BAB 2
DASAR TEORI

2.1 KAJIAN PUSTAKA

Penelitian Dedy Panji Agustino, Yohanes Priyoatmojo, Ni Wayan Wiwin Safitri pada tahun 2017, membahas tentang upaya peningkatan keamanan pada server dengan menggunakan layanan Honeypot yang pada implementasinya pada *Cloud Computing*. Layanan Honeypot bertujuan pada *cloud computing* bertujuan untuk melindungi server dari serangan seperti *brute force* dan *malware*. Pada penelitian tersebut, menggunakan aplikasi Honeypot yaitu Kippo untuk mendeteksi serangan *brute force* dan untuk mendeteksi *malware* dengan menggunakan *Dionaea*. Hasil penelitian tersebut adalah sistem dengan menggunakan Honeypot dapat melindungi layanan *Cloud Computing* dari serangan *brute force* dan *malware*. Penelitian tersebut mengkonfigurasi *port* dan sistem palsu pada server yang menyerupai layanan server aslinya. Kedua serangan tersebut dapat dideteksi dengan baik oleh Honeypot Cowrie dan Dionaea [5].

Penelitian Abdul Muhaimi, I Putu Hariyadi, Akbar Juliansyah pada Desember 2019, pada penelitiannya membahas tentang Pengamanan Server yang Terintegrasi dengan Telegram. Metode yang digunakan peneliti menerapakan *Intrusion Prevention System* (IPS) yang fungsinya untuk mengidentifikasi dan memblokir ancaman yang mengancam ke jaringan. Sistem IPS yang peneliti kembangkan diintegrasikan dengan Telegram sehingga administrator dapat memperoleh pemberitahuan ketika ada orang lain melakukan penyerangan terhadap server. Pemblokiran dengan menggunakan IPtables [6].

Thesis Davide Bove pada 30 Oktober 2018 membahas tentang penggunaan Honeypot untuk mendeteksi dan menganalisa infrastruktur *cloud*. Honeypot memberikan fungsi *monitor* penyerangan dan berperan seakan menjadi mesin yang sesungguhnya, penyerang seolah sedang melakukan penyerangan terhadap server sesungguhnya. Pada penelitiannya selama 2 bulan menghasilkan 170 juta *log entries*. Hasil analisa menunjukan penyerangan yang terjadi berasal dari China, USA dan Russia dan penyerangan menargetkan VNC dan layanan SSH.

Penyerangan tersebut terjadi secara otomatis dan berulang terus-menerus. Banyak dari *malware* disebarkan selama penyerangan dapat mudah dideteksi dengan menggunakan anti-virus yang *up-to-date*. Metode koleksi data menghasilkan database yang dapat dianalisa secara mendalam [7].

Penelitian Dias Utomo, Muchammad Sholeh, Arry Avorizano, membahas tentang *monitoring* server. *Network monitoring* adalah proses server yang secara rutin mengoleksi dan menghitung proses data pada sebuah jaringan yang dapat melihat perubahan yang terjadi pada server sehingga dapat memperbaiki sistem jaringan dengan mengetahui secara dini adanya kelainan pada server. Banyak perusahaan berkembang yang mempunyai masalah dengan server, terutama masalah pada pengawasan server dimana jika server tidak di-*monitor* dengan khusus dapat menyebabkan kerusakan yang termasuk fatal jika terjadi serangan yang tidak terdeteksi [8].

Pada penelitian Armin Ziaie Tabari dan Xinming Ou membahas Langkah pertama memahami serangan *cyber* pada perangkat IoT. Honeypot untuk mengumpukan data serangan penyerangan secara real-time. *Low interaction* honeypot digunakan pada perangkat IoT cameras atau Honeycamera. Hasil penelitian menunjukan bahwa perangkat Honeycamera mempunyai banyak celah untuk diserang dan penggunaan *low interaction* honeypot cowrie dapat meningkatkan keamanan dan merekam data penyerang [9].

Pada penelitian Jefree Fahana, Rusydi Umar, Faizin Ridho, penelitiannya membahas pemanfaatan Telegram. Metode yang digunakan peneliti yaitu dengan memanfaatkan *log system* yang mencatat semua proses pada server. Data *log* berfungsi untuk menganalisa proses yang terjadi pada server sehingga jika terjadi *anomaly* pada server maka administrator dapat menganalisa *log system* untuk menemukan *anomaly* tersebut. Namun, menganalisa *log* saja tidak cukup, peneliti perlu untuk membangun *system* yang dapat mendeteksi serangan dan memberi notifikasi informasi kepada administrator apabila terjadi serangan pada *system* server, peneliti menggunakan aplikasi Telegram, kemudahan akses Telegram pada semua platform memberikan kemudahan bagi *user* terutama admistrator untuk memanfaatkan *system* pemberitahuan jika terjadi serangan *system* dengan memanfaatkan *Application Programming Interface (API)* [10].

2.2 DASAR TEORI

2.2.1 SERVER

Komputer server adalah sistem komputer yang menyediakan 1 atau lebih jenis layanan tertentu yang diaplikasikan dalam jaringan komputer, contoh: *SSH server, web server, mail server, proxy server, telnet server*. Server mempunyai perangkat lunak administratif yang dapat mengatur/membatasi akses terhadap suatu jaringan , paket data dan mengatur sumber daya yang ada pada server. Server berjalan pada arsistektur *client-server*. *Server* adalah program yang melayani permintaan dari program lain oleh *client*. *Client* adalah pengguna yang menjalankan *software*, *hardware* dan memungkinkan berbagi data dan informasi [11].

2.2.2 TCP

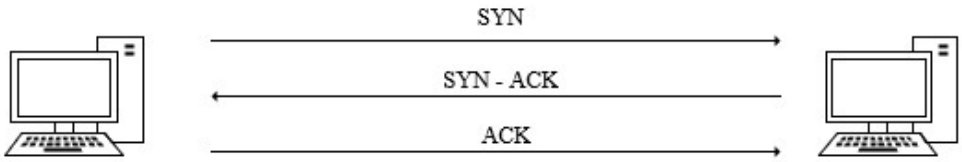
Protokol TCP/IP adalah suatu standar komunikasi antar perangkat jaringan. Contoh protokol TCP adalah http, *e-mail*, ftp, dan SSH. TCP bekerja dengan 2 pengelompokkan kategori *port*, yaitu:

1. *Low Port* (standar layanan *port*). *Port* ini berada pada rentang angka *port* 1 – *port* 1024
2. *High Port* (*port* untuk transmisi lanjutan). *Port* ini digunakan untuk layanan TCP yang bersifat *custom*. *Port* ini berada pada rentang angka *port* 1025 – *port* 65536.

Protokol TCP memiliki beberapa prinsip kerja, yaitu:

1. *Connection Oriented*

   Sebelum data ditransmisikan antara 2 *host*/perangkat jaringan pada *application layer* harus melakukan sesi komunikasi antar perangkat pada protokol TCP atau disebut jugan dengan proses “Three-way Handshake”, hal ini bertujuan untuk memastikan sinkronasi antar paket pada saat komunikasi antara perangkat satu dan perangkat lainnya. Paket mencocokkan nomor urut dan nomor *acknowledgement* dari perangkat yang berkomunikasi.



Gambar 2.1 *Three-way Handshake*

A. *Client* : *SYN -> Server* : *Client* akan mengirimkan SYN ke *server.*
B. *Server* : *SYN-ACK -> Client* : *Server* merespon SYN *Client* dengan mengirimkan *SYN-ACK* ke *Client*.
C. *Client* : *ACK -> Server* : Setelah menerima *SYN-ACK* dari *server*, *client* mengirim *ACK* ke *Server*.

Setelah melewati proses komunikasi *Three-way Handshake,* proses komunikasi dapat terbentuk. Proses ini akan memastikan kedua *host* berkomunikasi dengan transmisi data dan semua data diterima dengan baik dan lengkap. Koneksi *TCP* diakhiri dengan proses *FIN* (*TCP connection termination*).

2. *Reliable Transmission*

Data yang ditransmisikan menggunakan koneksi TCP akan diurutkan dengan nomor pada setiap *byte* data agar paket data dapat disusun kembali saat data diterima. Paket data yang diterima akan diurutkan dan paket data yang duplikat akan diabaikan.

3. *Error Detection*

Jika ada data *error* pada saat proses transmisi data, koneksi TCP dapat memperbaiki dengan melakukan pengiriman data yang *error*/hilang. TCP menggunakan perhitungan *TCP Checksum.*

4. *Flow Control*

Flow Control yaitu proses deteksi yang bertujuan supaya satu *host* tidak mengirimkan data ke *host* lainnya dengan terlalu cepat. *Flow control* mengatur agar *device* yang berbeda-beda tetap dapat

berkomunikasi dengan mengatur agar aliran data pada perangkat tidak kewalahan.

5. *Segment size control*

   Segment size control akan mendeteksi besaran MSS (*maximum segment size*) yang dapat dikirim untuk mencegah IP *fragmentation*. MSS adalah informasi ukuran data terbesar yang dapat ditransmisikan oleh TCP dalam bentuk *segment* tunggal. Fragmentasi IP mengakibatkan paket yang hilang dan proses retransmisi yang berlebihan.

6. *Congestion Control*

   Prinsip kerja TCP *Congestion Control* yaitu mengatur aliran data yang masuk ke suatu *network*[12].

2.2.3 SSH

SSH adalah protokol administrasi jaringan yang memungkinkan *user* untuk mengakses server secara *remote shell* pada sebuah jaringan. SSH memiliki kemampuan enkripsi dan dekripsi pada e*nd to end device* yang membuat *SSH* memiliki keamanan yang baik.

2.2.4 SSH *SERVER* DAN SSH *CLIENT*

SSH *server* adalah suatu perangkat lunak yang menyediakan fungsi menerima r*equest packet* data dari *client*. SSH server juga dapat menenkripsi dan dekripsi *client* dan menjalankan perintah yang diberikan oleh *client*. SSH *server* menyediakan *port* yang berjalan pada *port* 22 dengan koneksi TCP. Contoh aplikasi SSH *server* adalah : OpenSSH

SSH *Client* adalah perangkat lunak yang berfungsi untuk melakukan *request* layanan SSH terhadap SSH *server*. Contoh aplikasi SSH *Client* adalah : Putty dan WinSCP Farras [13].

2.2.5 UBUNTU

Ubuntu adalah sistem operasi dan distribusi Linux yang berbasis Debian, bersifat gratis dan *open-source.* Ubuntu dikembangkan dengan infratruktur Debian dan terdiri dari server, dekstop dan sistem operasi Linux [14].

Ubuntu Server adalah varian dari distro linux Ubuntu yang didesain untuk kebutuhan instalasi pada server. Perbedaan mendasar antara Ubuntu dengan Ubuntu Server adalah pada *user interface,* Ubuntu Server tidak tersedia pada GUI (*Graphical User Interface*), *user interface* ubuntu server menggunakan CLI (*Command Line Interface*). Ubuntu server bersifat gratis dan *open-source* yang dikembangkan oleh Canonical [15].

2.2.6 HONEYPOT

Honeypot adalah sumber daya yang menyediakan layanan server untuk diselidiki, diserang atau dikompromikan. Cara kerja Honeypot seperti membuat tiruan layanan komputer, data atau situs jaringan yang terlihat seperti bagian dari jaringan tersebut tapi sebenarnya layanan komputer dan data tersebut terisolasi dan dimonitor. Honeypot dapat diklasifikasikan berdasarkan tingkat interaksi yang dimiliki.

1. *Low-Interaction* Honeypot

   Low-interaction Honeypot merupakan honeypot yang mempunyai tingkat interaksi honeypot rendah, didesain untuk mengemulasikan layanan (*services*) tiruan yang mirip dengan server yang asli. Penyerang hanya dapat terhubung dengan satu atau beberapa *port* pada *low-interaction* Honeypot.

   Kelebihan *low-interaction* Honeypot diantaranya mudah diinstal mampu mengemulasi layanan seperti *http, ftp, telnet.*, dan dapat berfungsi mendeteksi serangan pada *scanning*. Kekurangan *low-interaction* Honeypot diantaranya layanan yang diberikan hanya emulasi, penyerang tidak berinteraksi dengan server secara penuh, informasi dari penyerang tergolong minim, penyerang mudah menyadari server tiruan.

2. *High-Interaction* Honeypot

   High-interaction honeypot terdapat sistem operasi dimana penyerang ketika menyerang sebuah server dapat berinteraksi dan interaksi tersebut tidak memiliki batasan layaknya seperti respon perintah *user* pada server yang sesungguhnya, namun penyerang tidak mengetahui jika berada di server palsu.

Kelebihan *high-interaction* Honeypot diantara penyerang berinteraksi secara langsung terhadap layanan server, sistem keamanan yang kompleks, sehingga siap berinteraksi dengan penyerang. Kekurangan dari penggunaan *high-interaction* Honeypot diantaranya perencanaan dan instalasi sistem yang rumit dan membutuhkan pengawawan berkala [16].

2.2.7 COWRIE

Cowrie adalah pengembangan dari Honeypot Kippo. Cowrie dikembangkan dengan penambahan fitur baru dan menyediakan layanan emulasi yang dapat merekam sesi penyerang pada server. Dengan merekam sesi, administrator dapat memahami alat pada penyerang, taktik dan prosedur penyerangan [17].

Cowrie adalah honeypot yang didesain untuk menangkap *log* dari cyber attack. Cowrie bekerja pada level *medium to high level interaction SSH* dan *Telnet* [18]. Cowrie dapat "membodohi" penyerang dengan membuat server palsu, sehingga penyerang menganggap sedang berada pada server yang asli. Administrator dapat menganalisa serangan yang didapat pada *log* Cowrie yang berisi informasi sukses/tidaknya penyerangan, lokasi IP penyerang dan *SSH fingerprints* [19].

2.2.8 IPTABLES

Iptables adalah aplikasi yang tersedia pada sistem operasi Linux yang mempunyai fungsi untuk membuat aturan dan melakukan *filter* terhadap trafik pada jaringan ke server, baik paket data yang masuk ke server maupun paket data yang keluar dari server. Iptables juga dapat berfungsi untuk melakukan fungsi NAT. NAT adalah merupakan *internet gateway* yang menghubungkan jaringan lokal dengan jaringan internet.

Iptables memiliki tabel aturan:

1. *Filter* berfungsi Menentukan paket yang akan di *DROP*, *LOG*, *ACCEPT* atau *REJECT.*
2. NAT dapat berfungsi untuk mengubah alamat asal atau tujuan paket data.

3. *Mangle* pada iptables berfungsi untuk melakukan *mangle* pada paket data.

Filter pada iptables memiliki 3 buah *chain*:

1. *Chain FORWARD* berfungsi untuk melakukan *filter* paket yang di-*forward* dari NIC satu ke NIC.
2. *Chain INPUT* berfungsi untuk melakukan *filter* paket menuju *firewall*.
3. *Chain OUTPUT* pada pengaturan iptables berfungsi untuk melakukan *fiter* paket yang keluar dari *firewall*.

NAT memiliki 3 buah *chain*

1. *Chain PRE - ROUTING* pada iptables NAT digunakan untuk mentranlasikan *address* sebelum proses *Routing*.
2. *Chain POST - ROUTING* pada iptables NAT digunakan untuk mentralasikan *address* setelah proses *Routing* terjadi.
3. *Chain OUTPUT* pada iptables NAT berfungsi untuk mentralasikan *address* paket data yang berasal dari *firewall* [20].

2.2.9 *SYSTEM LOG*

System log (*syslog*) memuat rekaman data operating *system* yang memuat data proses pada sistem dan *driver* yang bekerja pada sistem. *Syslog* dapat menampilkan informasi *error* dan peringatan pada komputer tersebut. Dengan adanya *system log*, administrator atau pengguna dapat menganalisa kesalahan pada sistem dan memperbaiki masalah [21].

Cowrie honeypot didesain untuk merekam aktivitas SSH dan Telnet. Sistem *log* dapat merekam data informasi penyerang, menganalisa teknik, taktik dan sistem kerja yang digunakan oleh penyerang, merekam total koneksi serangan yang dilakukan penyerang. Ada 2 jenis file *log* yang digunakan oleh Cowrie, yaitu: cowrie.log untuk menampilkan hasil dalam format *log* dan cowrie.json dimana *output* Cowrie ditampilkan dalam format JSON [22].

2.2.10 SPLUNK

Splunk adalah perangkat lunak berbasis layanan dalam bentuk web untuk membuat indek data mesin dan memudahkan pengguna untuk menganalisa data

yang tersimpan. Splunk dapat menggabungkan dan menganalisa *output* data dari berbagai macam sumber, termasuk *Application Program Interface* (*API*) dan *log* file dari aplikasi, server, perangkat *mobile* dan *websites.*

Splunk memiliki fungsi diantaranya mengizinkan akses dan konten untuk meningkatkan produktivitas pengguna, splunk merupakan *web monitoring* yang bagus untuk analisa dan visualisasi data para *user.* Splunk dapat membuat laporan dalam bentuk grafik, dan tabel untuk memudahkan pengguna. Performa, stabilitas dan inovasi dari Splunk sangat baik untuk memproses danmenampilkan data [23].

2.2.11 WIRESHARK

Wireshark adalah program *Network Protocol Analyzer*. Program ini dapat menangkap semua aktivitas paket yang lewat dan menampilkan informasi secara detail mengenai data tersebut. Cara kerja Wireshark yaitu dengan merekam semua aktivitas paket data yang melewati *interface* dan administrator dapat menganalisa hasil Wireshark

Wireshark dapat menyimpan semua informasi pada interface yang terhubung seperti informasi foto, gambar, video, dan teks. Hal ini memudahkan pengguna untuk menganalisa koneksi *device* yang terhubung dan mengetahui jika terdapat kelainan koneksi yang terjadi jika saat server mendapat serangan *cyber attack*, pengguna dapat menganalisa koneksi hingga informasi serangan [24].

2.2.12 *BRUTE FORCE ATTACK*

Brute force adalah jenis serangan kepada tujuan dengan menggunakan teknik yang ditentukan dan selanjutnya dianalisis sampai proses berhasil. Tingkat kesuksesan serangan ditentukan pada nilai yang ditentukan. Metode *brute force* digunakan untuk membobol password dan dapat digunakan untuk menemukan halaman dan konten yang tersembunyi.

Konsep serangan *brute force* tradisional pada umumnya yaitu penyerang mencoba password dengan kombinasi huruf dan angka untuk membobol password secara berurutan. *Hash* dihasilkan dari *password* acak dan dicocokkan sampai menemukan password yang benar [25].