Roles

Dibawah ini akan dijelaskan bagaimana konfigurasi ansible untuk ketiga stack yang akan kita buat beserta fungsi dan tujuannya yang akan dijelaskan pada gambar 4.10

```
danur@infplay-ansible-172-50-34-37:~/repo/infra_ansible$ sudo ls playbooks/infcentrallog/
elasticsearch.yml  files  fluentd.yml  kibana.yml
danur@infplay-ansible-172-50-34-37:~/repo/infra_ansible$
```

Gambar 4.10 Playbook EFK stack

Pada gambar tersebut ditunjukan dalam playbook ansible tersebut ada 4 file . Ada 3 buah file yang berekstensi ".yml" merupakan file yang digunakan untuk ansible melakukan pekerjaannya. Playbook pada ansible merupakan konfigurasi inti berisi urutan pekerjaan (*task*) operasi, yang nenentukan dari mana terlebih dahulu . cara menjalankan playbook itu cukup mudah dengan menggunakan command seperti berikut

```
Ansible-playbook "file yml yang dibuat" -v
```

Sebagai contoh semisal kita akan melakukan eksekusi untuk file bernama kibana.yml maka command pada ansiblenya akan seperti ini

```
Ansible-playbook playbook/skripsi/kibana.yml -v
```

lalu bagaimanakah isi dari dari tiap-tiap playbook tersebut? Dari tiap file yml itu berisi berbeda beda sesuai nama yang diberikan. Pada playbook ini sendiri berisi beberapa *task* maupun role yang nanti akan dijalankan berurutan ketika ada command ansible yang dijalankan. Role pada ansible berisi tentang semua hal terkait isi dari konten dari sebuah service yang akan berjalan seperti , *task, variable, template,handlers*. Ansible Role biasa kita sebut dengan Struktur file. Pada project ini memiliki cukup banyak role yang dipanggil untuk menjalankan tasknya. Berikut isi dari playbook yang nanti akan di eksekusi oleh ansible :

```yaml
---
- hosts: localhost
  become: yes
  pre_tasks:
    - name: ensure ES data/es dir is exist
      file:
        path: '/data/es'
        state: directory
        owner: root
        group: root
        mode: '0755'
      become: true
  roles:
    - role: shared/common
    - role: shared/monitoring
    - role: shared/xfs_volume
      xfs_volume_mount: /data/es
    - role: shared/elasticsearch
      elasticsearch_version_family: 7.x
      elasticsearch_version: 7.7.0
      es_jvm_heap_min: "1g"
      es_jvm_heap_max: "1g"
```

Gambar 4.11 Playbook Elasticsearch

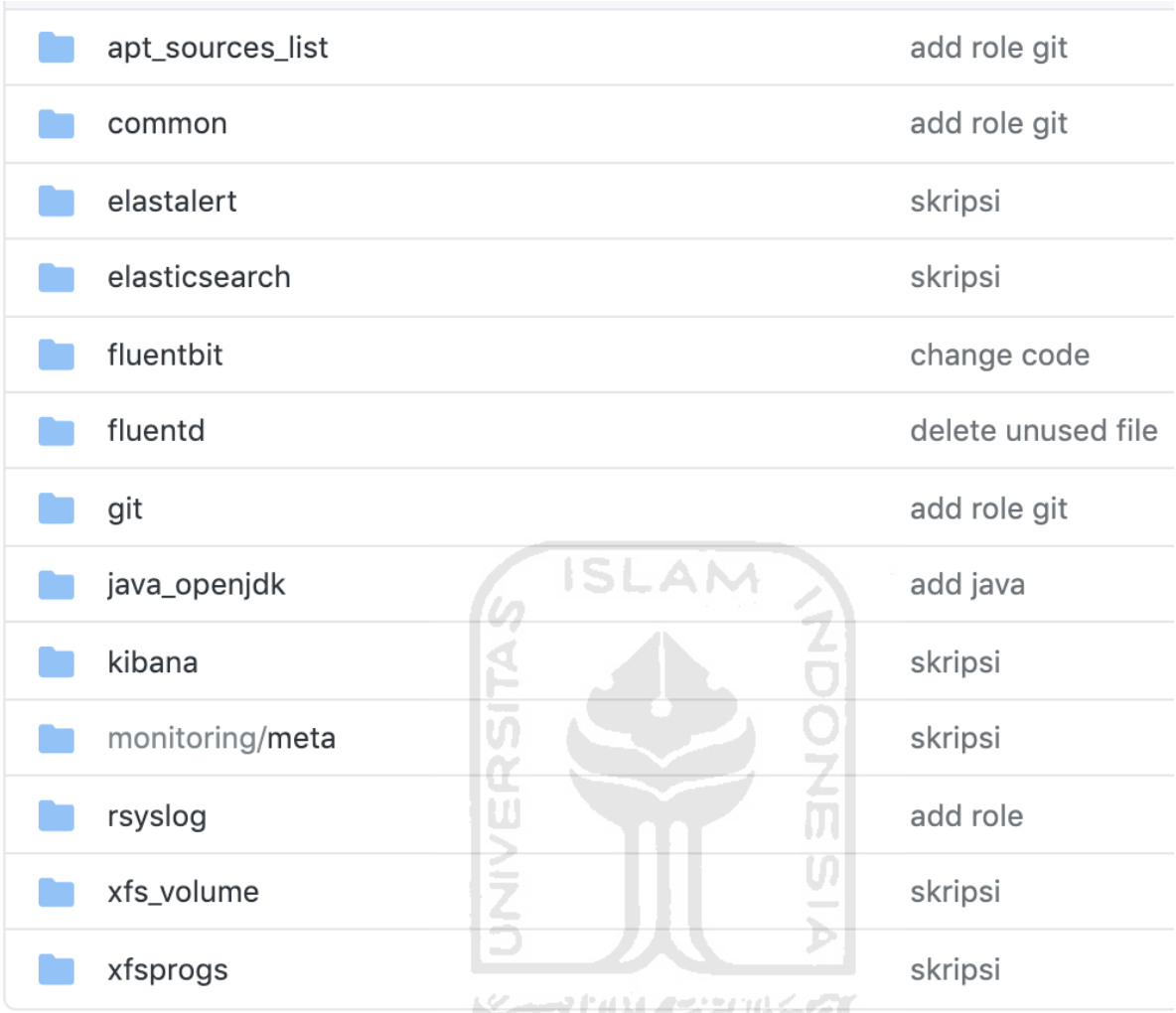
```yaml
---
- hosts: localhost
  become: yes
  roles:
    - role: shared/common
    - role: shared/monitoring
    - role: shared/fluentd
      tdagent_plugins:
        - fluent-plugin-elasticsearch
      tdagent_conf_copy: "files/td-agent.conf"
```

Gambar 4.12 Playbook Fluentd

```
---
- hosts: tag_Name_infcentrallog-kibana
  become: yes
  roles:
    - role: shared/common
    - role: shared/monitoring
    - role: shared/kibana
      kibana_version: 6.8.2
      ansible_kibana_cluster_name: "Danur Skripsi logging dashboard"
      kibana_es_endpoint: "infcentralloges.skripsi:9200"
    - role: shared/elastalert
      elastalert_es_host: "infcentralloges.skripsi"
```

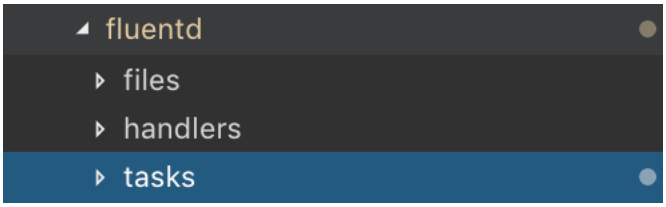
Gambar 4.13 Playbook Kibana

Pada ketiga gambar diatas ditunjukan bagaimana susunan atau isi playbook pada project yang dikerjakaan. Ada beberapa komponen yang ada seperti *Host* dimana ini menunjukan dimana ansible akan melakukan atau menjalankan tugasnya. *Become : yes* menunjukan ekskalasi *previllege* sebagai *sudo* pada Ubuntu. *Pre_task* merupakan task awal sebelum ansible menjalankan role yang ada . *Pre_task* ini hanya ada pada *playbook* untuk instalasi elasticsearch yang bertujuan untuk melakukan apakah sebuah direktori dengan *path* terkait itu ada. Lalu ada Role yang berfungsi sebagai struktur file yang berisi tentang banyak hal seperti *task, variable file, templates,* dan handlers. Tujuan pembuatan role ini adalah untuk mempermudah kita dalam penggunaannya , dalam artian kita dapat dengan mudah mengambil atau menggunakan role pada playbook manapun dan membuat code terlihat jauh lebih rapi daripada kita melakukan penulisan tasknya langsung pada playbook. Lalu bagaimanakah konfigurasi *role* yang ada , dan bagaimanasaja susunan file-file yang ada. Berikut akan dijelaskan pada Gambar 4.11 yang ada dibawah ini



Gambar 4.14 Playbook Fluentd

Isi dari Role itu sendiri ada banyak sekali tergantung bagaimana kita menyusunnya, pada project ini susunan folder role berdasarkan service yang dibutuhkan apa saja. Seperti ada folder untuk *EFK stack* itu sendiri dan ada juga folder untuk pendukung mesin. Untuk lebih focus terhadap *stack* yang digunakan akan ditunjukan dibawah ini untuk isi dari task di masing-masing role yang ada pada pada *EFK stack* ini



Gambar 4.15 Role Fluentd

Isi dari folder role fluentd ada 3 bagian, *file, handlers* dan *task.* Tiap bagian tersebut behubungan satu dengan yang lainnya. Untuk task apa saja yang akan dijalankan pada role fluentd ini akan di jabarkan pada code yang ada dibawah ini

```
- name: set role variables, if necessary
  include: set-role-variables.yml


- name: delegate to APT system for installation
  include: use-apt.yml
  when: ansible_pkg_mgr == "apt"

- name: mkdir for conf.d
  file:
    path: /etc/td-agent/conf.d
    state: directory
    owner: td-agent
    group: td-agent

- name: rename default td-agent.conf
  command: creates="/etc/td-agent/td-agent.conf.bak" mv /etc/td-agent/td-agent.conf /etc/td-agent/td-agent.conf.bak

- name: copy new td-agent.conf from role's default
  copy:
    src: ../files/td-agent.conf
    dest: /etc/td-agent/td-agent.conf
    owner: td-agent
    group: td-agent
  when: tdagent_conf_copy is not defined and tdagent_conf_template is not defined
  notify:
    - reload td-agent
```

Gambar 4.16 Fluentd task1

```
- name: copy new "td-agent.conf" verbatim from main playbook's
  copy:
    src:  "{{ playbook_dir }}/{{ tdagent_conf_copy }}"
    dest: /etc/td-agent/td-agent.conf
    owner: td-agent
    group: td-agent
  when: tdagent_conf_copy is defined
  notify:
    - reload td-agent

- name: copy (from template) new "td-agent.conf" from main playbook's
  template:
    src:  "{{ playbook_dir }}/{{ tdagent_conf_template }}"
    dest: /etc/td-agent/td-agent.conf
    owner: td-agent
    group: td-agent
  when: tdagent_conf_template is defined
  notify:
    - reload td-agent

- name: copy other config files from playbook's, if any
  template:
    src:  "{{ playbook_dir }}/{{ item.value.src }}"
    dest: "/etc/td-agent/conf.d/{{ item.value.dest }}"
    owner: td-agent
    group: td-agent
  with_dict: '{{ tdagent_conf_others | default({}) }}'
  notify:
    - reload td-agent
```

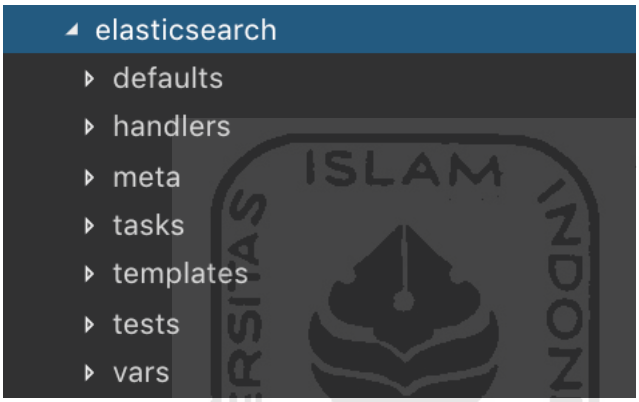
Gambar 4.17 Fluentd task2

```
# GEM installed on:
#   /opt/td-agent/embedded/lib/ruby/gems/2.1.0/gems/
- name: install plugins, if any
  gem:
      name: "{{ item }}"
      executable: /opt/td-agent/embedded/bin/fluent-gem
      state: latest
      user_install: no
  with_items: "{{ tdagent_plugins | default([]) }}"
  notify:
    - restart td-agent


- name: install plugins with specified versions, if any
  gem:
      name: "{{ item.value.name }}"
      executable: /opt/td-agent/embedded/bin/fluent-gem
      state: present
      version: "{{ item.value.version }}"
      user_install: no
  with_dict: "{{ tdagent_plugins_versions | default({}) }}"
  notify:
    - restart td-agent
```

Gambar 4.18 Fluentd task3

Untuk Role elasticserach akan di tunjukan pada Gambar 4.12 , file yang dibuat pada elasticsearch role ini lebih banyak dari yang dimiliki oleh fluentd. Ada sekitar sekitar 8 folder yang saling berhubungan satu sama lain.



Gambar 4.19 Role Elasticsearch

. Untuk task apa saja yang akan dijalankan pada role elasticsearch ini akan di jabarkan pada code yang ada dibawah ini:

```
---
- block:
    - name: add apt signing key elasticsearch
      apt_key:
        url: https://artifacts.elastic.co/GPG-KEY-elasticsearch
        state: present
        id: D88E42B4

    - name: install package dependencies
      apt:
        name: "{{ item }}"
        state: present
      with_items:
        - apt-transport-https
```

Gambar 4.20 task Elasticsearch 1

```yaml
- name: add elasticsearch repo to sources.list
  apt_repository:
    repo: "deb https://artifacts.elastic.co/packages/{{ elasticsearch_version_family }}/apt stable main"
    state: present
    filename: "elastic-{{ elasticsearch_version_family }}"
    update_cache: yes

- name: install elasticsearch
  apt:
    name: "elasticsearch={{ elasticsearch_version }}"
    state: present
    force: yes

- name: ensure ES data/es dir is exist
  file:
    path: "{{ item }}"
    state: directory
    owner: elasticsearch
    group: elasticsearch
    mode: '0755'
  with_items:
    - /data
    - /data/es

- name: ensure elasticsearch service running and enabled on boot
  service:
    name: elasticsearch
    enabled: yes
    state: started

- name: copy es main configuration
  template:
    src: "elasticsearch_config.yml.tpl"
    dest: "{{ es_config_dir }}/elasticsearch.yml"
    owner: root
    group: elasticsearch
    mode: '0660'
  notify: restart elasticsearch
```

Gambar 4.18 task Elasticsearch 2

```
  - name: copy es jvm configuration
    template:
      src: "jvm.options.tpl"
      dest: "{{ es_config_dir }}/jvm.options"
      owner: root
      group: elasticsearch
      mode: '0660'
    notify: restart elasticsearch

  - block:
    - name: check whether ingest-* plugins are installed
      command: "{{ es_binary_dir }}/elasticsearch-plugin list"
      register: list_result

    - name: install additional recommended plugins
      command: "{{ es_binary_dir }}/elasticsearch-plugin install {{ item }} --batch"
      with_items:
        - ingest-user-agent
        - ingest-geoip
      notify: restart elasticsearch
      ignore_errors: yes
      when: "'ingest-user-agent' and 'ingest-geoip' not in list_result.stdout_lines"

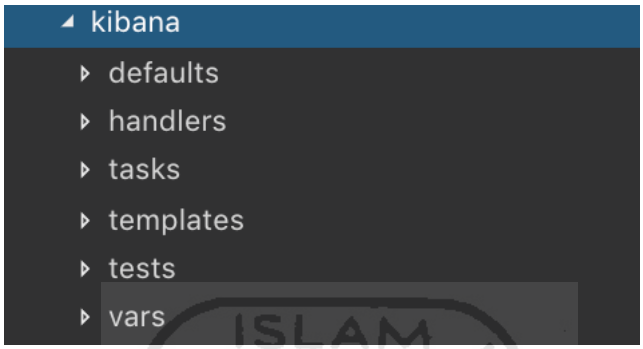
  - block:
      - name: check whether x-pack is installed
        command: "{{ es_binary_dir }}/elasticsearch-plugin list"
        register: elasticsearch_plugin_list_result

      - name: install elastic xpack source
        command: "{{ es_binary_dir }}/elasticsearch-plugin install x-pack --batch"
        notify:
          - restart elasticsearch
        when: "'x-pack@{{ elasticsearch_version }}' not in elasticsearch_plugin_list_result.stdout_lines"
    when: elasticsearch_xpack_enabled

become: true
tags:
  - elasticsearch
```

Gambar 4.22 task Elasticsearch 3

Untuk Role Kibana akan di tunjukan pada Gambar 4.13 , file yang dibuat pada kibana role ini lebih banyak dari yang dimiliki oleh fluentd. Ada sekitar sekitar 8 folder yang saling berhubungan satu sama lain.



Gambar 4.23 Role kibana

. Untuk task apa saja yang akan dijalankan pada role kibana ini akan di jabarkan pada code yang ada dibawah ini:

```
- block:
    - name: add apt signing key kibana
      apt_key:
        url: https://artifacts.elastic.co/GPG-KEY-elasticsearch
        state: present
        id: D88E42B4

    - name: install package dependencies
      apt:
        name: "{{ item }}"
        state: present
      with_items:
        - apt-transport-https
```

Gambar 4.24 task kibana 1

```yaml
  - name: add kibana repo to sources.list
    apt_repository:
      repo: "deb https://artifacts.elastic.co/packages/{{ kibana_version_family }}/apt stable main"
      state: present
      update_cache: yes

  - name: install kibana
    apt:
      name: "kibana={{ kibana_version }}"
      state: present

  - name: ensure kibana service running and enabled on boot
    service:
      name: kibana
      enabled: yes
      state: started

  - block:
      - name: check whether x-pack is installed
        command: "{{ kibana_binary_dir }}/kibana-plugin list"
        register: kibana_plugin_list_result

      - name: install elastic xpack source
        command: "{{ kibana_binary_dir }}/kibana-plugin install x-pack"
        notify:
          - restart kibana
        when: "'x-pack@{{ kibana_version }}' not in kibana_plugin_list_result.stdout_lines"
    when: kibana_xpack_enabled

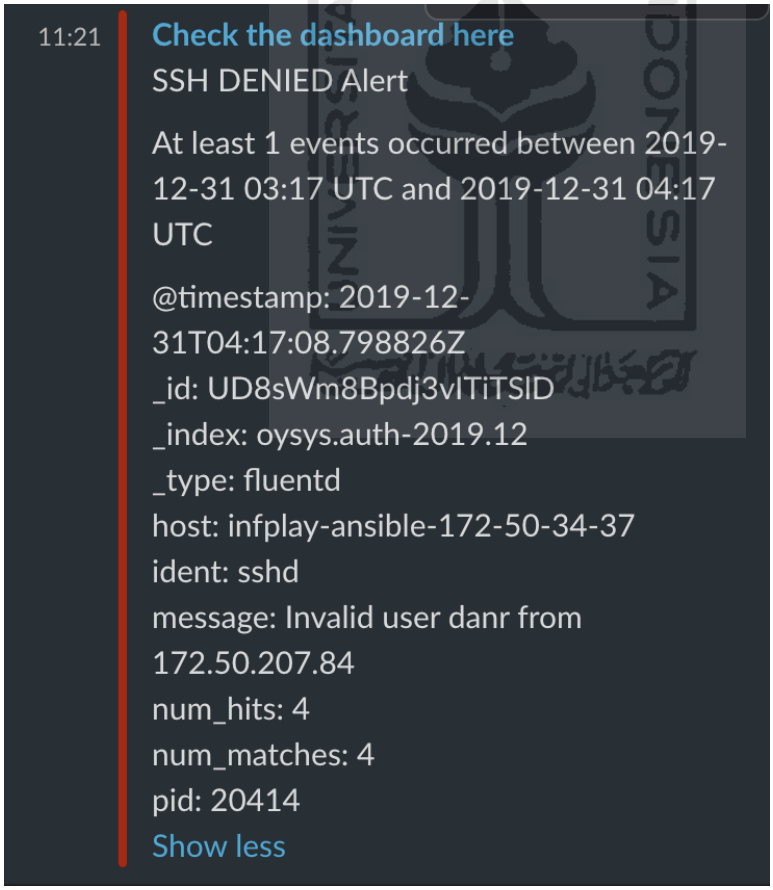

  - name: copy kibana basic config setup
    template:
      src: kibana_config.yml.tpl
      dest: "{{ kibana_config_dir }}/kibana.yml"
      owner: root
      group: root
      mode: '0644'
    notify:
      - restart kibana

become: true
become_method: sudo
tags:
  - kibana
```

Gambar 4.25 task kibana 2

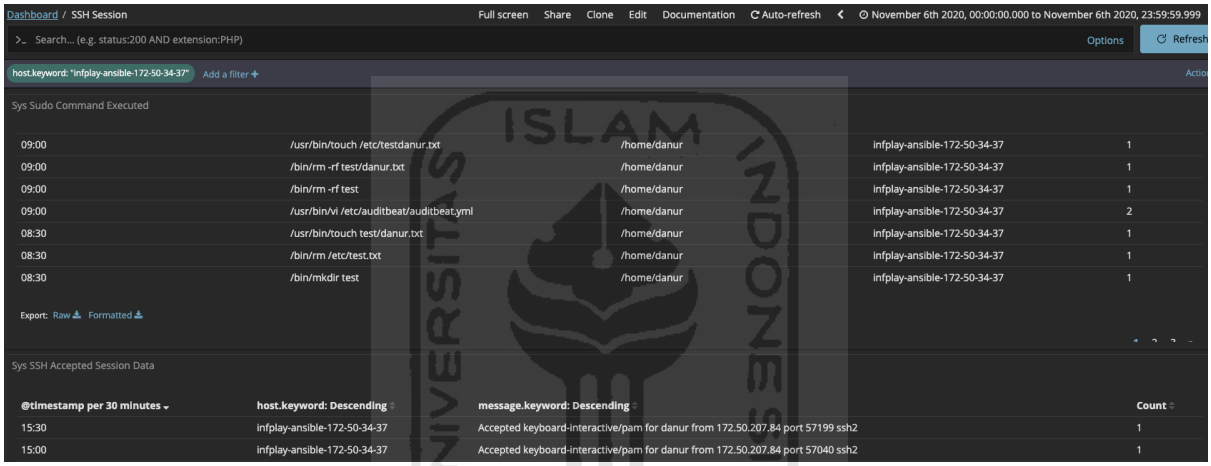
2.3 Hasil

Pada project ini penulis bertujuan untuk memonitoring segala aktivitas user yang ada pada server dengan cara mengambil log-log yang berkaitan dengan security. Ketika user bisa masuk kedalam server yang kita monitoring. Dalam pengujiannya cukup simpel dengan hanya kita masuk sebagai user yang salah dan melakukan edit terhadap suatu folder lalu kita check pada slack channel yang sudah kita atur untuk mendapatkan pesan dari kibana alertingnya. sebagai contoh penulis melakukan ssh menuju server yang sudah ditentukan tetapi dengan user yang tidak semestinya. User yang terdaftar pada server adalah "danur" tapi penulis mencoba login dengan menggunakan user "dnur" maka untuk alerting yang didapatkan akan seperti ini :



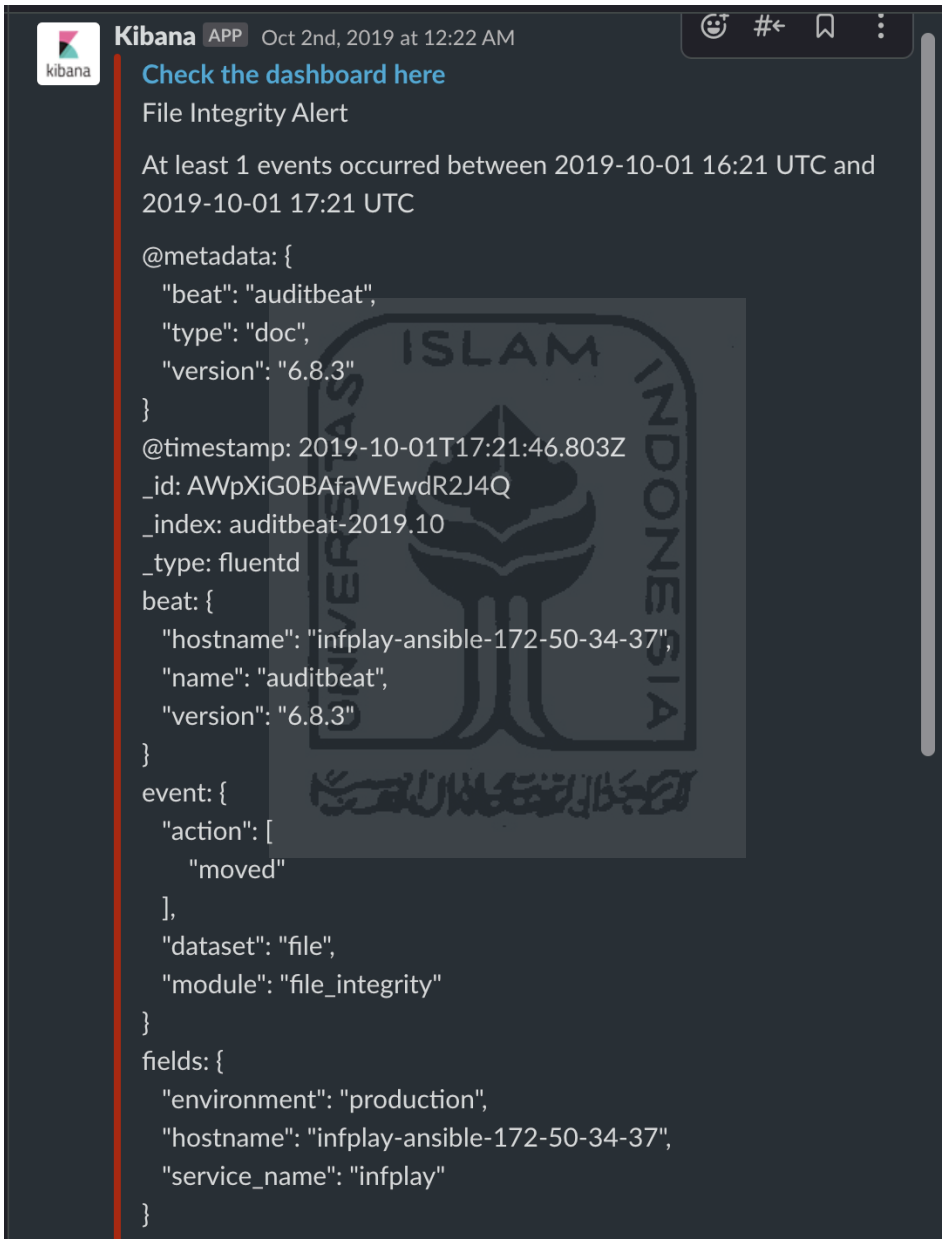
Gambar 4.26 Hasil alerting kibana

Pada gambar diatas ditu jukan bahwa ada seseorang dengan user danr ingin masuk ke server yang kita miliki dengan jumlah percobaan sekitar 4 kali percobaan. User mencoba memasuki server infplay dengan alamat ip tertera pada gambar tersebut. Selain itu pada *web dashboard* kibana juga dapat menampilkan hasil yang sesuai dengan menampilkan *user* yang dapat masuk ke server maupun yang tidak



Gambar 4.27 Visualisasi kibana

Untuk pengujian selanjutnya kita mencoba mengedit suatu file . disini penulis mencoba menambahkan spasi pada direktori /sshd, maka hasi yang akan keluar pada alertingnya adalah seperti ini:



Gambar 4.27 Hasil alerting auditd

BAB V
Kesimpulan dan Saran

a. Kesimpulan

Berdasarkan penelitian yang telah dilakukan oleh peneliti, kesimpulan sebagai berikut:

- Peneliti ini telah menghasilkan hasil implementasi penggunaan *EFK stack* dan dapat memonitoring user yang mencoba akses menuju ke sistem.
- Peneliti berhasil mendeteksi *sudo command* yang dijalankan oleh pengguna
- Peneliti juga telah mengimplementasi penggunaan auditd dan memadukannnya dengan menggunakan EFK stack, yang berfungsi sebagai agent untuk melakukan file integrity monitoring sebuah system yang ada

b. Saran

Berdasarkan kesimpulan yang telah dijabarkan peneliti, terdapat beberapa saran untuk penelitian selanjutnya, di antaranya adalah:

- Penggunaan EFK stack dapat diperbaharui dengan versi yang lebih baru, dengan adanya versi baru aka nada fitur yang ditawarkan
- Optimisasi kinerja elasalert

1. DAFTAR PUSTAKA

Bagnasco, S., Berzano, D., Guarise, A., Lusso, S., Masera, M., & Vallero, S. (2015). Monitoring of IaaS and scientific applications on the Cloud using the Elasticsearch ecosystem. *Journal of Physics: Conference Series*, *608*(1). https://doi.org/10.1088/1742-6596/608/1/012016

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

Kleindienst, P. (2016). *Building a real-world logging infrastructure with Logstash, Elasticsearch and Kibana*. 15. https://hdms.bsz-bw.de/files/5021/elk_paper_patrick_kleindienst.pdf

Lei, X., Wang, Z., & He, Y. (2015a). Log Real-time Management Scheme Based on LEK. *2nd International Workshop on Materials Engineering and Computer Sciences (IWMECS 2015)*, *Iwmecs*, 210–213. https://doi.org/10.2991/iwmecs-15.2015.38

Lei, X., Wang, Z., & He, Y. (2015b). The Data Management and Real-time Search Based on Elasticsearch. *4th International Conference on Computer, Mechatronics, Control and Electronic Engineering (ICCMCEE 2015)*, *Iccmcee*, 823–827. https://doi.org/10.2991/iccmcee-15.2015.149

Robles-Gómez, A., Ros, S., Martínez-Gámez, A., Hernández, R., Tobarra, L., Pastor, R., Caminero, A. C., & Cano, J. (2017). Using Kibana and ElasticSearch for the Recommendation of Job Offers to Students Antonio. *CEUR Workshop Proceedings*, *1925*, 93–99.

Vavliakis, K. N., Katsikopoulos, G., & Symeonidis, A. L. (2019). E-commerce Personalization with Elasticsearch. *Procedia Computer Science*, *151*(2018), 1128–1133. https://doi.org/10.1016/j.procs.2019.04.160

https://www.elastic.co/products/beats

https://www.elastic.co/products/beats/filebeat

ISLAM
UNIVERSITAS
INDONESIA

LAMPIRAN