BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Rujukan penelitian yang pertama yaitu jurnal Deris Stiawan, Mohd. Yazid Idris, Reza Firsandaya Malik, Siti Nurmaini, Nizar Alsharif dan Rahmat Budiarto dari Computer Engineering Department, Faculty of Computer Science, Universitas Sriwijaya, Palembang, Indonesia School of Computing, Faculty of Engineering, Universiti Teknologi Malaysia, Johor Bahru, Malaysia College of Computer Science & Information Technology, Albaha University, Albaha, Saudi Arabia dengan judul *Investigating Brute force Attack Patterns in IoT Network.* Dalam penelitiannya Mereka menganalisa serangan *brute force* pada jaringan *IoT* tanpa melakukan langkah pencegahan.

Rujukan penelitian yang kedua yaitu jurnal Mustapha Adamu Mohammed, Ashigbi Franlin Degadzo, Botchey Francis Effrim, Kwame Anim Appiah mahasiswa Department of Computer Science, Koforidua Technical University, Koforidua, Ghana tahun 2017 dengan judul *Brute force Attack Detection And Prevention on a Network Using Wireshark Analysis*. Dalam penelitiannya peneliti mendeteksi dan mencegah serangan *brute force* dengan menggunakan *software* wireshark.

2.2 Router MiktoTik

Router MikroTik merupakan sistem operasi *Linux Base* yang diperuntukkan sebagai *Network* Router. Didesain untuk memberikan kemudahan bagi penggunanya. Administrasinya bisa dilakukan melalui WinBox. Selain itu instalasi dapat dilakukan pada standar komputer *PC* (*Personal computer*). *PC* yang akan dijadikan Router pun tidak memerlukan *resource* yang cukup besar untuk penggunaan standar, misalnya hanya sebagai *Gateway*. Untuk keperluan beban yang besar (*network* yang kompleks, *routing* yang rumit) disarankan untuk mempertimbangkan pemilihan *resource PC* yang memadai (Zulkifli, 2018).

Gambar 2.1 Router

(Sumber: Web Mikrotik Indonesia)

Router memiliki kemampuan melewatkan paket *Ip address* dari satu jaringan ke jaringan lain yang mungkin memiliki banyak jalur diantara keduanya. Router yang saling terhubung dalam jaringan internet turut serta dalam sebuah Algoritma *routing* terdistribusi untuk menentukan jalur terbaik yang dilalui paket *Ip address* dari sistem ke sistem lain.

Router memiliki antarmuka konfigurasi *command-line* yang kuat namun mudah dipelajari dengan kemampuan *scripting* yang terintegrasi.

a. WinBox GUI overIP dan MAC.
b. CLI dengan Telnet, SSH, konsol lokal dan konsol serial.
c. API untuk memprogram alat Anda sendiri.

2.3 *Firewall*

Firewall adalah sebuah sistem atau perangkat yang mengizinkan lalu lintas jaringan yang dianggap aman untuk melaluinya dan mencegah lalu lintas jaringan yang tidak aman. Umumnya, *firewall* diterapkan dalam sebuah mesin yang terdedikasi, yang berjalan pada *gateway* antara jaringan lokal dan jaringan lainnya. *Firewall* umumnya digunakan untuk mengontrol akses terhadap siapa saja yang memiliki akses terhadap jaringan lokal dari pihak luar. (Suparsin and Mariaty, 2011).

Saat ini, istilah *firewall* menjadi istilah lazim yang merujuk pada sistem yang mengatur komunikasi antar dua jaringan yang berbeda. Secara fundamental , *firewall* dapat berfungsi sebagai berikut:

1. Mengatur dan mengontrol lalu lintas jaringan.
2. Melakukan autentikasi terhadap akses.
3. Melindungi sumber daya dalam jaringan lokal.

2.3.1 Jenis - Jenis *Firewall*

Firewall terbagi menjadi dua jenis, yaitu sebagai berikut:

1. *Personal Firewall*

Didesain untuk melindungi sebuah komputer yang terhubung ke jaringan dari akses yang tidak dikehendaki. *Firewall* jenis ini akhir-akhir ini berevolusi menjadi sebuah kumpulan program yang bertujuan untuk mengamankan komputer secara total dengan ditambahkannya beberapa fitur pengaman tambahan seperti perangkat proteksi terhadap *virus*, *anti-spyware, anti-spam*, dll. Bahkan beberapa produk *firewall* lainnya dilengkapi dengan fungsi pendeteksian gangguan keamanan jaringan (*Intrusion Detection System*). Contoh *firewall* dari jenis ini adalah *Microsoft Windows Firewall*. *Personal firewall* secara umum hanya memiliki dua fitur utama, yaitu *Packet Filter Firewall* dan *Stateful Firewall*.

2. *Network Firewall*

Didesain untuk melindungi jaringan secara keseluruhan dari berbagai serangan. Ada dua bentuk yaitu sebuah perangkat yang terdedikasi atau sebuah perangkat lunak yang diinstalasikan dalam sebuah *server*. *Network Firewall* secara umum mempunyai beberapa fitur utama yaitu *Packet Filter Firewall, Stateful Firewall, Circuit Level Gateway, Application Level Gateway*, dan *NAT Firewall*. *Network Firewall* umumnya bersifat transparan (tidak terlihat) dari pengguna dan menggunakan teknologi *routing* untuk menentukan paket mana yang dizinkan, dan paket mana yang ditolak (Suparsin and Mariaty, 2011).

2.4 Keamanan Data Jaringan

Keamanan data jaringan adalah data-data yang berada pada perangkat keras dan perangkat lunak dalam sistem jaringan dilindungi dari tindakan-tindakan yang bersifat jahat atau merusak, modifikasi dan hal-hal yang bersifat membocorkan data ke pihak lain, untuk memastikan sistem akan berjalan secara konsisten dan handal tanpa adanya gangguan pada sistem tersebut.

Keamanan jaringan adalah komponen yang paling vital dalam keamanan informasi karena bertanggung jawab untuk mengamankan semua informasi melewati komputer berjejaring. Keamanan Jaringan mengacu pada semua fungsi perangkat keras dan perangkat lunak, karakteristik, fitur, prosedur operasional, akuntabilitas, tindakan, kontrol akses, dan administrasi dan manajemen kebijakan yang diperlukan untuk memberikan tingkat perlindungan yang dapat diterima untuk perangkat keras dan perangkat lunak, dan informasi dalam jaringan.

Masalah keamanan jaringan dapat dibagi secara kasar menjadi empat bidang yang saling terkait: kerahasiaan, otentikasi, nonrepudiation, dan integrity control. Kerahasiaan berkaitan dengan menjaga informasi dari tangan dari pengguna yang tidak berhak. Inilah yang biasanya terlintas dalam pikiran ketika orang memikirkan keamanan jaringan. Otentikasi berhubungan dengan menentukan siapa yang Anda ajak bicara sebelum mengungkapkan informasi sensitif atau memasuki kesepakatan bisnis. Nonrepudiasi berurusan dengan tanda tangan Integritas Pesan: Sekalipun pengirim dan penerima saling mengotentikasi, mereka juga ingin memastikan bahwa isi komunikasi mereka tidak berubah (Fauzi and Suartana 2018).

2.5 *Password Cracking*

Sebuah *password* dapat dibongkar dengan menggunakan program yang disebut sebagai *password cracker*. Program *password cracker* adalah program yang mencoba membuka sebuah *password* yang telah terenkripsi dengan menggunakan sebuah algoritma tertentu dengan cara mencoba semua kemungkinan. Teknik ini sangatlah sederhana, tapi efektivitasnya luar biasa, dan

tidak ada satu pun sistem yang aman dari serangan ini, meski teknik ini memakan waktu yang sangat lama, khususnya untuk *password* yang rumit.

Namun ini tidak berarti bahwa *password cracker* membutuhkan *decrypt*. Pada praktiknya, mereka kebayakan tidak melakukan itu. Umumnya, kita tidak dapat melakukan *decrypt password-password* yang sudah terenkripsi dengan algoritma yang kuat. Proses-proses enkripsi modern kebanyakan hanya memberikan satu jalan, di mana tidak ada proses pengembalian enkripsi. Namun, anda menggunakan tool – tool simulasi yang mempekerjakan algoritma yang sama yang digunakan untuk mengenkripsi *password* orisinal. *Tool - tool* tersebut membentuk analisa komparatif.

Program *password cracker* tidak lain adalah mesin-mesin ulet. Ia akan mencoba kata demi kata dalam kecepatan tinggi. Mereka menganut "Asas Keberuntungan", dengan harapan bahwa pada kesempatan tertentu mereka akan menemukan kata atau kalimat yang cocok. Teori ini mungkin tepat mengena pada anda yang terbiasa membuat *password* asal-asalan. Dan memang pada kenyataannya, *passwordpassword* yang baik sulit untuk ditembus oleh program *password cracker* (Pramudita, 2011).

2.6 *Brute force*

2.6.1 Definisi *Brute force*

Algoritma *brute force* adalah algoritma yang memecahkan masalah dengan sangat sederhana, langsung, dan dengan cara yang jelas/lempang. Penyelesaian permasalahan *password* cracking dengan menggunakan algoritma *brute force* akan menempatkan dan mencari semua kemungkinan *password* dengan masukan karakter dan panjang *password* tertentu tentunya dengan banyak sekali kombinasi *password*.

Algoritma *brute force* adalah algoritma yang lempang atau apa adanya. Pengguna hanya tinggal mendefinisikan karakter set yang diinginkan dan berapa ukuran dari *password*nya. Tiap kemungkinan *password* akan di generate oleh algoritma ini (Pramudita, 2011).

2.6.2 Definisi Serangan *Brute force*

Serangan *brute force* adalah sebuah teknik serangan terhadap sebuah sistem keamanan komputer yang menggunakan percobaan terhadap semua kunci yang mungkin. Pendekatan ini pada awalnya merujuk pada sebuah program komputer yang mengandalkan kekuatan pemrosesan komputer dibandingkan kecerdasan manusia. Sebagai contoh, untuk menyelesaikan sebuah persamaan kuadrat seperti $x^2+7x-44=0$, di mana x adalah sebuah integer, dengan menggunakan teknik serangan *brute force*, penggunanya hanya dituntut untuk membuat program yang mencoba semua nilai integer yang mungkin untuk persamaan tersebut hingga nilai x sebagai jawabannya muncul. Istilah *brute force* sendiri dipopulerkan oleh Kenneth Thompson, dengan mottonya: "*When in doubt, use brute force*" (jika ragu, gunakan *brute force*).

Secara sederhana, menebak *password* dengan mencoba semua kombinasi karakter yang mungkin. *Brute force attack* digunakan untuk menjebol akses ke suatu *host* (*server/workstation/network*) atau kepada data yang terenkripsi. Metode ini dipakai para *cracker* untuk mendapatkan *account* secara tidak sah, dan sangat berguna untuk memecahkan enkripsi. Enkripsi macam apapun, seperti *Blowfish, AES, DES, Triple DES*, dsb secara teoritis dapat dipecahkan dengan *brute force attack*. Pemakaian *password* sembarangan, memakai *password* yang cuma sepanjang 3 karakter, menggunakan kata kunci yang mudah ditebak, menggunakan *password* yang sama, menggunakan nama, memakai nomor telepon, sudah pasti sangat tidak aman. Namun *brute force attack* bisa saja memakan waktu bahkan sampai berbulan - bulan atau bertahun - tahun bergantung dari bagaimana rumit *password*nya.

Brute force attack tidak serumit dan *low-tech* seperti algoritma *hacking* yang berkembang sekarang. Seorang penyerang hanya cukup menebak anama dan kombinasi *password* sampai dia menemukan yang cocok. Mungkin terlihat bahwa *brute force attack* atau *dictionary attack* tidak mungkin berhasil. Namun yang mengejutkan, kemungkinan berhasil *brute force attack* menjadi membaik ketika *site* yang ingin diretasi tidak dikonfigurasikan dengan baik (Pramudita, 2011).

2.6.3 Metode Serangan *Brute force*

Brute force attack ada sebuah metode untuk menjebol kode rahasia (yaitu, mendekripsi sebuah teks yang telah terenkripsi) dengan mencoba semua kemungkinan kunci yang ada. *Feasibility* dari sebuah *brute force attack* tergantung dari panjangnya *cipher* yang ingin dipecahkan , dan jumlah komputasi yang tersedia untuk penyerang. Salah satu contohnya bernama *Cain's Brute force Password Cracker* mencoba semua kombinasi yang mungkin dari karakter yang telah didefinisikan sebelum atau set karakter yang kustom melawan sebuah *password* yang telah terenkripsi di *brute force* dialog.

Kuncinya adalah mencoba semua kemungkinan *password* dengan formula seperti berikut.

KS = L(m) + L(m+1) + L(m+2) + ... + L(M)

L = jumlah karakter yang kita ingin definsikan

m = panjang minimum dari kunci

M = panjang maksimal dari kunci

Contohnya saat kita ingin meretas sebuah LanManager paswords (LM) dengan karakter set "ABCDEFGHIJKLMNOPQRSTUVWXYZ" dengan jumlah 26 karakter, maka brute fore *cracker* harus mencoba KS = 26^1 + 26^2 + 26^3 + ... + 26^7 = 8353082582 kunci yang berbeda. Jika ingin meretas *password* yang sama denganset karakter set "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789! @#$%^&*()-_+=~`[]{}|\:;'"<>,.?/", jumlah kunci akan dihasilkan akan naik menjadi 6823331935124.

Brute force attack melakukan perbandingan *string matching* antara *pattern* dengan *text* per karakter dengan *pseudocode* berikut : *do if (text letter == pattern letter) compare next letter of pattern to next letter of text else move pattern down text by one letter while (entire pattern found or end of text) Exhaustive key search cracking* mungkin saja memerlukan waktu yang sangat panjang untuk berhasil, tetapi jika character setnya sidah benar sesaui *password*, maka tinggal hanyalah jadi masalah waktu (Pramudita, 2011).

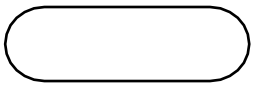
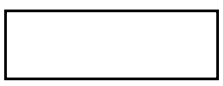
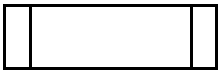
2.7 *Flowchart*

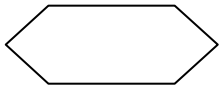
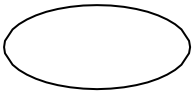
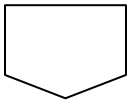
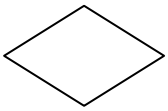
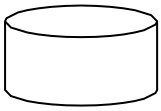
Flowchart adalah cara penyajian visual aliran data melalui sistem informasi. *Flowchart* dapat membantu menjelaskan pekerjaan yang saat ini dilakukan dan bagaimana cara meningkatkan atau mengembangkan pekerjaan tersebut. Dengan menggunakan *flowchart* dapat juga membantu untuk menemukan elemen inti dari sebuah proses, selama garis digambarkan secara jelas antara di mana suatu proses berakhir dan proses selanjutnya dimulai.

Tujuan utama penggunaan *flowchart* adalah untuk menggambarkan suatu tahap penyelesaian masalah secara sederhana, terurai, rapi, dan jelas dengan menggunakan simbol-simbol yang standar. Dalam penulisan *flowchart* dikenal dua model yaitu *flowchart* sistem dan *flowchart* program. *Flowchart* sistem merupakan diagram alir yang menggambarkan suatu sistem peralatan komputer yang digunakan dalam proses pengolahan data serta hubungan antara peralatan tersebut. *Flowchart* program merupakan diagram alir yang menggambarkan suatu logika dari suatu prosedur pemecahan masalah (Kristanti, 2012).

Simbol diagram *Flowchart* dapat diliihat pada tabel 2.1 dibawah ini:

Tabel 2.1 Simbol Diagram *Flowchart*

No	Simbol	Keterangan
1.		Terminal menyatakan awal atau akhir dari suatu algoritma.
2.		Menyatakan proses.
3.		Proses yang terdefinisi atau sub program.

4.		Persiapan yang digunakan untuk memberi nilai awal suatu besaran.
5.		Menyatakan masukan dan keluaran (*input/output)*.
6.		Menyatakan penyambung ke simbol lain dalam satu halaman.
7.		Menyatakan penyambung ke halaman lainnya.
8.		Menyatakan pencetakan (dokumen) pada kertas.
9.		Menyatakan *desicion* (keputusan) yang digunakan untuk penyeleksian kondisi di dalam program.
10.		Menyatakan media penyimpanan drum magnetik.