BAB 2
DASAR TEORI

2.1 KAJIAN PUSTAKA

Penelitian Emeralda Dian Islami mengenai *"Analisa Penerapan Unified Threat Management (UTM) untuk Keamanan Layanan Database Pada Container Docker"* menjelaskan bahwa, Mengintegrasikan sebuah sistem keamanan UTM dengan layanan database pada docker adalah solusi yang dapat ditawarkan untuk membantu menyelesaikan masalah yang pengaksesan data secara illegal. *International Data Corporation* (IDC) mendefinisikan UTM sebagai sebuah produk yang menggabungkan berbagai fitur keamanan menjadi suatu *platform hardware* tunggal. UTM memiliki kemampuan untuk melakukan banyak kemanan dalam satu perangkat, beberapa fitur keamanannya seperti *firewall, gateway anti-virus* (AV), *gateway anti-spam*, VPN, *load balancing, content filtering dan reporting tools*. Produk UTM yang digunakan dalam penelitian Emeralda Dian Islami dan I Putu Hariyadi adalah *Endian* UTM Firewall. Layanan *database* pada *Docker* dibangun menggunakan MySQL. MySQL adalah sistem manajemen *database* yang memiliki sifat relational, yang berarti bahwa data yang dikelola di dalam *database* akan diletakkan pada beberapa tabel yang terpisah sehingga manipulasi data yang dilakukan dapat jauh lebih cepat[4].

Penelitian Rukmin Ulfa Ayu Lestari mengenai *"Analisis Penerapan Unified Threat Management (UTM) pada Docker untuk virtualisasi container"* menjelaskan bahwa, berdasarkan pengujian yang telah dilakukan dapat disimpulkan bahwa *Endian Firewall* UTM berhasil di implementasi dengan adanya pembuktian berdasarkan dropping paket ketika terjadi serangan SYN *Flood*. Selain itu, diperoleh perbandingan sebelum UTM diimplementasikan terjadi peningkatan jumlah paket SYN yang diterima dengan selisih rata-rata sebesar 3,06% dengan nilai paket 64,48 dalam MB. Dengan rincian pada layanan *web server* peningkatannya 2,23% dan 20,53 dalam Mb, pada layanan PhpMyAdmin peningkatan 0,17% dengan 24,63 dalam Mb, dan pada layanan MySQL peningkatannya 0,66% dengan nilai 19,32 dalam Mb. Sedangkan keadaan setelah UTM diaktifkan berbanding terbalik dengan terjadinya penurunan jumlah paket

SYN yang diterima dengan selisih rata-rata sebesar 3,85% dengan nilai 39,74 dalam Mb. Dimana rincian pada layanan *web server* terjadi penurunan sebesar 1,38% dengan nilai 12,28 dalam Mb, pada layanan MySQL penurunannya sebesar 0,55% dengan nilai 10,85 dalam Mb, dan pada layanan PhpMyAdmin terjadi penurunan sebesar 1,65% dengan nilai 16,61 dalam Mb[5].

Penelitian Bambang Heru, Benny, Defendy dan Wahyu Hento mengenai *"Keamanan Jaringan Menggunakan Unified Threat Management Pada Server Berbasiskan Linux"* menjelaskan bahwa, berdasarkan penelitian yang telah dilakukan dapat ditarik sebuah simpulan yaitu, suatu system komputer yang terhubung jaringan dengan memiliki kepentingan khusus, harus lebih memperhatikan keamanan datanya. Aplikasi keamanan jaringan yang ada saat ini sangatlah beragam, hal itu dapat merumitkan dalam hal pengaturan aplikasi dan dibutuhkan staf yang ahli dalam bidangnya. Dengan adanya UTM, kerumitan dalam hal pengaturan aplikasi keamanan akan berkurang karena untuk mengatur semuanya dapat dipusatkan menjadi satu kesatuan dan UTM dapat diatur melalui program administrasi berbasiskan web, sehingga administrator jaringan dapat mengaksesnya dimanapun dan kapanpun[1].

Berdasarkan penelitian Gigin Anggriawan mengenai *"Analisis Dan Perancangan Prototype Sistem Keamanan Jaringan Menggunakan Unified Threat Management (UTM) Firewall Berbasis Linux di DPRD Provinsi Jawa Barat"* menjelaskan bahwa, berdasarkan hasil pengamatan inventarisasi terdapat jenis ancaman serangan pada jaringan komputer di DRPD Provinsi Jawa Barat yaitu *DDos, deface, illegal access server, spam, virus, malware, content illegal* dan *Spoofing*. Salah satu sistem keamanan untuk mengatasi ancaman serangan tersebut yaitu dengan UTM *firewall*. Metode yang digunakan pada penelitian ini yaitu PPDIOO merupakan singkatan dari *Prepare, Plan, Design, Implement, Operate,* dan *Optimize*. *Cisco Lifecycle Services* adalah pendekatan dengan enam fase. Pada penelitian ini menekankan pada analisis data Â– data jaringan komputer dan ancaman penyerangan serta melakukan pengujian penyerangan pada topology komputer yang berjalan dengan menggunakan sistem keamanan jaringan UTM *firewall* berbasis *Linux*. Berdasarkan pengujian dengan metode *blackbox* yang telah dilakukan, maka dapat disimpulkan bahwa dengan diterapkannya UTM *firewall* di

DPRD Provinsi Jawa Barat mengatasi ancaman serangan seperti *DDos, deface web, access illegal database, virus, malware, spam* dan *spoofing*. Dalam mengatasi acamana tersebut diterapkanya kombinasi antara sistem keamanan seperti *firewall, Intrusion Prevent System* (IPS), *anti-virus, anti-spam, content web filtering dan notification email* dari masing- masing ancaman serangan[2].

Berdasarkan penelitian Felix, Olyvia Gunawan dan Andi Juanda mengenai *"Analisis Dan Implementasi Sistem Keamanan Jaringan Dengan Menggunakan PC Linux Sebagai UTM (Unified Threat Management) Pada PT. Duta Palma Nusantara"* menjelaskan bahwa, UTM merupakan suatu produk yang menggabungkan dan mengintegrasikan beberapa fitur keamanan menjadi suatu system keamanan jaringan terpadu. Fitur keamanan tersebut meliputi : *Firewall, Web Proxy, Intrsion Detection System,* dan *Mail Filtering.* Tujuan dari penulisan ini adalah mengimplementasikan aplikasi system keamanan jaringan UTM pada PT. Duta Palma Nusantara. Metode yang digunakan adalah metode analisis dan metode implementasi. Setelah dilakukan implementasi UTM pada PT. Duta Palma Nusantara, beberapa evaluasi dilaukan untuk menguji keefektifan UTM. Dari hasil evaluasi, terbukti bahwa serangan virus yang berbahaya terhadap jaringan internal perusahaan menurun. UTM dapat membatasi penggunaan bandwith pada tiap user, sehingga koneksi internet tetap lancar pada jam sibuk. Dengan mengaktifkan *web proxy* dan *content filtering,* situs yang berbahaya tidak dapat diakses. Kesimpulan yang diperoleh dengan adanya UTM adalah serangan virus terhadap jaringan internal menurun, koneksi internet tidak lagi terputus pada jam sibuk kantor, serta pengaksesan situs web yang dapat membahayakan jaringan internal perusahaan dapat dikendalikan[3].

2.2 SISTEM KONTAINERISASI DENGAN DOCKER

2.2.1 Sistem Kontainerisasi

Pada dasarnya ketika seorang developer sedang mengembangkan sebuah perangkat lunak, terkadang developer tersebut sering menemui sebuah kasus dimana *environment* yang digunakan mempunyai *resource* yang tidak memadai, oleh karena itu *environmment* tersebut sering mengalami *down* sehingga tidak bisa melakukan pekerjaanya dengan baik. Selain itu dalam dunia teknologi IT seperti sistem komputasi jaringan maupun server, sering mengenal istilah teknologi

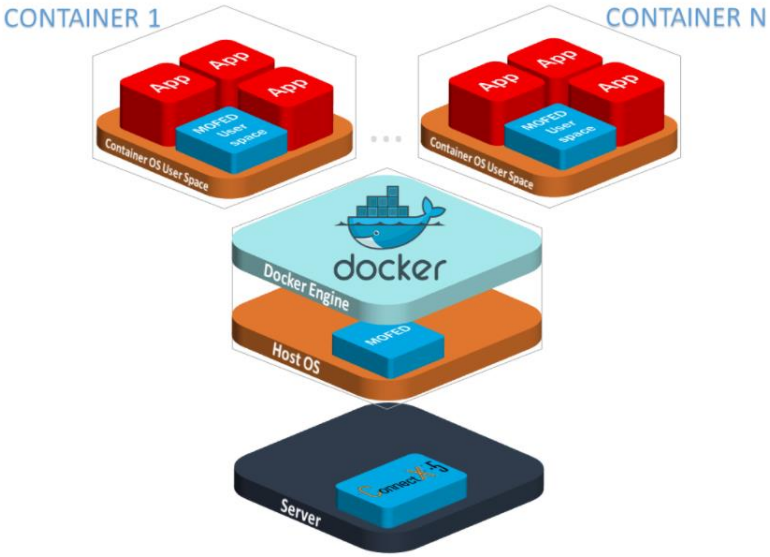
virtualisasi, teknologi virtualisasi sendiri, merupakan sebuah teknologi yang sangat membantu dalam menghemat penggunaan *resource* dari sebuah PC atau sever, salah satunya adalah dengan membuat sebuah *virtual machine*

Gambar 2.1 perbedaan sistem kontainerisasi dengan *Virtual Machine*[12]

Dengan menggunakan *virtual machine* seorang developer dapat membuat beberapa *guest OS* dalam satu fisik server. Namun penggunaan *virtual machine* tersebut membutuhkan *resource* yang cukup banyak dalam satu fisik server tersebut karena seperti pada gambar 2.1 diatas *virtual machine* berjalan diatas *guest OS,* hal tersebut yang membuat *virtual machine* jadi membutuhkan *resource* yang lebih banyak. oleh karena itu diciptakanlah sebuah sistem dengan *resource* yang lebih ringan dibandingkan dengan *virtual machine,* yaitu adalah sistem kontainerisasi. Virtualisasi dilakukan dengan menggunakan sebuah kontainer, dimana didalam kontainer tersebut hanya berisi aplikasi atau program yang dibutuhkan, dan dijalankan dengan menggunakan sebuah *engine* yang dihubungkan dengan sebuah OS pada server tersebut gambaran mengenai sistem kontainerisasi dapat dilihat pada gambar 2.1. Sistem kontainerisasi ini berjalan secara terisolasi jadi tidak akan mempengaruhi kinerja dari sistem lain meski dalam jumlah yang banyak dan antara kontainer satu dengan yang lain dapat saling terhubung dan berkomunikasi[11].

2.2.2 *Docker Container*

Docker Container merupakan sebuah aplikasi yang bersifat open source, berfungsi sebagai wadah atau tempat untuk kontainer yang bersifat mengikat dan sebagai kontroller dari kontainer tersebut, *Docker Container* dapat mengendalikan dan memanajemen kontainer yang berjalan diatas *Docker engine.*



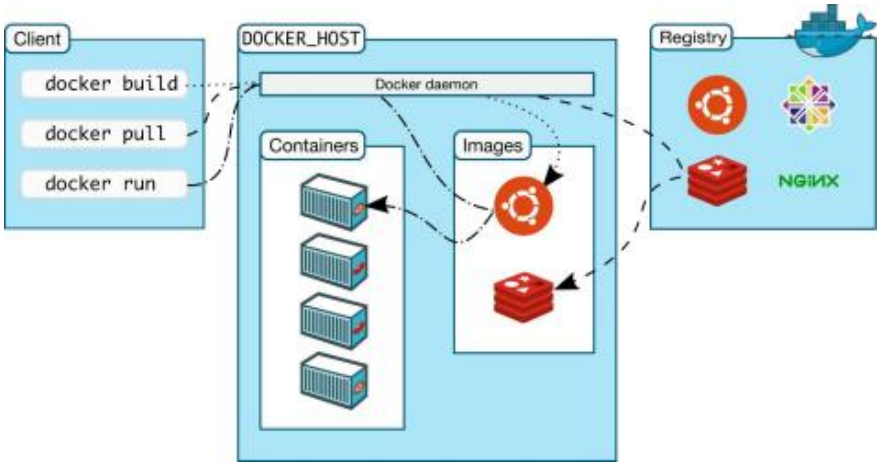
Gambar 2.2 Konsep *Docker Container*[11]

Berdasarkan gambar 2.2, pada dasarnya *Docker Container* bekerja dengan memvirtualisasikan sistem dengan menenempatkan masing-masing sistem didalam sebuah *container* yang bersifat terisolasi. Didalam *container* tersebut hanya terdapat aplikasi yang dibutuhkan tanpa menggunakan *guest OS*, jadi penggunaan *Docker Container* ini sangat efektif untuk menghemat sumber daya. beberapa keuntungan dari penggunaan *Docker Container* seperti berikut :

1. Penggunaan resource atau sumber daya kecil
2. Terisolasi dan dilakukan secara langsung oleh sistem operasi utama tanpa harus mengandalkan guest OS
3. Manajemen *deploy* lebih rapi, karena menggunakan konsep *Docker image*.
4. *Images* memungkinkan dapat terbuatnya sebuah *node* kontainer dengan jumlah banyak dan dengan cara yang mudah[6].

2.2.3 Arsitektur *Docker*

Docker Container memiliki arsitektur yang berbeda dengan teknologi *hypervisor,* perbedaan tersebut sudah cukup jelas bahwa pada *Docker* sendiri menggunakan sistem kontainerisasi dalam memvirtualisasikan sebuah sistem. Pada gambar 2.3 arsitektur dari *Docker Container* terdiri dari beberapa elemen penting seperti *Docker client, Docker daemon, Docker Container, Docker images* dan *Docker registry.*

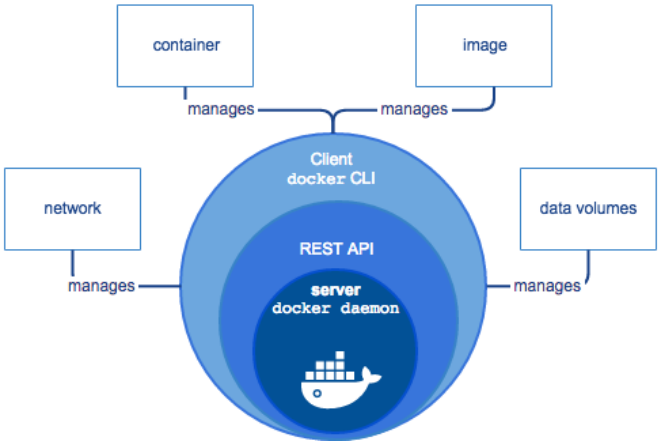


Gambar 2.3 Arsitektur *Docker Container*[6]

Docker menggunakan teknologi *client server* untuk menghubungkan antara *Docker client* dengan *Docker daemon.* Berdasarkan gambar 2.3 diatas diketahui bahwa, *Docker client* bertugas untuk mengirimkan perintah kepada *Docker host* dan *Docker daemon* akan melakukan request kepada *Docker registry* untuk meminta *file images* yang akan digunakan untuk membangun sebuah kontainer. *Docker Client* dan *Server* dapat berjalan pada sistem yang sama, antar *Docker Client* dan *Docker Server* dapat berkomunikasi *via socket* menggunakan *Restful API or CLI*[6].

2.2.4 *Docker Daemon*

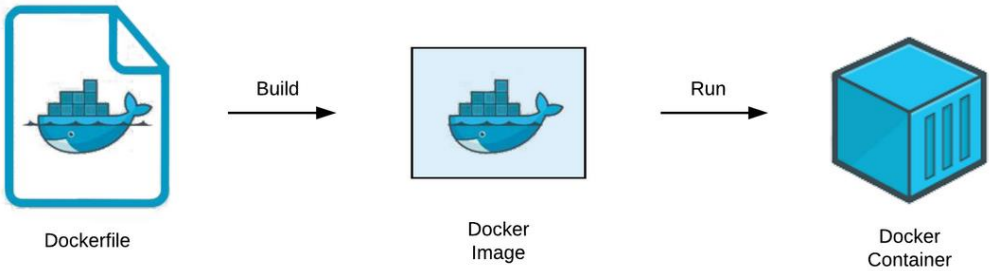
Docker daemon berfungsi untuk membangun, mendistribusikan dan menjalankan kontainer. Pengguna tidak dapat langsung menggunakan *Docker daemon*, akan tetapi untuk menggunakan *Docker daemon* maka pengguna harus menggunakan *Docker client* sebagai perantaranya. Seperti pada gambar 2.4 *Docker Daemon* berkomunikasi dengan mengekspos *REST API* dengan cara memasukan *command* melalui *Command Line Interface* (CLI) [6].



Gambar 2.4 *Docker Daemon*[6].

2.2.5 *Docker Images*

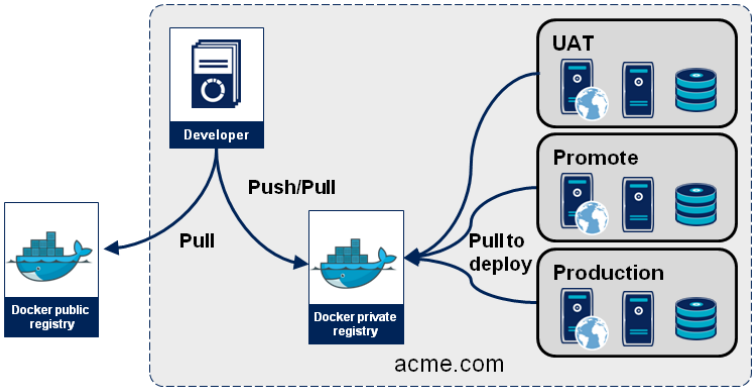
Docker Images merupakan sebuah template yang bersifat read-only, di dalamnya akan terdapat sistem operasi beserta aplikasi web , database dan segala library terkait. Image dasar dapat dibuat sendiri dengan menambahkan aplikasi yang dibutuhkan (commiting the change) dan dapat pula hasilnya kita simpan ke repository yang tersedia (push to repository), atau dapat mengambil yang telah tersedia (pull from repository). Pada gambar 2.5 dapat dilihat bahwa alur untuk membangun sebuah *container* adalah dengan membuat sebuah *Docker File,* file ini bisa berupa skrip yang didalamnya sudah terisi *images* yang akan dipakai dan aplikasi yang akan dijalankan, setelah itu barulah ketika *Docker File* tersebut dieksekusi, maka ia akan melakukan *build images*. *Template* ini sebenarnya adalah sebuah OS yang telah di install berbagai aplikasi didalamnya, *Docker images* berfungsi untuk membuat sebuah kontainer. Dengan menggunakan 1 *images* dapat digunakan untuk membuat banyak kontainer sekaligus[13]



Gambar 2.5 *Docker Images*[13]

2.2.6 *Docker Registry*

Didalam penggunaan *Docker Container*, ketika seorang *sytem administrator* akan membuat sebuah *container*, maka ia harus mempunyai *image file* yang akan digunakan. *Image file* dapat ditemukan di *Docker Registry*. *Docker registry* adalah kumpulan *Docker images* yang bersifat private maupun public yang dapat diakses oleh pengguna *Docker*. Dengan menggunakan *Docker registry public*, maka pengguna *Docker* dapat menggunakan *images* yang dibuat oleh *developer* lain. Pada dasarnya *Docker Registry* bertugas untuk mendistribusikan *images* ke penyimpanan lokal maupun publik seperti pada gambar 2.6, setiap *container* dapat melakukan *pull* dan *deploy* dari mana saja. salah satu *Docker registry* yang paling banyak diakses adalah *Docker Hub*[12].



Gambar 2.6 Konsep *Docker Registry*[12]

2.3 UNIFIED THREAT MANAGEMENT

Unified Threat Management merupakan sebuah sistem keamanan yang mengkombinasikan beberapa fitur kemanan dengan manajemen terpusat. Fitur keamanan yang dimiliki oleh sebuah UTM bisa berupa VPN, Firewall, IPS, *antrivirus gateway*, antispam, *content filtering*, dan *bandwith management.* Seluruh fitur tersebut dikonfigurasikan dengan console terpusat. Dengan menggunakan satu UTM seorang *network engineer* dapat mendapatkan sejumlah fitur keamanan yang lengkap disertai dengan tingkat kemudahan dalam pengoperasianya. UTM dapat berupa *software* dan *hardware*, beberapa vendor yang mengeluarkan *hardware* UTM seperti sophos, cyberoam, dan netgear. Sedangkan untuk *software appliance* vendor yang mengeluarkan seperti *Endian*, untangle dan ipcop. Sistem UTM sering kali menyertakan teknologi atau fitur keamanan jaringan sebagai berikut :

1. Layanan antispam, untuk memblokir atau menandai serangan berbasis email yang masuk dengan memindai lalu lintas *Simple Mail Transfer Protocol* (SMTP) yang masuk dan keluar. Pemfilteran antispam memungkinkan bisnis untuk menggunakan daftar blokir spam berbasis server pihak ketifa atau untuk membuat daftar putih dan hitam lokal untuk memfilter pesan email. Beberapa sistem UTM meminda ancaman keamanan jaringan lain yang dibawa dalam lalu lintas aplikasi, seperti layanan pesan instan yang digunakan peretas untuk menyebarkan malware.
2. Didalam perangkat UTM terdapat kontrol aplikasi yang digunakan untuk memasukan layanan tertentu kedalam daftar putih dan memindai layanan mana yang boleh dan tidak boleh untuk digunakan. Kontrol aplikasi penting untuk keamanan jaringan karena banyaknya layanan dan aplikasi yang rentan terhadap serangan dari pihak lain.
3. Firewall merupakan fungsi keamanan jaringan tertua dan paling mendasar. Salah satu fitur dari UTM adalah terdapat firewall didalamnya, firewall membatasi pembentukan koneksi jaringan antara host di dalam maupun diluar jaringan lokal dengan tujuan menghilangkan eksposur ke host external.
4. fitur *Intrusion detection* dan *intrusion prevention sytem,* fitur ini melakukan pencegahan dan indentifikasi pada serangan IDS dan IPS dengan mendeteksi saat penyerang mencoba mengakses jaringan dan mencegah jenis serangan tersebut terjadi. Perangkat dan layanan UTM yang paling efektif mengatasi jenis ancaman keamanan ini melalui kombinasi metode, termasuk mendeteksi serangan berdasarkan *signature malware,* anomali atau deteksi berbasis reputasi untuk mengehentikan serangan yang tidak dikenal.
5. Fitur *Virtual Private Network* (VPN). UTM mendukung layanan VPN, meskipun sebagaian besar fungsi keamanan jaringan UTM dimaksudkan untuk mendeteksi dan menghentikan serangan, VPN ini dirancang khusus untuk melindungi aktivitas jaringan lokal dari manipulasi atau penyadapan, VPN menyediakan jaringan privat yang terlindungi dan aman untuk diakses kapanpun. VPN dapat dikonfigurasi untuk menyalurkan semua lalu lintas dari host seluler ke perangkat UTM dan memungkinkan semua pemeriksaan

keamanan jaringan UTM diterapkan ke lalu lintas seluler dan mengurangi jumlah serangan yang melibatkan perangkat seluler.

6. Fitur *web filtering.* Fitur ini digunakan untuk melakukan filtering terhadap konten atau URL yang mencakup berbagai teknik yang menentukan apakah permintaan web yang melibatkan situs web atau URL harus diizinkan atau tidak. Beberapa UTM menggunakan teknik analitik yang dapat memindai situs web untuk pelanggaran keamanan yang menunjukan bahwa situs web dapat menimbulkan ancaman keamanan[7].

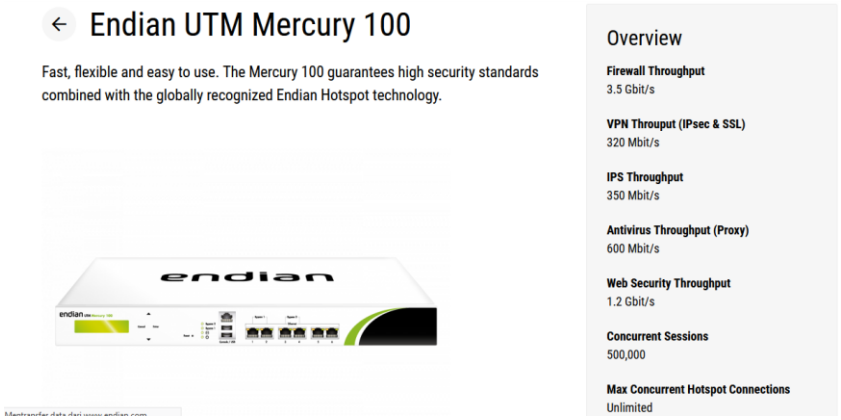
2.3.1 *ENDIAN FIREWALL*

Endian Firewall merupakan salah satu UTM yang dibangun berbasis linux, yang berguna untuk menjadi sbuah *firewall, Router* dan seluruh aplikasi untuk keperluan *Network Security.* Project ini pertama kali dikembangkan di Italy dengan mengadopsi distro IPCop. Namun seiring dengan perkembanganya, akhirnya hanya setengah saja yang masih mengadopsi dari IPCop, selebihnya merupakan pengembangan yang dilakukan secara mandiri. Gambar 2.8 *Endian firewall* memiliki fungsi *networking* seperti *firewall, Router, load balancing, NAT, VPN, DHCP Server, content filtering* dan masih banyak lagi.

Gambar 2.7 *Dashboard Endian Firewall*[14]

UTM ini sangatlah cocok untuk *network engineer* yang ingin memiliki server atau *appliance* dengan fitur *networking* yang lengkap. Ada beberapa versi yang di keluarkan oleh *Endian*:

1. *Endian Firewall Community* (EFW): Versi *Opensource* dari *Endian*. Berbentuk ISO file dan bisa diunduh gratis dari website *Endian* Community nya. Versi ini bisa langsung di install di PC/Server. Dikarenakan versi gratisan, maka fitur-fitur yang didapat terbatas dan tidak sebanyak yang versi Professional nya.
2. *Endian* UTM *Professional*: Versi *commercial* dari *Endian*. Pada versi ini memiliki fitur yang lebih lengkap dibanding versi *Community*. *Endian* sendiri menyediakan beberapa jenis dari *Endian* UTM Pro.
   a) *Virtual Appliance*: *Endian* berbentuk *Appliance* namun di *hosted* oleh *Endian* nya. Bisa dibilang seperti VPS tapi berisi *Endian Firewall*.
   b) *Software Appliance*: *Endian* berbentuk ISO. Versi nya hampir mirip dengan yang yang versi *Community*, hanya saja fitur-fitur nya diperbanyak.
   c) *Hardware Appliance*: *Endian* berbentuk *Appliance* atau *hardware* seperti pada gambar 2.9. Jadi kita membeli *Endian* beserta *hardware* nya. *Hardware* nya mirip-mirip seperti Server berukuran 1U. Sehingga memudahkan kita untuk di *attach di Rack Server*[14].

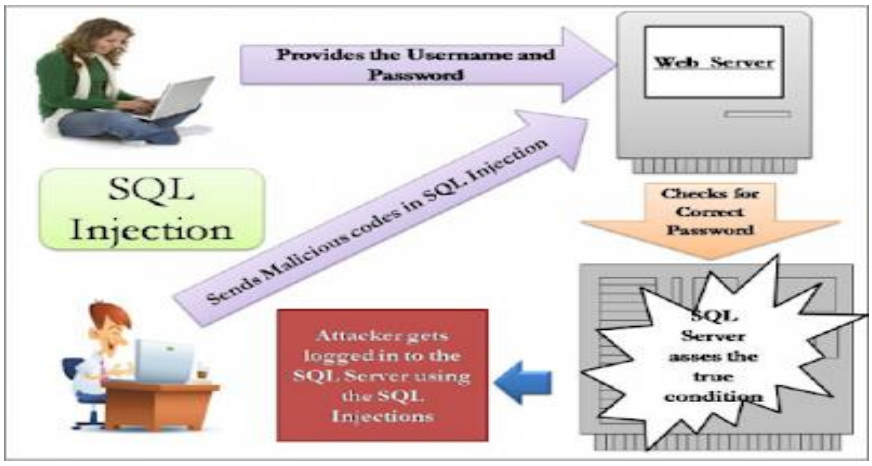


Gambar 2.8 salah satu contoh *Hardware* UTM *Endian*[14]

2.4 SQL INJECTION

Structured Query Language (SQL) merupakan sebuah bahasa yang digunakan untuk melakukan query, mengoperasikan dan mengelola *database system.* Penggunaan SQL pada ummnya disemua sistem database sama saja namun

ada beberapa perbedaan khusus untuk setiap database tersebut. Dalam penggunaan SQL *database* sering kali terjadi serangan yang mengakibatkan rusak dan hilangnya data yang terdapat pada SQL *database*, serangan pada tersebut adalah berupa SQL *Injection.* Serangan ini memanfaatkan celah keamanan pada basis data sebuah aplikasi, jadi ketika *user* melakukan inputan namun oleh sistem tersebut karakter inputan user tidak di *share* secara benar yang mana akan menjadi celah dari SQL itu sendiri.



Gambar 2.9 Konsep SQL *Injection*[8]

Konsep atau cara kerja dari *SQL Injection* dapat dilihat pada gambar 2.10. dimana dalam gambar tersebut dapat dijelaskan bahwa setiap aplikasi web yang menyimpan data pengguna pasti memerlukan autentikasi untuk mengaksesnya, contoh sederhananya adalah dengan melakukan login dengan menggunakan akun dan password. Proses ini sering diterapkan untuk meningkatkan keamanan. Namun dengan *SQL Injection* sistem ini dapat dirusak dengan mudah. Misalnya terdapat pengguna salah satu aplikasi dengan username “joko” dan password “satudua”. Jika pengguna melakukan input sesuai data yang tersimpan pada database maka tidak akan terjadi masalah pada sistemnya, namun apabila pengguna membuat username menjadi “joko #” maka karakter “#” akan menyebabkan karakter selanjutnya tidak dianggap sebagai kode SQL. Akibatnya, username “joko” dapat melakukan input data tanpa harus mengetahui passwordnya.

Dalam dunia IT terutama pada bidang *network security* penggunaan SQL *Injection* masih diperbolehkan, namun dengan rasa tanggung jawab terhadap

penggunaan dari aplikasi tersebut, berikut merupakan fitur yang terdapat pada SQL *Injection* :

1. Bypass otentikasi

   Teknik ini memungkinkan seorang *attacker* untuk melakukan bypass login atau masuk kedalam sistem dengan hak administratif tanpa harus mempunyai akun username dan password.
2. Pengumpulan informasi

   Teknik ini memungkinkan seorang *attacker* untuk mendapatkan informasi sesitif pada *database*, seperti informasi pribadi client, username, password dan data lainya.
3. *Compromised* integritas data

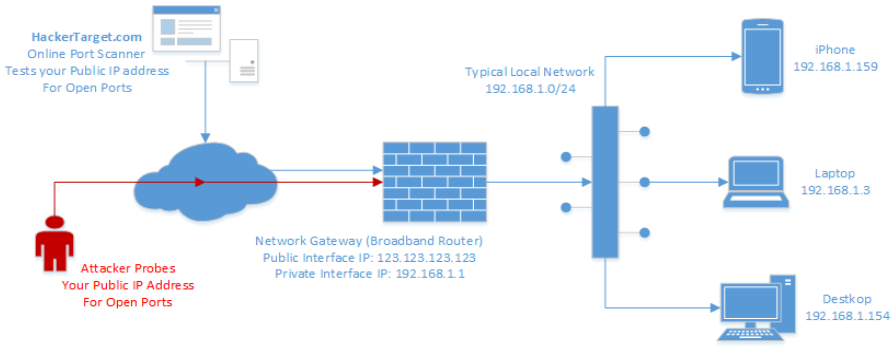
   Attacker dapat melakukan perubahan *database*, dengan teknik ini *attacker* dapat melakukan deface ataupun memasukan konten berbahaya didalam suatu *database.*
4. Compromised ketersediaan data

   Teknik ini memungkinkan *attacker* untuk menghapus informasi pada *database* yang bertujuan untuk merusak log atau audit informasi pada *database.*
5. *Remote command execution*

   Memungkinkan *attacker* untuk melakukan perintah eksekusi melalui *database* untuk dilakukan pada sistem operasi server[8].

2.5 PORT SCANNING

Port merupakan sebuah jalur atau pintu dimana seluruh data dapat berjalan melewatinya agar informasi dapat tersampaikan kepada *client.* Ketika perangkat terhubung dengan jaringan internet, perangkat tersebut akan menetapkan nomor port TCP atau UDP dari 0 hingga 65535. Namun, beberapa port yang sering digunakan adalah misalnya port 20, yang digunakan untuk layanan *File Transfer Protocol* (FTP), selanjutnya ada port 22 yang digunakan untuk melakukan koneksi terhadap *Secure Shell* (SSH), dan port 80 yang digunakan pada layanan *Hypertext Transfer Protocol* (HTTP). Selain itu terdapat port yang mempunyai standart keamananya sendiri seperti port SSH dan HTTPS.



Gambar 2.10 Konsep *Port Scanning*[15]

Port scanning merupakan proses memeriksa semua port pada alamat IP tertentu untuk melihat apakah port tersebut terbuka atau tertutup. Pada gambar 2.11 Perangkat lunak pemindaian port akan memeriksa seluruh port yang terhubung dengan jaringan internet dan semua jalan sampai ke port 65535. Firewall dapat memblokir atau menurunkan lalu lintas data, sehingga pemindaian port dapat dicegah agar semua orang tidak dapat melakukan pemindaian port secara random. Alat yang digunakan untuk melakukan port scanning salah satunya dapat menggunakan Nmap. Tujuan dari dilakukanya *port scanning* adalah untuk mengetahui port mana saja yang terbuka pada sebuah sistem dan dapat digunakan untuk membantu mencegah terjadinya suatu serangan[9].

2.6 SYN FLOOD

Pada serangan SYN flood, pesan sinkronisasi (SYN) diterima di mesin host untuk memulai dengan "jabat tangan". Permintaan ini diakui oleh server dengan mengirimkan tanda pengesahan (ACK) ke host awal dan menunggu koneksi ditutup. Koneksi akan selesai ketika mesin yang meminta akan menutup koneksi. Secara umum, proses permintaan SYN adalah :

1. Client meminta SYN yang berarti *synchronize* ke *server*.
2. Server memberi jawabab SYN-ACK ke *client.*
3. Client menjawab ACK dan koneksi dibuat.

Dalam serangan SYN flood, permintaan palsu dikirim dan server merespon dengan paket ACK untuk menyelesaikan koneksi TCP tetapi sambungan diarahkan kee timeout, daripada menutupnya. Oleh karena itu, sumber daya server menjadi lelah dan server pun akhirnya offline[16].