JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 1, Maret 2025, Pp. 720-738

IMPLEMENTASI KEAMANAN JARINGAN KOMPUTER DENGAN IPTABLES SEBAGAI FIREWALL MENGGUNAKAN *PORT KNOCKING* METODE DINAMIS

Reza Setya Budi *[1)], Irwan Sembiring [2)]
1. Teknik Informatika, Fakultas Teknologi Informasi, Universitas Kristen Satya Wacana, Indonesia
2. Teknik Informatika, Fakultas Teknologi Informasi, Universitas Kristen Satya Wacana, Indonesia

Article Info
Kata Kunci: Dinamis; Firewall; *Port*; *Port knocking;* SSH

Keywords: Dinamis; Firewall; *Port*; *Port knocking;* SSH

Article history:
Received 24 October 2024
Revised 15 November 2024
Accepted 18 December 2024
Available online 1 March 2025

DOI :
https://doi.org/10.29100/jipi.v10i1.5750

* Corresponding author.
Corresponding Author
E-mail address:
672020181@student.uksw.edu

ABSTRAK
Keamanan jaringan adalah aspek penting dalam perlindungan data dan layanan dari ancaman siber. Salah satu metode inovatif yang digunakan untuk meningkatkan keamanan adalah dinamis *port knocking* (*dynamic port knocking*). Metode ini menggabungkan konsep dasar *port knocking* dengan elemen dinamis untuk memberikan lapisan perlindungan yang lebih kuat terhadap akses tidak sah. *Port knocking* tradisional melibatkan pengiriman serangkaian koneksi ke *port* tertutup dalam urutan tertentu untuk membuka akses ke layanan yang dilindungi. Namun, pendekatan ini dapat rentan terhadap serangan *brute force* dan *replay*. Dinamis *port knocking* memperbaiki kelemahan ini dengan mengubah urutan dan *port* yang harus diketuk berdasarkan parameter dinamis, seperti waktu atau informasi sesi yang dienkripsi. Dalam dinamis *port knocking*, pola *knocking* dapat berubah secara periodik atau berdasarkan algoritma tertentu, sehingga lebih sulit bagi penyerang untuk menebak urutan yang benar. Parameter dinamis dapat disesuaikan untuk menambah lapisan keamanan tambahan, seperti menggunakan token berbasis waktu atau informasi unik lainnya yang hanya diketahui oleh pengguna sah. Keuntungan utama dari dinamis *port knocking* meliputi peningkatan keamanan melalui perubahan urutan *port* secara berkala, mengurangi risiko deteksi oleh penyerang, dan meningkatkan kompleksitas serangan *brute force* dan replay. Selain itu, metode ini dapat diintegrasikan dengan protokol keamanan lain untuk membangun sistem pertahanan yang lebih komprehensif. Namun, implementasi dinamis *port knocking* juga memiliki tantangan, termasuk kebutuhan akan sinkronisasi waktu yang presisi antara klien dan server, serta kompleksitas dalam pengaturan dan pemeliharaan sistem. Dengan desain yang hati-hati dan pemanfaatan teknologi enkripsi yang kuat, dinamis *port knocking* dapat menjadi elemen penting dalam strategi keamanan jaringan modern, memastikan bahwa hanya pengguna yang berwenang dapat mengakses sumber daya yang dilindungi.

ABSTRACT
Network security is a crucial aspect of protecting data and services from cyber threats. One innovative method used to enhance security is dynamic *port knocking*. This method combines the basic concept of *port knocking* with dynamic elements to provide a stronger layer of protection against unauthorized access. Traditional *port knocking* involves sending a series of connections to closed *port*s in a specific sequence to open access to protected services. However, this approach can be vulnerable to brute force and replay attacks. Dynamic *port knocking* addresses these weaknesses by changing the sequence and *port*s to be *knock*ed based on dynamic parameters, such as time or encrypted session information. In dynamic *port knocking*, the *knocking* pattern can change periodically or based on a specific algorithm, making it more difficult for attackers to guess the correct sequence. Dynamic parameters can be adjusted to add additional security layers, such as using time-based tokens or other unique information known only to authorized users. The main advantages of dynamic *port knocking* include enhanced security through periodic changes in *port* sequences, reducing the risk of detection by attackers, and increasing the complexity of brute force and replay attacks. Additionally, this method can be integrated with other security protocols to build a more

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 1, Maret 2025, Pp. 720-738

comprehensive defense system. However, implementing dynamic *port knocking* also presents challenges, including the need for precise time synchronization between clients and servers, and the complexity of system setup and maintenance. With careful design and the use of strong encryption technology, dynamic *port knocking* can be an im*port*ant element in modern network security strategies, ensuring that only authorized users can access protected resources.

.

I. PENDAHULUAN

Keamanan jaringan adalah komponen penting dari sebuah sistem karena membantu menjaga validitas dan intregritas data serta memastikan bahwa layanan dapat diakses oleh pengguna. Menurut definisi keamanan jaringan komputer oleh Organisasi Internasional untuk Standardisasi, keamanan jaringan komputer mengacu pada perlindungan terhadap perangkat keras, perangkat lunak, dan sumber daya data dalam sistem komputer[1]. Sistem keamanan jaringan komputer harus dilindungi dari serangan dan pemindaian oleh orang yang tidak berhak. Jaringan komputer semakin penting untuk berbagai tujuan, seperti pendidikan, pekerjaan, dan permainan. Karena banyaknya akses ke jaringan, akan ada banyak peluang kejahatan di jaringan atau peretas yang dapat menghancurkan sumber daya server. Jaringan komputer adalah sebuah sistem yang terdiri dari beberapa komputer yang digunakan untuk berbagi sumber daya seperti *printer* dan *CPU*, berkomunikasi melalui surel dan pesan instan, dan menggunakan peramban web untuk mengakses data. Tujuan dari jaringan komputer adalah untuk memastikan bahwa setiap komponen dalam jaringan dapat meminta dan memberikan layanan. Komponen yang meminta atau menerima layanan disebut klien, sedangkan yang memberikan atau mengirim layanan disebut server[2]. Dalam penggunaan perangkat banyak aplikasi yang terhubung dengan jaringan baik itu lokal maupun jaringan internet yang menyebabkan *port* komunikasi terbuka. Melalui *port* komunikasi tersebut akan menjadikan komunikasi antar perangkat lain akan berjalan dengan lancar. Terbukanya *port* itu menjadi celah untuk bisa diakses oleh orang tidak bertanggung jawab secara ilegal. *Port* komunikasi yang terbuka memungkinkan penyusup masuk ke perangkat dan akan bisa menguasai seluruh perangkat jika dibiarkan. Oleh karena itu terdapat *firewall* yang dapat digunakan.

Firewall merupakan sistem keamanan yang melindungi komputer dari berbagai ancaman di jaringan internet, berfungsi sebagai pembatas antara komputer dan jaringan internet. *Firewall* membatasi akses dengan menutup *port* yang tidak penting dan memblokir *port* yang berbahaya. Namun, penggunaan firewall dapat membatasi komunikasi dengan perangkat lain di jaringan, bahkan menolak akses untuk terhubung. Untuk mengatasi hal ini, metode *port knocking* merupakan solusi yang sangat efektif. *Port knocking* adalah teknik untuk membangun komunikasi antar komputer dalam suatu jaringan tanpa membuka *port* komunikasi secara bebas. Dengan metode ini, perangkat komputer dapat diakses dari luar menggunakan pola konfigurasi *port knocking*, yaitu serangkaian koneksi yang dikirimkan ke *port* tertentu. *Port knocking* terbagi menjadi dua jenis, yaitu statis dan dinamis, yang menentukan urutan ketukan yang diperlukan.

Pada penelitian berjudul ” Analisa dan Implementasi Sistem Keamanan Jaringan Komputer dengan Iptables sebagai Firewall menggunakan Metode *Port knocking*”, berisi tentang implementasi *port knocking* yang dapat menyembunyikan *port* yang terbuka maupun tertutup, sehingga orang lain tidak bisa mengetahui[3]. Penelitian yang berjudul “IMPLEMENTASI KEAMANAN JARINGAN DENGAN IPTABLES SEBAGAI FIREWALL MENGGUNAKAN METODE *PORT KNOCKING*”, Penelitian ini menyimpulkan bahwa penggunaan daya server saat menjalankan daemon relatif stabil, sehingga kinerja server tetap terjaga. Administrator dapat menghubungkan server meskipun semua *port* sedang diblokir. Konfigurasi yang dikembangkan untuk memanfaatkan *port* memberikan perlindungan yang memadai terhadap serangan dari luar. Baik dengan menggunakan firewall maupun tanpa firewall, *port* yang digunakan untuk autentikasi *port knocking* tetap tidak terlihat saat dilakukan stealth scanning menggunakan nmap[4]. Penelitian yang berjudul “Rancang Bangun Keamanan *Port* Secure Shell (SSH) Menggunakan Metode *Port knocking*”, Penelitian ini bertujuan membangun konfigurasi *Port* SSH dengan menggunakan metode *Port knocking* dengan pemanfaatan *port* SSH yang praktis dalam memantau jaringan internet pada server[5]. Penelitian yang berjudul “IMPLEMENTASI METODE *PORT KNOCKING* PADA SISTEM KEAMANAN SERVER UBUNTU VIRTUAL BERBASIS WEB MONITORING”, Penelitian ini berhasil mengamankan server yang memiliki layanan jaringan pada *port* terbuka dengan memblokir *port* 22 dan menggunakan *port knocking* untuk membuka *port* SSH 22 saat diperlukan. *Port knocking* efektif dalam mengatasi

masalah pemblokiran *port* oleh firewall. Setelah melakukan *port* scanning, serangan *DDOS*, dan *brute force*, metode *port knocking* terbukti mampu melindungi server secara efektif[6].

Berdasarkan pada penelitian terdahulu, Pada penelitian ini akan dilakukan implementasi metode dinamis *port knocking* dengan membangun dan memperbaiki metode *port knocking* statis, penelitian tentang metode *port knocking* dinamis membuat keamanan yang lebih tinggi dengan mengubah urutan *knock* secara berkala atau berdasarkan kejadian tertentu, metode dinamis menambah tingkat kompleksitas bagi penyerang[7]. Perubahan urutan *knock* yang terus-menerus membuatnya sulit untuk ditebak atau digunakan ulang oleh penyerang, sehingga meningkatkan keamanan sistem secara keseluruhan..Metode ini memberikan fleksibilitas, adaptabilitas, dan responsivitas yang lebih tinggi terhadap ancaman yang terus berkembang, menjadikannya pendekatan yang lebih efektif dalam keamanan jaringan modern[8]. Penelitian tentang *port knocking* dinamis sangat relevan untuk meningkatkan keamanan jaringan dengan mengontrol akses ke *port* menggunakan paket khusus dan mencegah serangan zero-day. Konfigurasi dinamis memberikan fleksibilitas dalam menyesuaikan aturan akses *port* sesuai kebutuhan keamanan yang berkembang, serta integrasi yang efektif dengan firewall untuk melindungi dari serangan luar dan teknik *port* scanning.

II. METODE PENELITIAN

Dasar teori yang digunakan dalam penelitian ini antara lain :

A. *Port*

Port logis adalah jalur yang digunakan oleh aplikasi untuk berkomunikasi dengan komputer lain melalui jarigan TCP/IP. Ketika komputer terhubung ke internet adalah salah satu contohnya. Jaringan komputer memiliki banyak port yang berfungsi sebagai berbagai jenis dan fungsi. Ini termasuk *port* 20 dan 21 (*FTP*), *port* 22 (*SSH*), *port* 23 (*Telnet*), *port* 25 (*SMTP*), *port* 53 (*DNS*), *port* 443 (*HTTPS*), *port* 143 (*IMAP*), *port* 8080 dan 3128 (*Proxy*), *port* 67 dan 68 (*DHCP*), serta *port* 80 (*HTTP/Web Server*)[9]. Selain itu, terdapat *port* fisik dan *port* serial.

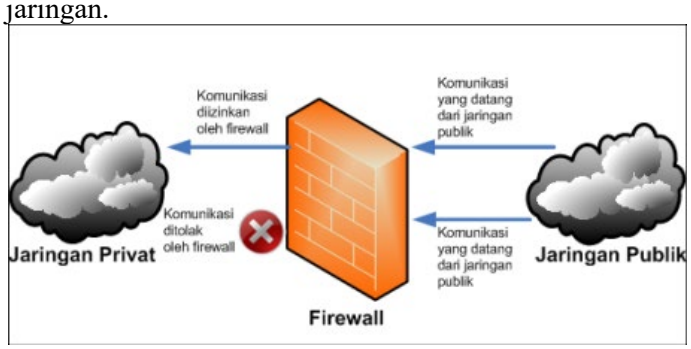
B. *Iptables*

Iptables merupakan aplikasi untuk sistem operasi Linux yang memungkinkan penyaringan atau filtrasi lalu lintas data[10]. Sederhananya, dapat disebut sebagai pengatur lalu lintas data, ini memungkinkan untuk mengatur semua jenis lalu lintas yang masuk, keluar, dan melewati komputer. Sebagai sistem firewall dalam sumber terbuka (open source), iptables mendukung berbagai lapisan OSI seperti lapisan 3 (Network layer), lapisan 4 (Trans*port* layer), dan lapisan 7[11].

Iptables menerapkan prinsip alamat IP, protokol (*TCP, UDP, ICMP*), dan *port*. Selain itu, konsep chain (*INPUT, OUTPUT,* dan *FORWARD*) juga digunakan oleh iptables. Jika data yang sedang diproses melibatkan paket IP, paket tersebut akan melewati tabel penyaringan terlebih dahulu berdasarkan chain yang ditentukan, seperti *INPUT, OUTPUT,* atau *FORWARD*[12].

C. *Firewall*

Firewall adalah teknologi keamanan siber yang digunakan untuk memperkuat keamanan komputer yang terhubung ke jaringan, seperti internet atau *Jaringan Area Lokal (LAN)*[13]. Firewall merupakan suatu perangkat atau program yang berperan dalam mengontrol lalu lintas jaringan antar host atau jaringan dengan menerapkan berbagai langkah keamanan. Firewall biasanya dibicarakan tentang konektivitas internet, tetapi juga diterapkan pada jenis koneksi jaringan lainnya. Sebagai contoh, banyak perusahaan menerapkan firewall dalam jaringannya untuk membatasi koneksi ke dan dari jaringan internal, yang digunakan untuk fungsi tertentu seperti personal dan akuntansi. Dengan memanfaatkan firewall, kontrol konektivitas di berbagai bagian jaringan dapat diatur untuk mencegah akses yang tidak diizinkan oleh sistem. Penggunaan firewall yang tepat dapat memberikan keamanan jaringan yang kuat untuk host dan sebuah jaringan.

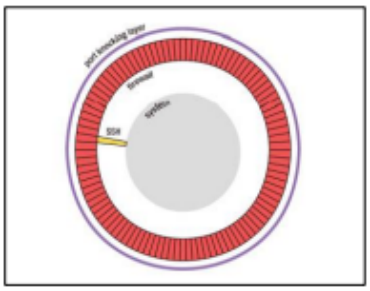


Gambar 1 Firewall

D. Port knocking

Port knocking adalah ide untuk menyembunyikan layanan jarak jauh di balik firewall dengan memberikan akses ke *port* tersebut hanya setelah klien berhasil melewati proses otentikasi pada firewall. Pendekatan ini bertujuan untuk mencegah pemindai mengetahui layanan yang tersedia di host dan juga berfungsi sebagai bentuk pertahanan terhadap serangan *zero-day*. *Port knocking* adalah sistem keamanan yang menggunakan firewall pada perangkat jaringan untuk membuka atau menutup akses ke port tertentu. Ini dicapai dengan mengirimkan koneksi atau paket tertentu. Untuk koneksi ini, protokol TCP, UDP, atau ICMP dapat digunakan[14].

Metode *port knocking* memungkinkan untuk berkomunikasi dari mana saja dengan perangkat komputer yang tidak membuka *port* komunikasi apapun secara bebas. Dengan kata lain, meskipun perangkat komputer ini tidak memiliki port komunikasi yang terbuka bebas untuk dimasuki, Anda masih dapat mengakses perangkat dari luar[15]. Gagasan di balik *port knocking* adalah menyembunyikan layanan dari jarak jauh di belakang firewall dan hanya mengizinkan akses ke *port* tersebut ketika klien telah berhasil diautentikasi dengan firewall. Ini dapat melindungi server dari serangan zero-day dan mencegah pemindai mengetahui layanan apa yang saat ini tersedia.



Gambar 2 Port knocking

Gambar menunjukkan bahwa port server tertutup dan tidak ada yang bisa mengaksesnya karena port tersebut dilindungi oleh port knocking. Dengan metode ini, user harus memberikan autentikasi ke server agar firewall menulis ulang aturannya sehingga user dapat mengakses port tersebut. Setelah selesai, user mengirimkan autentikasi kembali untuk menutup port agar firewall menghapus aturan yang ditulis sebelumnya yang memungkinkan port untuk dibuka[16].

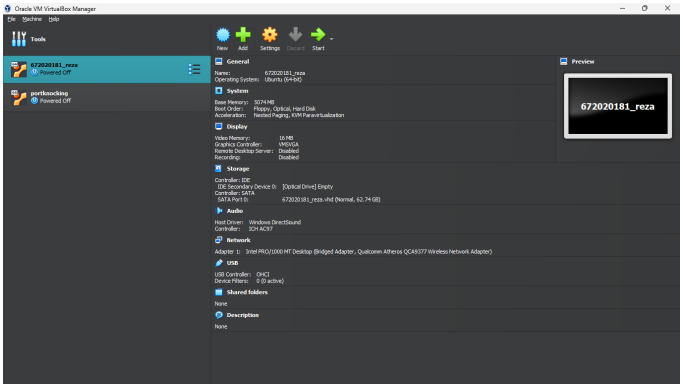
E. SSH

Protokol jaringan yang disebut Secure Shell (juga dikenal sebagai Secure Socket Shell atau Secure Shell) memungkinkan pengguna, terutama administrator sistem, untuk mengakses komputer mereka melalui jaringan yang tidak aman[17]. SSH juga merupakan nama sekumpulan utilitas yang digunakan oleh protokol SSH. Secure Shell adalah salah satunya, yang memungkinkan otentikasi kata sandi dan kunci publik yang kuat serta komunikasi data terenkripsi antara dua komputer yang terhubung melalui jaringan terbuka, seperti internet. Selain itu, SSH banyak digunakan oleh administrator jaringan untuk mengelola sistem dan aplikasi dari jarak jauh, memungkinkan mereka untuk masuk ke komputer orang lain. SSH adalah singkatan dari protokol jarin-gan kriptografi dan rangkaian utilitas yang digunakan untuk mengimplementasikannya. SSH menggunakan model client-server, menghubungkan aplikasi klien Secure Shell, yang merupakan akhir tempat sesi ditampilkan, dengan server SSH. [5]. SSH biasanya mendukung protokol aplikasi yang digunakan untuk emulasi terminal atau transfer file.

F. VirtualBox

VirtualBox adalah program *virtualisasi* gratis dan open source yang dapat digunakan untuk komputer desktop, server, dan laptop. Program ini memungkinkan virtualisasi sistem operasi *32-bit* dan *64-bit* pada komputer yang menggunakan *prosesor Intel* dan *AMD*, serta virtualisasi perangkat lunak dan perangkat keras. Alasan utama untuk menggunakan *virtualBox* adalah karena program ini menawarkan banyak kemudahan dalam virtualisasi, seperti virtualisasi perangkat lunak dan perangkat keras.

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 1, Maret 2025, Pp. 720-738



Gambar 3 Tampilan Virtual Box

G. *Linux Ubuntu*

Ubuntu adalah sejenis sistem operasi berbasis *Debian* yang didistribusikan sebagai perangkat lunak sistem operasi yang bebas. Perusahaan Canonical LTD mendukung proyek Ubuntu, yang berasal dari Afrika Selatan. Namanya berasal dari filosofi Afrika Selatan yang berarti "Kemanu-siaan kepada sesama". Meskipun dimaksudkan untuk pengguna individu, Ubuntu juga tersedia sebagai sistem operasi server. Sangat banyak versi Ubuntu, termasuk *Kubuntu, Xubuntu, Lubuntu, Edubuntu, Mythbuntu,* dan *BlackBuntu*, tetapi hanya tiga yang dirilis oleh *Canonical LTD: Xub-untu, Lubuntu,* dan *Kubuntu*.

Gambar 4 Tampilan Ubuntu

H. *Kali Linux*

Kali Linux adalah distribusi Linux khusus yang dikembangkan untuk keperluan pengujian keamanan dan penetrasi. Distribusi ini dirancang untuk menyediakan sejumlah besar perangkat lunak keamanan dan forensik yang diperlukan oleh para profesional keamanan informasi. Kali Linux dibuat dengan tujuan utama untuk memberikan lingkungan uji penetrasi yang kokoh dan kaya fitur. Ini memungkinkan para profesional keamanan untuk mengidentifikasi dan mengevaluasi kelemahan dalam suatu sistem atau jaringan.

Kali Linux dilengkapi dengan lebih dari 600 alat keamanan dan penetrasi, termasuk alat untuk pemindaian jaringan, analisis forensik, pengujian penetrasi, dan pemulihan data. Kali Linux dibangun di atas distribusi Debian, salah satu distribusi Linux yang sangat stabil. Ini memberikan keamanan dan stabilitas dasar, sambil menambahkan repositori khusus untuk alat keamanan.

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 1, Maret 2025, Pp. 720-738

Gambar 5 Tampilan Kali Linux

Penelitian dilakukan menggunakan virtual machine yaitu ubuntu 20.20 sebagai server dan kali linux sebagai klien. Dengan menggabungkan bahasa dasar pemrograman python yaitu library pypacker. Pypacker adalah sebuah library Python yang digunakan untuk memanipulasi dan menganalisa paket-paket jaringan seperti Ethernet, IP, TCP, UDP, DNS, dan protokol jaringan lainnya. Library ini memungkinkan untuk membuat atau mengubah paket-paket jaringan secara programatik, yang berguna dalam berbagai aplikasi seperti pembuatan alat keamanan jaringan, pemantauan jaringan, dan pengujian keamanan.

Penggunaan library ini tidak secara khusus terkait dengan *port* dalam konteks penggunaan pada umumnya. Namun, dalam implementasi *port knocking*, pypacker dapat digunakan untuk membuat dan mengirimkan paket-paket jaringan yang dibutuhkan untuk melakukan *knocking* ke *port-port* tertentu. *Port knocking* sendiri adalah sebuah teknik keamanan yang digunakan untuk mengamankan akses ke layanan jaringan dengan cara "memukul" (*knocking*) urutan *port-port* tertentu sebelum akses ke layanan tersebut dibuka[18]. Dalam implementasinya, pypacker digunakan untuk mengirimkan paket UDP atau TCP ke *port-port* yang telah ditentukan sesuai dengan urutan *knocking* yang diinginkan. Setelah urutan *knocking* selesai, akses ke layanan yang terkunci biasanya akan terbuka untuk sementara waktu, memungkinkan koneksi dari host yang melakukan *knocking*.

Kombinasi ketukan dinamis berada pada file ketuk.py. Hal yang dimaksudkan dinamis disini adalah ketukan yang dapat berubah-ubah. Namun, tetap berada pada aturan atau skema yang telah ditentukan pada ketuk.py. skema urutan masuk *port* yaitu ketukan 1, ketukan 2, ketukan 3 dan untuk keluar *port* yaitu ketukan 3, ketukan 2, ketukan 1. Dengan range *port* yaitu ketukan 1 dari range 1000-1999, ketukan 2 dari range 2000-2999, dan ketukan 3 dari range 3000-3999.

Uji validasi untuk memastikan bahwa client dapat melakukan koneksi ke remote server ssh menggunakan putty. Langkah-langkah dalam kinerja menggunkan 3 step ketukan dengan range waktu 0,5. Time out memberikan indikasi batasan waktu koneksi, jika Upaya koneksi ke server tidak dapat dilakukan melebihi batas waktu 0.5 detik maka upaya koneksi akan dihentikan. Keberhasilan penelitian dapat diukur dari Putty yang berhasil memblokir koneksi ssh. Putty ada

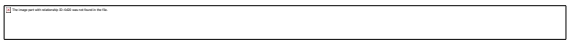
III. HASIL DAN PEMBAHASAN

Penelitian ini dilakukan menggunakan hardware yaitu Laptop dengan spesifikasi *Processor Intel Core I3-8145U CPU @2.10Ghz, RAM 8GB.* Penelitian ini terdiri dari proses instalasi *software* yaitu *Virtual Box*, proses instalasi *Ubuntu Lts 20.04* pada *Virtual Machine* sebagai server, proses instalasi *Kali Linux* sebagai klien.

A. Instalasi *phyton3*

Sebelum program *port knocking* dijalankan, pertama kali dilakukan instalasi program pada komputer target yang berfungsi untuk mendeteksi adanya ketukan *port* secara dinamis dari komputer klien. Tahap awal adalah menyiapkan instalasi pypacker, Pypacker merupakan Pustaka program yang digunakan untuk memanipulasi paket dengan cepat dan sederhana dengan bahasa dasar pemrograman Python. Dengan pypacker dapat membuat paket secara manual dengan mendefinisikan setiap aspek dari semua data header, membedah paket dengan mengurai byte paket, mengirim/menerima paket pada lapisan berbeda dan mencegah paket. Instalasi python 3 dilakukan dalam komputer server dan klien, dengan cara sebagai berikut :

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 1, Maret 2025, Pp. 720-738



Gambar 6 Intalasi Python3

Setelah dilakukan instalasi python 3, pada komputer server dapat dilakukan instalasi pypacker menggunakan pip dengan perintah sebagai berikut :

pip install pypacker

Gambar 7 Instalasi Pypacker

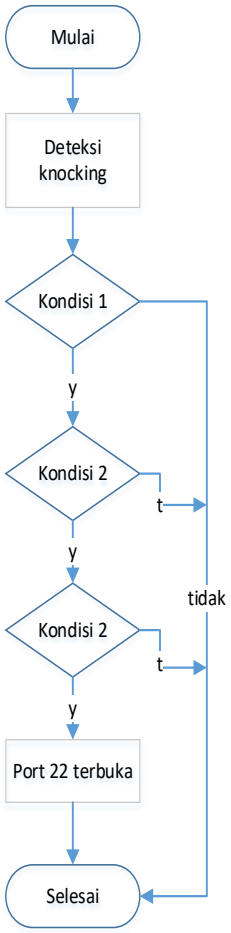
Proses berikut menunjukkan bahwa instalasi pypacker telah berhasil

Gambar 8 berhasil instal pypacker

B. Konfigurasi Program

Konfigurasi program yang dilakukan pada komputer server adalah dengan cara mengatur batasan atau range *port* dalam masing-masing skenario atau kondisi. Terdapat 3 urutan *knock* yang terbagi menjadi 3 kondisi. Jika kondisi terpenuhi maka *port* dibuka terhadap klien tertentu.

Alur deteksi *knocking* berdasarkan konfigurasi kondisi yang telah ditentukan dijelaskan pada gambar alur sistem berikut.



Gambar 9 proses ketukan

Langkah selanjutnya adalah konfigurasi program pada komputer target agar dapat mendengar ketukan secara dinamis. Untuk melakukan konfigurasi, dibuat sebuah file config.py.

Kondisi 1 mengijinkan komputer klien untuk melakukan urutan ketukan pada *port* diantara 1000-1999

```
def condition1(pkt):
  if pkt[tcp.TCP].dport >=1000 and pkt[tcp.TCP].dport < 2000 :
     return True
  else :
     return False
```

Kondisi 2 mengijinkan komputer klien untuk melakukan urutan ketukan pada *port* diantara 2000-2999

```
def condition2(pkt):
  if pkt[tcp.TCP].dport >=2000 and pkt[tcp.TCP].dport < 3000 :
     return True
  else :
     return False
```

Kondisi 3 mengijinkan komputer klien untuk melakukan urutan ketukan pada *port* diantara 3000-3999

```
def condition3(pkt):
  if pkt[tcp.TCP].dport >=3000 and pkt[tcp.TCP].dport < 4000 :
     return True
  else :
     return False
```

Jika ketukan berhasil maka komputer klien dapat membuka akses koneksi ssh

Untuk membuka koneksi maka urutan ketuk yaitu :

1. Ketukan 1 melalui *port* 1000-1999
2. Ketukan 2 melalui *port* 2000-2999
3. Ketukan 3 melalui *port* 3000-3999

Untuk menutup koneksi maka urutan ketuk dibalik yaitu :

1. Ketukan 1 melalui *port* 3000-3999
2. Ketukan 2 melalui *port* 2000-2999
3. Ketukan 3 melalui *port* 1000-1999

Terdapat empat bagian pada file config, yaitu:

1. Kondisi urutan *port* untuk masuk
2. Kondisi urutan *port* untuk keluar dari koneksi
3. Fungsi yang berisi perintah untuk “open *port*”
4. Fungsi yang berisi perintah untuk “close *port*”

Pada bagian openSSH ditentukan tiga urutan nomor *port* ketukan, dimana-mana masing-masing urutan dapat diantara *port* 1000-1999 secara bebas. Jika terjadi urutan ketukan terhadap ketiga *port* tersebut maka komputer target akan membuka *port port* tujuan sesuai dengan perintah iptables yang ditentukan.
Pada bagian close SSH ditentukan tiga urutan nomor *port* ketukan, dimana-mana masing-masing urutan dapat diantara *port* 2000-2999 secara bebas. Jika terjadi ketukan terhadap ketiga *port* tersebut secara berurutan maka komputer target akan menutup *port* tujuan sesuai dengan perintah iptables yang ditentukan.

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 1, Maret 2025, Pp. 720-738

Secara default, ketika Ubuntu pertama kali diinstal, remote access melalui SSH tidak diijinkan. Perintah untuk mengaktifkan SSH di Ubuntu menggunakan command berikut. Langkah-langkah berikut sebagai root atau pengguna dengan hak sudo untuk menginstal dan mengaktifkan SSH di sistem Ubuntu:

sudo apt update
sudo apt install openssh-server
sudo systemctl status ssh

Pada terminal akan memberi tahu bahwa layanan sedang berjalan dan diaktifkan untuk memulai saat boot system.

ssh.service - OpenBSD Secure Shell server
Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
Active: active (running)

Ubuntu firewall dengan konfigurasi firewall yang disebut UFW. Jika firewall diaktifkan pada sistem, maka perlu untuk membuka *port* SSH dengan perintah :

sudo ufw allow ssh

Pada *port knocking* dengan urutan yang sesuai dengan strategi yang yang telah ditentukan maka, pada system akan memerintahkan untuk membuka *port* maupun untuk menutup *port*. Konfigurasi perintah untuk membuka *port* sesuai urutan ketuk, menggunakan perintah berikut :

sudo iptables -A INPUT -s %s -p tcp --dport 22 -j ACCEPT ip-address

Perintah tersebut mengijinkan koneksi ssh ke server ssh melalui ip address client

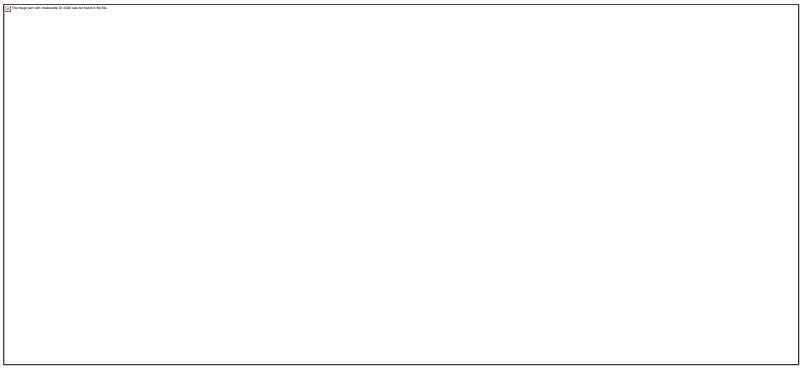
Konfigurasi perintah untuk menutup *port* sesuai urutan ketuk, menggunakan perintah berikut :

sudo iptables -A INPUT -s %s -p tcp --dport 22 -j DROP ip-address

Perintah tersebut menutup koneksi ssh ke server ssh melalui ip address client

C. Pengujian Program

Merupakan Langkah-langkah yang dilakukan untuk menguji program pada komputer target apakah dapat bekerja dengan baik untuk mendeteksi adanya urutan ketukan dengan benar. Sebelum memulai untuk menjalankan program deteksi ketuk hal penting yang perlu diketahui adalah mengetahui ip address komputer target, dengan mengetikkan perintah pada terminal .



Gambar 10 Ipconfig

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 1, Maret 2025, Pp. 720-738

Dan diperoleh ip address komputer target adalah 192.168.1.22

Perintah untuk menjalan kode program python pada komputer target dengan mengetikkan perintah, sebagai berikut :

Gambar 11 sudo python3 knock.py

Info “tekan enter untuk keluar” menjelaskan bahwa program deteksi ketuk telah berjalan pada komputer target.

Langkah berikutnya adalah mengatur konfigurasi skenario *port* dengan urutan 1200,2105,3301. Dimana :
1.Ketukan 1 melalui *port* 1200
2.Ketukan 2 melalui *port* 2105
3.Ketukan 3 melalui *port* 3301

Dilanjutkan dengan menjalankan program ketuk pada komputer klien yang bertujuan untuk request open connection, konfigurasi *port* pada kode program python sebagai berikut :

```
port_ketuk = [1200,2105,3301]
```

Gambar 12 Port ketukt

Program *port knocking* pada sisi client dengan mengatur konfigurasi dengan *ip address server* 192.168.1.22
knocking port 1 1200
knocking port 2 2105
knocking port 3 3301

- ***Knocking port* 1**

knocking port 1 *port* : 1200, sisi client akan berupaya melakukan koneksi ke ip address 192.168.1.22 : 1200, intruksi program pada sisi client dapat dijelaskan sebagi berikut :

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.settimeout(TIMEOUT_SECS)
s.connect_ex((ip, port))

dimana :
ip : 192.168.1.22
port : 1200
TIMEOUT_SECS : 0.5

<socket.socket fd=436, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('0.0.0.0', 51541), raddr=('192.168.1.22', 1200)>
Time out memberikan indikasi batasan waktu koneksi, jika Upaya koneksi ke server tidak dapat dilakukan melebihi batas waktu 0.5 detik maka upaya koneksi akan dihentikan.

- ***Knocking port* 2**

knocking port 1 *port* : 2105, sisi client akan berupaya melakukan koneksi ke ip address 192.168.1.22 : 2105, intruksi program pada sisi client dapat dijelaskan sebagi berikut :

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.settimeout(TIMEOUT_SECS)

s.connect_ex((ip, port))

dimana :
ip : 192.168.1.22
port : 2105
TIMEOUT_SECS : 0.5

<socket.socket fd=436, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('0.0.0.0', 51542), raddr=('192.168.1.22', 2105)>

Time out memberikan indikasi batasan waktu koneksi, jika Upaya koneksi ke server tidak dapat dilakukan melebihi batas waktu 0.5 detik maka upaya koneksi akan dihentikan.

- ***Knocking port* 3**

knocking port 1 *port* : 3301, sisi client akan berupaya melakukan koneksi ke ip address 192.168.1.22 : 3301, intruksi program pada sisi client dapat dijelaskan sebagi berikut :

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.settimeout(TIMEOUT_SECS)
s.connect_ex((ip, *port*))

dimana :
ip : 192.168.1.22
port : 3301
TIMEOUT_SECS : 0.5

<socket.socket fd=436, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('0.0.0.0', 51543), raddr=('192.168.1.22', 3301)>

Time out memberikan indikasi batasan waktu koneksi, jika Upaya koneksi ke server tidak dapat dilakukan melebihi batas waktu 0.5 detik maka upaya koneksi akan dihentikan.

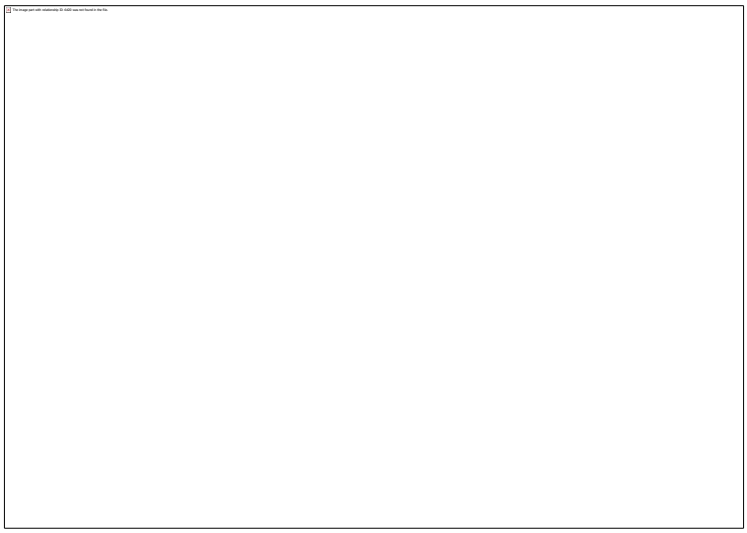
Proses menjalankan program ketuk pada klien ditunjukkan pada gambar berikut :

Gambar 13 Proses Ketuk

Proses tersebut menjelaskan proses pada ip 192.168.60.189 pada *port* 1200, 2105 dan 3301.

Karena urutan *port* sesuai dengan strategi yang ditentukan pada komputer target maka komputer target dapat mendeteksi ketukan urutan *port* dengan benar.

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 1, Maret 2025, Pp. 720-738



Gambar 14 Open Port

Pada deteksi ketuk pada sisi server Ubuntu dapat dijelaskan sebagai berikut :

1. Server mendeteksi ketukan pertama yang diterima dari client, jika *port* ketuk client berada pada range strategi 1, maka ketukan pertama diindikasi kan sebagai ututan ketukan yang benar. Kode program server untuk mendeteksi *port* ketuk dijelaskan sebagai berikut :

```
if pkt[tcp.TCP].dport >=1000 and pkt[tcp.TCP].dport < 2000 :
    return True
else :
    return False
```

   Pada kode program menjelaskan jika nilai *port* ketuk berada diantara 1000 hingga 1999 maka sebagai ketukan 1 yang benar, jika tidak berada pada range maka diindikasikan sebagai ketukan yang salah.
2. Server mendeteksi ketukan kedua yang diterima dari client, jika *port* ketuk client berada pada range strategi 2, maka ketukan kedua diindikasi kan sebagai ututan ketukan yang benar. Kode program server untuk mendeteksi *port* ketuk dijelaskan sebagai berikut :

```
if pkt[tcp.TCP].dport >=2000 and pkt[tcp.TCP].dport < 3000 :
    return True
else :
    return False
```

   Pada kode program menjelaskan jika nilai *port* ketuk berada diantara 2000 hingga 2999 maka sebagai ketukan 2 yang benar, jika tidak berada pada range maka diindikasikan sebagai ketukan yang salah.
3. Server mendeteksi ketukan ketiga yang diterima dari client, jika *port* ketuk client berada pada range strategi 3, maka ketukan ketiga diindikasi kan sebagai ututan ketukan yang benar. Kode program server untuk mendeteksi *port* ketuk dijelaskan sebagai berikut :

```
if pkt[tcp.TCP].dport >=3000 and pkt[tcp.TCP].dport < 4000 :
    return True
else :
    return False
```

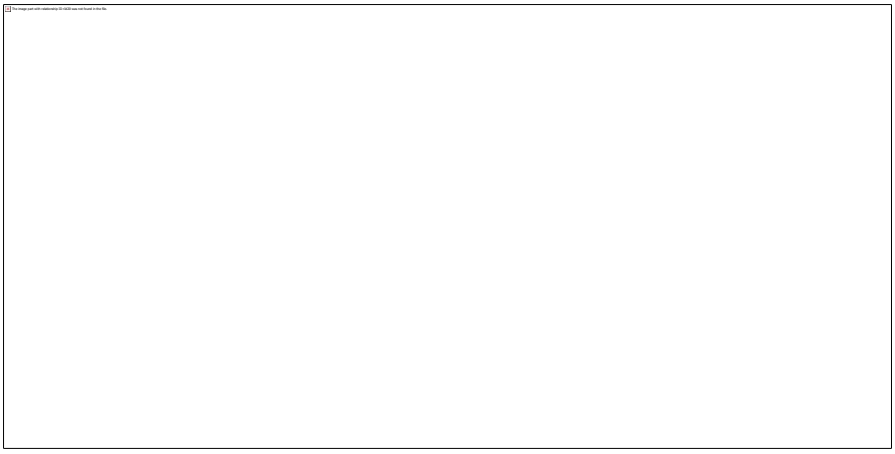
   Pada kode program menjelaskan jika nilai *port* ketuk berada diantara 3000 hingga 3999 maka sebagai ketukan 3 yang benar, jika tidak berada pada range maka diindikasikan sebagai ketukan yang salah.

Jika urutan ketuk sesuai dengan batasan nilai *port* yang telah ditentukan maka, program mengijinkan untuk membuka *port* 22 bagi client 192.168.1.5. Jika strategi sesuai maka program akan memberikan perintah untuk mengatur rule pada ip tables dengan perintah "*sudo iptables -A INPUT -s %s -p tcp --dport 22 -j ACCEPT 192.168.1.5*". Dengan demikian client 192.168.1.5 dapat melakukan remote access ke komputer server melalui koneksi ssh.

Uji validasi untuk memastikan bahwa client dapat melakukan koneksi ke remote server ssh ditunjukkan pada pengujian berikut. Pada uji koneksi ssh tool yang digunakan adalah putty. Uji koneksi pertama kali dilakukan uji

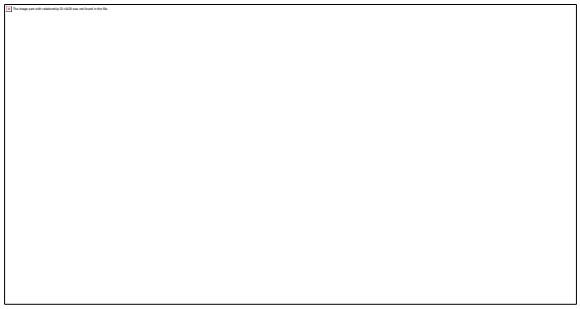
JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 1, Maret 2025, Pp. 720-738

sebelum proses ketuk dilakukan, Langkah-langkah koneksi ssh yaitu :



Gambar 15 Konfigurasi Koneksi

Selanjutnya klik tombol open koneksi, hasil koneksi ditunjukkan pada gambar berikut.



Gambar 16 Kegagalan Koneksi SSH

Dari hasil pengujian koneksi menjelaskan jika *port* 22 tertutup sehingga client 192.168.1.5 tidak dapat melakukan koneksi ssh.

Uji koneksi kedua kali dilakukan uji setelah proses ketuk dilakukan, hasil koneksi ssh ditunjukkkan pada gambar berikut.

Gambar 17 Koneksi SSH terbuka

Langkah selanjutnya adalah menjalankan program ketuk pada komputer klien, dengan mengatur *port knock* dengan urutan terbalik yang ditunjukkan pada kode program python sebagai berikut.

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 1, Maret 2025, Pp. 720-738

```
port_ketuk = [3301,2105,1200]
```

Gambar 18 port ketuk

Dimana :
1. Ketukan 1 melalui *port* 3301
2. Ketukan 2 melalui *port* 2105
3. Ketukan 3 melalui *port* 1200

Proses menjalankan program ketuk pada klien ditunjukkan pada gambar berikut.

Gambar 19 Proses Ketuk

Proses tersebut menjelaskan proses pada ip 192.168.60.189 pada *port* 1200, 2105 dan 3301.

Karena urutan *port* sesuai dengan strategi yang ditentukan pada komputer target maka komputer target dapat mendeteksi ketukan urutan *port* dengan benar.

Gambar 20 Close Port

Pada deteksi ketuk pada sisi server Ubuntu dapat dijelaskan sebagai berikut :

1. Server mendeteksi ketukan pertama yang diterima dari client, jika *port* ketuk client berada pada range strategi 1, maka ketukan pertama diindikasi kan sebagai ututan ketukan yang benar. Kode program server untuk mendeteksi *port* ketuk dijelaskan sebagai berikut.

```
if pkt[tcp.TCP].dport >=3000 and pkt[tcp.TCP].dport < 4000 :
    return True
 else :
    return False
```

   Pada kode program menjelaskan jika nilai *port* ketuk berada diantara 2000 hingga 2999 maka sebagai ketukan 1 yang benar, jika tidak berada pada range maka diindikasikan sebagai ketukan yang salah
2. Server mendeteksi ketukan kedua yang diterima dari client, jika *port* ketuk client berada pada range strategi 2, maka ketukan kedua diindikasi kan sebagai ututan ketukan yang benar. Kode program server untuk mendeteksi *port* ketuk dijelaskan sebagai berikut.

```
if pkt[tcp.TCP].dport >=2000 and pkt[tcp.TCP].dport < 3000 :
    return True
 else :
    return False
```

   Pada kode program menjelaskan jika nilai *port* ketuk berada diantara 1000 hingga 1999 maka sebagai ketukan 2 yang benar, jika tidak berada pada range maka diindikasikan sebagai ketukan yang salah
3. Server mendeteksi ketukan ketiga yang diterima dari client, jika *port* ketuk client berada pada range strategi 3, maka ketukan ketiga diindikasi kan sebagai ututan ketukan yang benar. Kode program server untuk

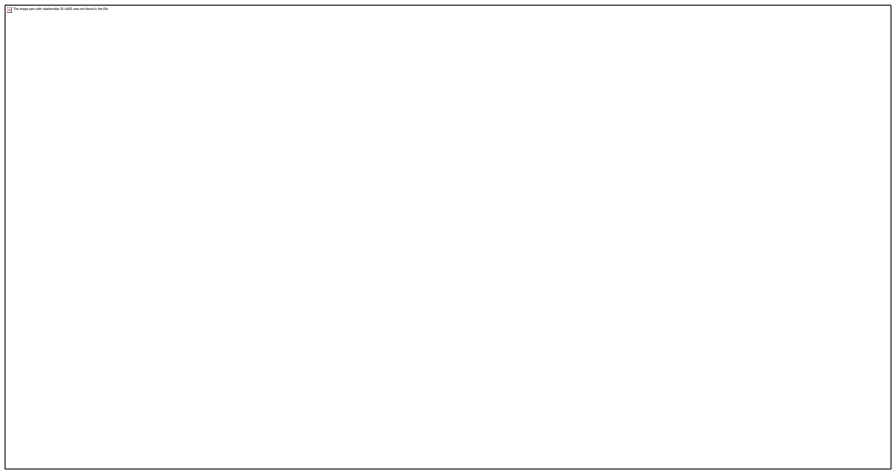
mendeteksi *port* ketuk dijelaskan sebagai berikut.

```
if pkt[tcp.TCP].dport >=1000 and pkt[tcp.TCP].dport < 2000 :
    return True
 else :
    return False
```

Pada kode program menjelaskan jika nilai *port* ketuk berada diantara 3000 hingga 3999 maka sebagai ketukan 3 yang benar, jika tidak berada pada range maka diindikasikan sebagai ketukan yang salah

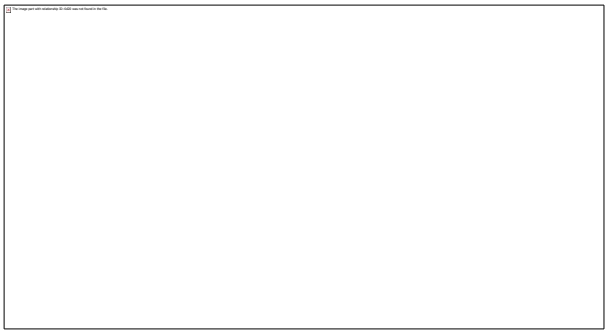
Jika urutan ketuk sesuai dengan batasan nilai *port* yang telah ditentukan maka, program mengijinkan untuk membuka *port* 22 bagi client 192.168.1.5. Jika strategi sesuai maka program akan memberikan perintah untuk mengatur rule pada ip tables dengan perintah "*sudo iptables -A INPUT -s %s -p tcp --dport 22 -j DROP 192.168.1.5*". Dengan demikian client 192.168.1.5 tidak dapat melakukan remote access ke komputer server melalui koneksi ssh, karena *port* ditutup bagi client 192.168.1.5.

Uji validasi untuk membuktikan bahwa *port* 22 telah tertutup dilakukan menggunakan tool putty.



Gambar 21 Uji validasi koenksi ssh

Jika koneksi telah di drop oleh server melalui pengaturan rule ip tables maka hasil koneksi ssh akan ditunjukkan pada gambar berikut.



Gambar 22 Uji validasi koenksi ssh yang tertutup

Pengujian serangan ke ubuntu server, dilakukan dengan mengatur *port knock* secara random sehingga diperoleh susunan *port* sebagai berikut.

```
[1301,2105,2200]
```

Gambar 23 Ketukan random

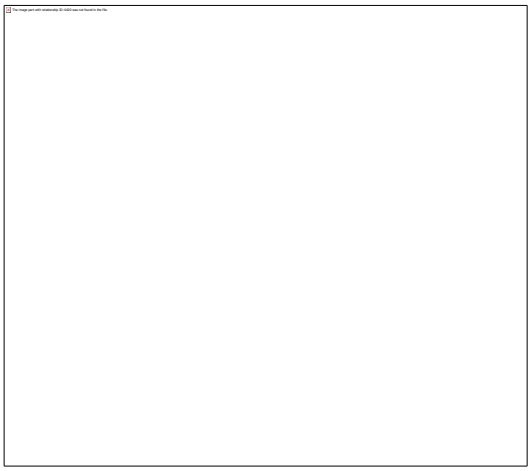
Saat proses ketuk ke server menggunakan susunan *port* tersebut maka pada komputer penyerang diperoleh informasi proses ketuk sebagai berikut.

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 1, Maret 2025, Pp. 720-738

Ketuk 1 : 192.168.1.18 1301
Ketuk 2 : 192.168.1.18 2105
Ketuk 3 :192.168.1.18 2200

Interval waktu tiap ketuk diatur pada timeout 0.5 detik

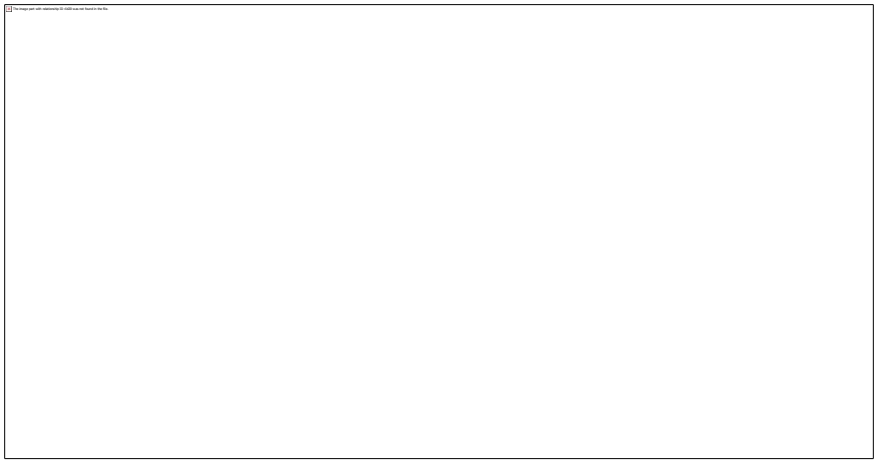
Hasil deteksi ketuk yang dijalankan pada komputer server akan menampilkan log info yang menjelaskan bahwa komputer server tidak mengijinkan untuk membuka *port* disebabkan karena urutan ketuk yang tidak sesuai. Informasi yang yang menunjukkan bahwa urutan ketuk tidak sesuai ditunjukkan pada log info berikut ini.



Gambar 24 Log Info Tidak Sesuai

Pengujian kualitas jaringan suatu jaringan dilakukan dengan mencari nilai QoS (Quality of Services) adalah Kemampuan untuk memastikan pengiriman data penting atau dengan kata lain, serangkaian kriteria yang menentukan tingkat kepuasan dalam penggunaan suatu jaringan[19]. Parameter QoS yang digunakan adalah antara lain delay, throughput, jitter, dan packet loss. Software network analyzer yang digunakan untuk mendapatkan hasil analisa dari parameter tersebut adalah bmon merupakan software analyzer jaringan yang dapat diinstall pada Ubuntu.

Berikut tampilan GUI bmon pada proses analisis jaringan saat komputer klien melakukan send data ke komputer server untuk menjalankan proses *knocking*.



Gambar 25 Capture Traffic Menggunakan bmon

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 1, Maret 2025, Pp. 720-738

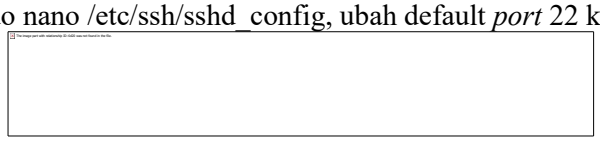
Gambar 26 Capture Traffic Menggunakan Bmon

Dalam implementasi untuk menangulangi adanya upaya serangan *bruteforce* pada *server ssh*, selain menggunakan *port knocking* seperti yang dijelaskan sebelumnya dapat digunakan alternatif lain, yaitu :

1. Mengubah default *port* ssh 22 ke *port* lain, 1022
2. Tentukan pengguna yang dapat login di layanan ssh
   Gunakan direktif AllowUsers dalam konfigurasi ssh untuk hanya mengizinkan pengguna atau IP tertentu. *Di /etc/ssh/sshd_config*, Dapat menentukan list pengguna yang diizinkan seperti ini:
   AllowUsers reza root@192.168.1.5 root@192.168.1.6
3. Batasi akses menggunakan aturan iptables
   Jika client selalu terhubung ke server Ubuntu dari alamat IP yang sama, dapat mematikan *port* 22 ke lainnya.
   iptables -A INPUT -p tcp -d 0/0 -s 192.168.1.5 --dport 22 -j ACCEPT
   iptables -A INPUT -p tcp -d 0/0 --dport 22 -j DROP
   Pengaturan lebih banyak IP ke dalam daftar putih menggunakan perintah berikut
   iptables -I INPUT -p tcp -m tcp --dport 22 -m state --state NEW -m hashlimit --hashlimit 1/min --hashlimit-mode srcip --hashlimit-name ssh -j ACCEPT

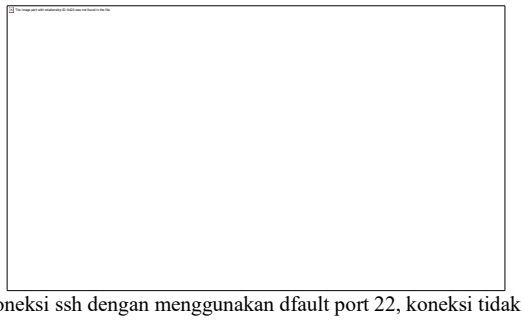
Pengujian pengaturan ssh *port* dengan mengubah *port* default 22 ke 1022, dilakukan Langkah-langkah berikut

1. sudo nano /etc/ssh/sshd_config, ubah default *port* 22 ke 1022



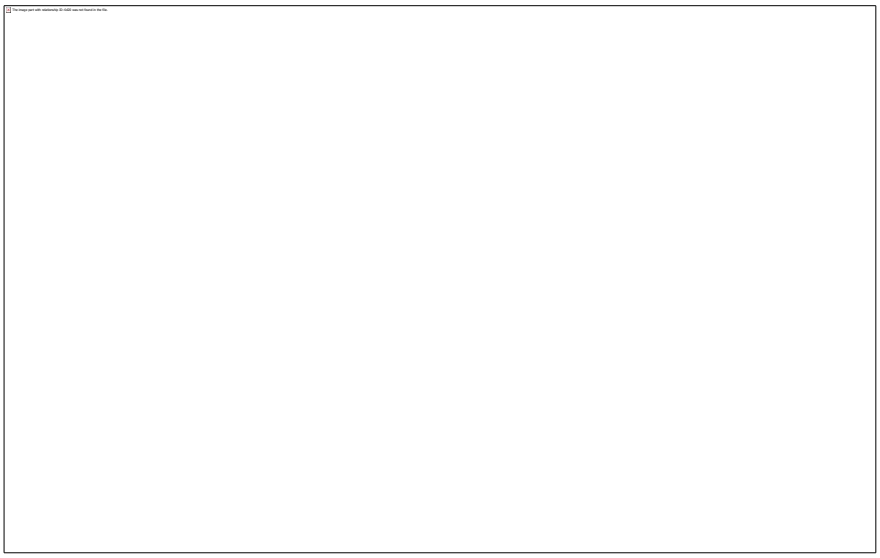
2. Simpan konfigurasi
3. Restart ssh service, systemctl restart ssh

Pengujian koneksi ssh dengan *port* baru, ditunjukkan pada gambar berikut :



Gambar 27 Uji koneksi ssh dengan menggunakan dfault port 22, koneksi tidak dapat dilakukan

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 1, Maret 2025, Pp. 720-738



Gambar 28 Uji koneksi ssh dengan menggunakan port10 22, koneksi berhasil

IV. KESIMPULAN

Pada hasil peneletian ini bertujuan melakukan pengembangan metode *portknocking*. *Port knocking* adalah teknik keamanan yang efektif untuk mengendalikan akses ke layanan jaringan dengan memerlukan urutan ketukan pada *port* tertentu sebelum akses diberikan. Hasil uji, dilakukan penggunaan *port knocking* dinamis yaitu menambahkan rule baru dengan urutan ketuk yang dapat berubah. Ketukan masuk *port* yaitu ketukan 1, ketukan 2, ketukan 3 dan untuk keluar *port* yaitu ketukan 3, ketukan 2, ketukan 1. Dengan range *port* yaitu ketukan 1 dari range 1000-1999, ketukan 2 dari range 2000-2999, dan ketukan 3 dari range 3000-3999. Lebih jelasnya, jika susuan *port* 1200,2105 dan 3105 yang dikirimkan komputer client ke komputer server maka server mengijinkan *port* 22 terbuka bagi client. Sebaliknya, susuan ketuk dengan urutan 3105, 2105 dan 1200 yang dikirimkan komputer client ke komputer server. Maka komputer server akan menutup komunikasi ssh pada *port* 22. Akses yang sudah ditutup *port knocking* dapat dilihat dengan menggunakan remote access yaitu Putty. Putty adalah Sebuah program open source yang memungkinkan penggunaan protokol jaringan seperti SSH, Telnet, dan Rlogin. Protokol ini dapat digunakan untuk menjalankan sesi remote pada komputer melalui jaringan, baik itu LAN maupun internet[20]. Berdasarkan hasil penelitian, Putty akan memblokir koneksi ssh. Penggunaan dinamis *port* dapat meningkatkan keamanan yang memiliki fleksibilitas lebih baik, karena *knocking port* yang dinamis dapat menyediakan banyak alternatif *port* yang terbuka. Pemanfaatan *port knocking* dapat meningkatkan keamanan pada jaringan komputer. Penggunaan metode *port knocking* sangat membantu dalam dalam penanganan keamanan akses ke server ssh dan hanya bagi client yang berhak dengan skenario ketukan tertentu yang dapat melakukan akses ke server ssh. Sehingga penggunaan dinamis *port* dapat meningkatkan keamanan yang memiliki fleksibilitas lebih baik, karena *knocking port* yang dinamis dapat menyediakan banyak alternatif *port* yang terbuka dan penerapan *knocking port* dapat dilakukan dengan beberapa skenario ketukan yang dapat diatur oleh administrator pada komputer server.

DAFTAR PUSTAKA

[1] M. J. N. Yudianto, "Jaringan Komputer dan Pengertiannya," *Ilmukomputer.Com*, vol. Vol.1, pp. 1–10, 2014.
[2] Z. Munawar, M. Kom, and N. I. Putri, "Keamanan Jaringan Komputer Pada Era Big Data," *J. Sist. Informasi-J-SIKA*, vol. 02, no. 01, pp. 14–20, 2020.
[3] I. Sembiring, I. Widiasari, and S. D. Prasetyo, "Analisa dan Implementasi Sistem Keamanan Jaringan Komputer dengan Iptables sebagai Firewall Menggunakan Metode *Port Knocking*," *J. Inform.*, vol. 5, no. 2, 2011, doi: 10.21460/inf.2009.52.74.
[4] J. Al Amien, "Implementasi Keamanan Jaringan Dengan Iptables Sebagai Firewall Menggunakan Metode *Port Knocking*," *J. Fasilkom*, vol. 10, no. 2, pp. 159–165, 2020, doi: 10.37859/jf.v10i2.2098.
[5] D. Desmira and R. Wiryadinata, "Rancang Bangun Keamanan *Port* Secure Shell (SSH) Menggunakan Metode *Port Knocking*," *J. Ilmu Komput. dan Sist. Inf.*, vol. 5, no. 1, pp. 28–33, 2022, doi: 10.55338/jikomsi.v5i1.242.
[6] R. Ernawati, I. Ruslianto, and S. Bahri, "Implementasi Metode *Port Knocking* Pada Sistem Keamanan Server Ubuntu Virtual Berbasis Web Monitoring," *Coding J. Komput. dan Apl.*, vol. 10, no. 01, pp. 158–169, 2022, [Online]. Available: https://jurnal.untan.ac.id/index.php/jcskommipa/article/view/54226
[7] A. Zidan, K. Amin, and T. Ghanem, "Enhanced User Authentication Based on Dynamic *Port Knocking* Technique," *IJCI. Int. J. Comput. Inf.*, vol. 8, no. 2, pp. 115–124, 2021, doi: 10.21608/ijci.2021.207854.
[8] P. D. Oktaviansyah, "Penerapan Sistem Pengamanan *Port* pada Mikrotik Menggunakan Metode *Port Knocking*," *NetPLG J. Netw. Comput. Appl.*, vol. 1, no. 2, pp. 13–24, 2022.
[9] Mursyidah *et al.*, "Analysis and implementation of the *Port Knocking* method using Firewall-based Mikrotik RouterOS," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 536, no. 1, 2019, doi: 10.1088/1757-899X/536/1/012129.
[10] A. Z. Mardiansyah, Y. M. Abdussyakur, and A. H. Jatmika, "OPTIMASI *PORT KNOCKING* DAN HONEYPOT MENGGUNAKAN IPTABLES SEBAGAI KEAMANAN JARINGAN PADA SERVER (*Port Knocking* and Honeypot Optimization using IPTables for Serverrs Network Security)," vol. 3, no. 2, 2021, [Online]. Available: http://jtika.if.unram.ac.id/index.php/JTIKA/
[11] R. Iqromullah, K. Khairil, and E. Suryana, "Security System Implementation And Monitoring Networks At Sma N 10 City Of Bengkulu," *J. Media Comput. Sci.*, vol. 2, no. 2, pp. 153–168, 2023, doi: 10.37676/jmcs.v2i2.4431.
[12] N. Dwipoyono, K. Khairil, and A. Sudarsono, "Penerapan Firewall Pada Sistem Keamanan Jaringan Komputer Di Sekolah Smk Negeri 5 Seluma," *J. Media Infotama*, vol. 19, no. 2, pp. 454–464, 2023, doi: 10.37676/jmi.v19i2.4355.
[13] B. Cahya, F. Rizki, A. Sutiyo, Y. El Saputra, and M. Elfarizi, "Implementasi Firewall Pada Mikrotik Untuk Keamanan Jaringan," *J. JOCOTIS-Journal Sci. Inform. Robot. E*, vol. 1, no. 2, pp. 63–80, 2023, [Online]. Available: https://jurnal.ittc.web.id/index.php/jct/
[14] D. Novianto, L. Tommy, and Y. S. Japriadi, "Implementation of a Network Security System Using the Simple *Port Knocking* Method on a Mikrotik-Based Router Implementasi Sistem Keamanan Jaringan Menggunakan Metode Simple *Port Knocking* Pada Router Berbasis Mikrotik," vol. 1, no. 2, pp. 407–417, 2021.
[15] W. Farhan Fatoni, A. Hidayat, P. Studi Ilmu Komputer, and F. Ilmu Komputer, "Implementasi Sistem Keamanan Jaringan Komputer Dengan Metode *Port Knocking* Pada Lkp Surya Komputer," *J. Mhs. Ilmu Komput.*, vol. 03, no. 01, 2022.
[16] Z. Amir, S. Syaifuddin, and D. Risqiwati, "Implementasi Asymmetric Encryption Rsa Pada *Port Knocking* Ubuntu Server Menggunakan *Knock*d Dan Python," *J. Repos.*, vol. 2, no. 6, p. 787, 2020, doi: 10.22219/repositor.v2i6.270.
[17] J. A. Raval and S. Johnson, "*Port Knocking*- An Additional Layer of Security for SSH and HTTPS".
[18] A. P. A. Kusuma and Asmunin, "Implementasi Simple *Port Knocking* Pada Dynamic Routing (Ospf) Menggunakan Simulasi Gns3," *J. Manaj. Inform.*, vol. 5, no. 2, pp. 7–17, 2016.
[19] I. Nurrobi, K. Kusnadi, and R. Adam, "PENERAPAN METODE QoS (QUALITY OF SERVICE) UNTUK MENGANALISA KUALITAS KINERJA JARINGAN WIRELESS," *J. Digit*, vol. 10, no. 1, p. 47, 2020, doi: 10.51920/jd.v10i1.155.
[20] S. Dwiyatno, E. Rachmat, A. P. Sari, and O. Gustiawan, "Implementasi Virtualisasi Server Berbasis Docker Container," *PROSISKO J. Pengemb. Ris. dan Obs. Sist. Komput.*, vol. 7, no. 2, pp. 165–175, 2020, doi: 10.30656/prosisko.v7i2.2520.