JUSTIN
Jurnal Sistem dan Teknologi Informasi
p-ISSN : 2460-3562 / e-ISSN : 2620-8989
DOI: 10.26418/justin.v12i3.79617
Vol. 12, No. 3, Juli 2024

Penggunaan *Snort* dan *Fail2ban* sebagai IDS untuk Mengatasi *Brute Force Attack* dengan Notifikasi Telegram: Studi Kasus pada Institusi XYZ

Riska Kurniyanto Abdullah[1], Muhammad Thariq Fudhail[2], Syamsul Mujahidin[3]

Program Studi Informatika Jurusan Matematika dan Teknologi Informasi,
Institut Teknologi Kalimantan
Jalan Soekarno-Hatta KM.15, *Karang Joang*,
Kec.Balikpapan Utara, *Balikpapan*, *Kalimantan Timur 76128*
[1]riska.abdullah@lecturer.itk.ac.id
[2]11191055@student.itk.ac.id
[3]syamsul@lecturer.itk.ac.id

Abstrak

Dengan kebutuhan ketergantungan pada jaringan komputer meningkat, hal ini tentu menuntut sistem yang lebih aman dari segi teknologi informasi. Ancaman utama seperti *Brute Force Attack* bisa diatasi menggunakan *firewall*, tetapi *firewall* sering kali sulit dikonfigurasi, mahal, dan terbatas dalam pencegahan. Penelitian ini mengusulkan penggunaan *Intrusion Detection System* (IDS) seperti *Snort* dan *Fail2Ban* yang dikombinasikan dengan *honeypot* untuk meningkatkan keamanan. Melalui simulasi berbasis SPDLC dan pengujian serangan, terungkap bahwa kombinasi alat ini, jika ditempatkan dengan benar, dapat mendeteksi 100% serangan dengan waktu respons yang cepat. Walaupun demikian, konfigurasi yang salah dapat mengurangi efektivitas dan meningkatkan kemungkinan tidak ada alarm hingga 60%, dan juga dengan kejadian *false alarm*. Hasil menunjukkan bahwa IDS bisa menjadi solusi efektif, terutama ketika *firewall* yang baik tidak tersedia.

Kata kunci: *Intrusion detection system, Security Policy Development Life Cycle (SPDLC), Brute Force Attack, false alarm, Snort, File2Ban, Honeypot*

The use of Snort and Fail2ban as IDS to overcome Brute Force Attack with Telegram notification: Case study at XYZ Institute

Abstract

With the need for dependence on computer networks increasing, this certainly demands a more secure system in terms of information technology. Major threats such as brute force attacks can be addressed using firewalls, but firewalls are often difficult to configure, expensive, and limited in prevention. This research proposes the use of intrusion detection systems (IDS) such as Snort and Fail2Ban combined with honeypots to enhance security. Through SPDLC-based simulation and attack testing, it is revealed that this combination of tools, if properly deployed, can detect 100% of attacks with a fast response time. However, incorrect configuration can reduce the effectiveness and increase the probability of no alarm by 60%, while also having a low incidence of false alarms. These results indicate that IDS can be an effective solution, especially when a good firewall is not available.

Keywords: *Intrusion detection system, Security Policy Development Life Cycle (SPDLC), Brute Force Attack, false alarm, Snort, File2Ban, Honeypot*

I. PENDAHULUAN

Dalam era perkembangan teknologi informasi dan komunikasi yang pesat, jaringan komputer telah menjadi infrastruktur krusial bagi berbagai institusi, perusahaan, dan organisasi. Seiring dengan meningkatnya ketergantungan terhadap jaringan komputer, masalah keamanan menjadi isu yang mendesak dan memerlukan perhatian serius. Institusi-institusi modern bergantung pada sistem informasi kompleks yang terhubung dengan jaringan internal dan eksternal, sehingga kompleksitas infrastruktur yang digunakan semakin meningkatkan ancaman keamanan, termasuk *Brute Force Attack* [1]. Oleh karena itu, perlindungan terhadap sistem informasi dan jaringan komputer menjadi hal yang sangat penting. Keamanan Teknologi Informasi (TI) menjadi fokus utama, dengan upaya menjaga aset TI dan meminimalisir risiko serta ancaman keamanan. Pada are khusus yaitu Keamanan web server adalah aspek penting untuk melindungi data sensitif dan menjaga integritas sistem informasi. Kegagalan dalam mengamankan server web dapat menyebabkan pelanggaran data, yang berpotensi mengakibatkan kerugian finansial yang besar dan kerusakan reputasi. Mengidentifikasi dan memperbaiki kerentanan pada web

aplikasi sangat vital untuk menjamin keamanan informasi. Web aplikasi yang rentan menjadi sasaran empuk bagi penyerang, yang bisa mengontrol aplikasi dan sistem operasi, mengakses data pribadi, dan mengganggu operasional bisnis [2].

Keamanan TI melibatkan aspek-aspek penting seperti tata kelola keamanan TI, audit manajemen keamanan TI, metode keamanan informasi dan *vulnerability assessment*, serta evaluasi tata kelola keamanan TI. Dalam konteks ini, keamanan sistem informasi menjadi krusial, terutama dalam sektor *Smart City* yang mengandalkan TI untuk sektor pendidikan, kesehatan, tata kelola kota, dan berbagai kegiatan lainnya[3]. Keamanan TI yang baik tidak hanya meningkatkan efisiensi pelayanan, mobilitas, dan tata kelola pemerintahan, tetapi juga memainkan peran signifikan saat server harus bekerja secara masif, menangani permintaan dari pengguna dalam skala besar, atau melakukan tugas komputasi tingkat tinggi. Untuk melawan ancaman keamanan, penggunaan *firewall* telah umum, meskipun *firewall* memiliki kelemahan seperti konfigurasi rumit, biaya tinggi, *false positives*, dan proteksi yang terbatas[4]. Oleh karena itu, penelitian ini mengusulkan penggunaan *Intrusion Detection System* (IDS) sebagai alternatif untuk meningkatkan keamanan TI.

IDS dirancang untuk mendeteksi serangan keamanan pada jaringan komputer atau sistem informasi dengan memonitor lalu lintas jaringan atau aktivitas sistem. Dalam konteks ini, penelitian akan menguji efektivitas dan performa IDS *open-source* seperti snort dan fail2ban, yang diintegrasikan dengan *honeypot* dalam simulasi jaringan[5]. Kesulitan seperti *false alarms* dan *no alarms* dalam IDS menjadi perhatian utama, dan penelitian ini juga akan mengevaluasi performa cowrie sebagai *honeypot* untuk mengatasi masalah tersebut. Dengan demikian, penelitian ini bertujuan untuk memberikan pemahaman yang lebih mendalam tentang solusi alternatif untuk keamanan TI, mengidentifikasi topologi yang efektif, dan memberikan kontribusi terhadap pengembangan sistem keamanan yang lebih andal dan efisien[6].

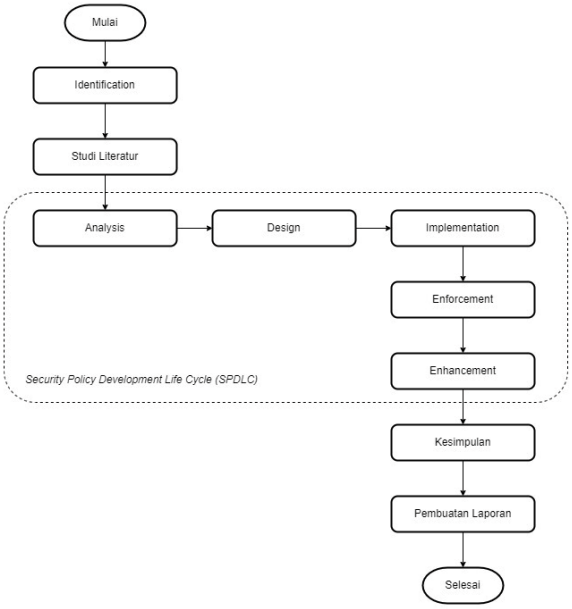
Pada penelitian sebelumnya, Keamanan Jaringan dengan Cowrie *Honeypot* dan *Snort Inline-Mode* sebagai *Intrusion Prevention System* memiliki hasil penelitian dimana uji parameter integritas secara keseluruhan IPS *snort inline-mode* lebih unggul dibandingkan *cowrie honeypot*. Mengacu kepada uji parameter integritas secara keseluruhan IPS *snort inline-mode* lebih direkomendasikan untuk digunakan dalam sistem keamanan jaringan dibandingkan *cowrie honeypot*[7].

Adapun penelitian lain yang dilakukan oleh Mutaqin pada tahun 2016 dengan judul "Rancang Bangun Sistem Monitoring Keamanan Jaringan Prodi Teknik Informatika Melalui SMS Alert dengan Snort" yang membahas performa snort dalam menghadapi berbagai bentuk penyerangan pada jaringan serta penggunaan SMS *alert*[8].

Berdasarkan penelitian sebelumnya, disimpulkan bahwa snort dapat mendeteksi serangan *Ping Attack* (ICMP *Traffic*), DOS/DDOS *Attack*, *Port Scanning* sesuai dengan *rules* yang dibuat, selain itu serangan yang terjadi dapat di-*monitoring* secara *realtime* dengan adanya SMS *Gateway*. Dalam konteks penelitian ini, dapat disimpulkan bahwa temuan tersebut masih memiliki relevansi yang tinggi. Namun, penelitian ini memiliki perbedaan signifikan, dimana akan digunakan kombinasi Snort sebagai *Intrusion Detection System* (IDS) dan Fail2Ban sebagai *Intrusion Prevention System* (IPS), bersama dengan Cowrie sebagai *Honeypot*. Penekanan pada efektivitas kombinasi *tools* dalam mendeteksi *Brute Force Attack*, mengurangi *false alarm*, dan memberikan tingkat keamanan maksimal menjadi fokus utama penelitian ini. Target utama yaitu untuk mendapatkan posisi yang tepat pada topologi dan formasi yang sesuai dalam peletakan tools fail2ban, snort dan honeypot.

.

II. METODOLOGI PENELITIAN



Gambar 1 Metode Penelitian

Pada Gambar 1 terdapat diagram alir yang memiliki tahapan pada penelitian ini. Tahapan tersebut dimulai dengan tahapan *identification*, studi literatur, kemudian masuk pada tahapan pengembangan *security policy development life cycle (SPDLC)* yang terdiri dari beberapa tahap yaitu tahap *analysis*, *design*, *implementation*, *enforcement*, dan *enhancement*. Diakhiri dengan membuat kesimpulan. Secara khusus proyek ini menerapkan Security Policy Development Life Cycle dalam melakukan implementasi aturan keamanan. Berikut adalah langkah-langkah spesifik yang diambil dalam setiap tahap.

A. Identification

Tahap awal penelitian ini melibatkan identifikasi permasalahan melalui analisis sistem informasi dan jaringan di Institusi XYZ. Wawancara dengan administrator dilakukan untuk menentukan apakah sistem informasi tersebut merupakan aset penting dan menentukan skala CIA berdasarkan Tabel 1.

TABEL 1
SKALA CIA

Skala Nilai	Kerahasiaan	Integritas	Ketersediaan
Rendah	Informasi tidak sensitif atau umum	Perubahan minor tidak signifikan	Gangguan sementara tidak berdampak besar
Sedang	Informasi agak sensitif	Perubahan dapat berdampak, tetapi bisa diperbaiki	Gangguan sementara dengan dampak moderat
Tinggi	Informasi sangat sensitif atau rahasia	Perubahan dapat berdampak serius	Gangguan berkepanjangan dengan dampak besar

Wawancara tersebut akan menentukan perlunya peningkatan kualitas keamanan di Institusi XYZ. Salah satunya, dengan menerapkan sistem deteksi snort di layer jaringan dan fail2ban di layer aplikasi. Untuk mengurangi risiko *false alarm* dan *human error*, akan digunakan sistem *honeypot* cowrie. Selain itu, informasi dari sistem deteksi akan diintegrasikan melalui aplikasi Telegram dengan bantuan SMS *gateway*.

Beberapa langkah yang diambil dalam tahap ini adalah:

1. Wawancara dengan Administrator Jaringan: Untuk memahami kebutuhan dan permasalahan yang ada.
2. Penentuan Skala CIA: Menggunakan tabel skala *Confidentiality, Integrity, Availability* (CIA) untuk menentukan tingkat keamanan yang diperlukan.
3. Analisis Risiko: Menentukan risiko keamanan yang mungkin dihadapi oleh sistem jaringan ITK.

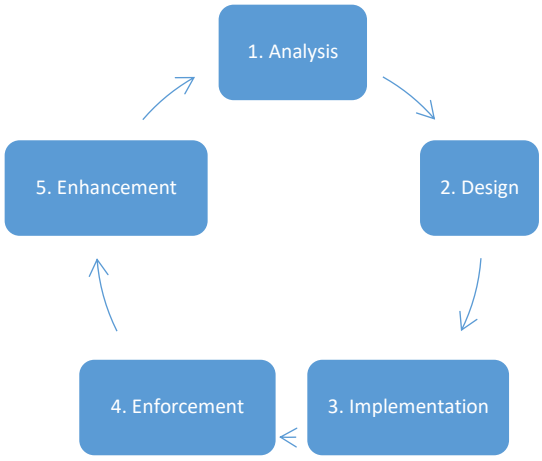
B. Studi Literatur

1. Keamanan Jaringan Komputer

Keamanan mencakup upaya untuk melindungi individu, kelompok, atau lingkungan dari risiko dan ancaman, termasuk keamanan siber yang fokus pada perlindungan terhadap serangan di dunia digital[8]. Keamanan jaringan komputer bertujuan mencegah akses tidak sah, mengantisipasi risiko fisik dan logika, serta menjaga aspek CIA (*Confidentiality, Integrity, Availability*) dalam sistem informasi[9].

2. *Security Policy Development Life Cycle* (SPDLC)

Security Policy Development Life Cycle (SPDLC) adalah metodologi untuk mengembangkan dan menerapkan keamanan yang komprehensif serta efektif. SPDLC mencakup berbagai konteks, seperti keamanan jaringan, nirkabel, dan sistem komputer dengan jaminan tinggi, dengan memberikan rekayasa keamanan berprosedur[10].



Gambar 2 *Security Policy Development Life Cycle*

SPDLC seperti pada Gambar 2 juga dapat diartikan sebagai model terstruktur yang menggambarkan proses dari pengembangan kebijakan keamanan dan mengintegrasikannya ke dalam *development lifecycle*[11].

3. *Intrusion Detection System* (IDS)

Intrusion detection adalah proses *monitoring* dan analisis peristiwa dalam sistem komputer atau jaringan untuk menemukan tanda-tanda upaya intrusi. IDS, baik *software* maupun *hardware*, otomatisasi proses ini dan terbagi menjadi tiga kategori: *Signature-based Detection* (SD) menggunakan pola yang sesuai dengan serangan yang diketahui, *Anomaly-based Detection* (AD) memantau dan mempelajari perilaku normal untuk mendeteksi perbedaan signifikan, dan *Stateful Protocol Analysis* (SPA) menggunakan profil generik untuk protokol tertentu. IDS seperti snort, suricata, fail2ban, dan OSSEC memonitor dan menganalisis lalu lintas jaringan serta aktivitas sistem untuk mengidentifikasi serangan atau perilaku mencurigakan[12].

Snort adalah *software* IDS *open-source* yang dapat mendeteksi berbagai serangan melalui analisis lalu lintas dan pencatatan paket pada jaringan Protokol Internet (IP). Snort dapat diatur dalam tiga mode utama: *sniffer*, *packet logger*, dan deteksi intrusi jaringan, mampu membaca, mencatat, dan menganalisis lalu lintas sesuai aturan yang ditentukan[13].

Fail2ban, aplikasi pencegah *brute force*, memonitor aktivitas log server untuk mendeteksi pola perilaku mencurigakan dan memblokir alamat IP penyerang melalui *firewall*. Meskipun Fail2ban meningkatkan keamanan dengan mencegah serangan *brute force*, ia memiliki keterbatasan dan tidak dapat melindungi dari serangan jenis lain seperti DDoS atau SQL *injection*[14].

4. SMS *Gateway*

SMS *Gateway* memfasilitasi pengiriman dan penerimaan pesan SMS melalui jaringan telekomunikasi dengan pengembangan dalam berbagai bidang dan menggunakan bahasa pemrograman berbasis web[15]. Dengan kemampuan otomatis dan cepat, SMS *gateway* memungkinkan pengiriman pesan ke ratusan nomor yang terhubung dengan *database*, menghindari pengetikan

manual. Fitur-fitur seperti SMS interaktif, informasi berbasis permintaan, dan pengiriman terjadwal memperluas peluang interaksi dengan pengguna serta meningkatkan efisiensi pengiriman pesan[8].

5. *Honeypot*

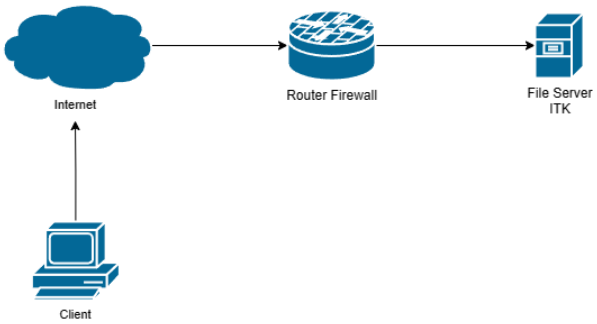
Honeypot adalah perangkat keamanan server yang menciptakan replika server palsu. Pengguna dapat memilih dari tiga jenis layanan: *Low Interaction Honeypot*, di mana pemilik server tetap memiliki kontrol penuh; *Medium Interaction Honeypot*, yang menciptakan sistem operasi palsu dan merekam informasi penyerang; serta *High Interaction Honeypot*, di mana server asli direplikasi sepenuhnya tanpa pengawasan aktif, menjaga keamanan server asli dari dampak serangan[16].

C. Analysis

Pada tahapan *analysis* data terdapat beberapa hal yang dilakukan yaitu melakukan analisis mengenai spesifikasi sistem yang akan dibangun serta spesifikasi sistem yang dibutuhkan mencakup perangkat keras(*hardware*) dan perangkat lunak(*software*). Pada tahapan ini yang akan dijelaskan yaitu:

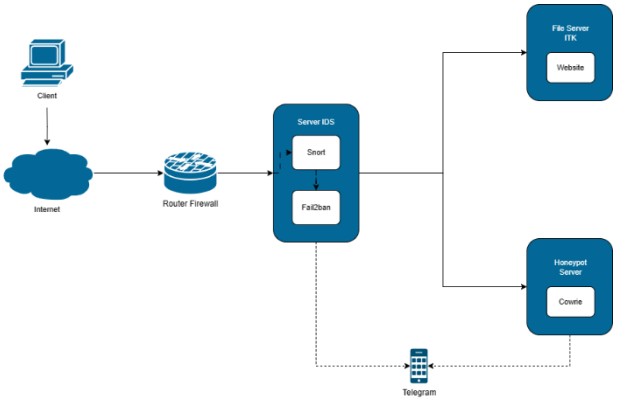
- Spesifikasi sistem yang akan dibangun, mencakup:
  - Sistem *Intrusion Detection System*, menjelaskan tentang jenis sistem IDS yang dibangun dalam sistem yang baru.
  - *Client*, sebagai sistem *client* yang terhubung langsung dengan jaringan dan bertugas sebagai sistem penyerangan dalam menguji sistem IDS.
- Spesifikasi perangkat keras yang digunakan, mencakup:
  - PC *Client*, yang digunakan untuk menjalankan sistem yang dibutuhkan dalam penelitian yaitu melakukan konfigurasi dan melakukan serangan menggunakan *tools*.
  - *Smartphone*, yang digunakan untuk menjalankan aplikasi Telegram sebagai sistem *SMS Gateway* dalam menerima dan memberikan informasi dari sistem IDS.
- Spesifikasi perangkat lunak yang digunakan, mencakup:
  - Sistem Operasi
  - *Tools Intrusion Detection System*, mencakup Snort dan Fail2Ban.
  - *Tools* Pengujian, berupa *tools* yang dapat melakukan serangan *Brute Force Attack*.
  - Cowrie, sebagai sistem *honeypot* yang menjadi *fake server* dalam menerima serangan selama pengujian.
  - *Tools* SMS *gateway*, berupa aplikasi telegram.

D. Design



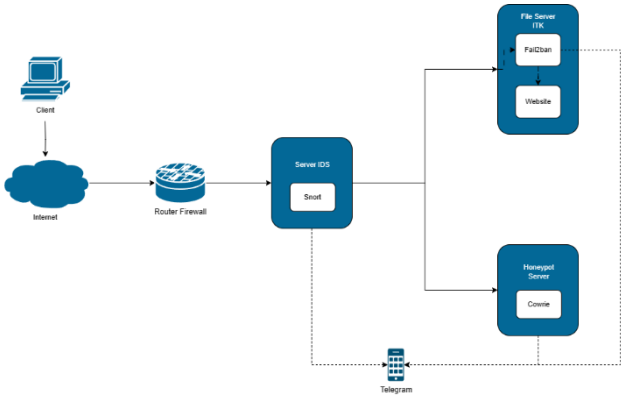
Gambar 3 Topologi Awal

Gambar 3 menunjukkan topologi awal dengan *firewall* di *router* sebagai satu-satunya sistem keamanan. Prosesnya melibatkan pengiriman permintaan *login* dari *client* ke *server* melalui jaringan internet. *Router* memeriksa akses *client* ke *server* sebelum *server* menerima dan memproses permintaan tersebut.



Gambar 4 Usulan Topologi Sistem 1

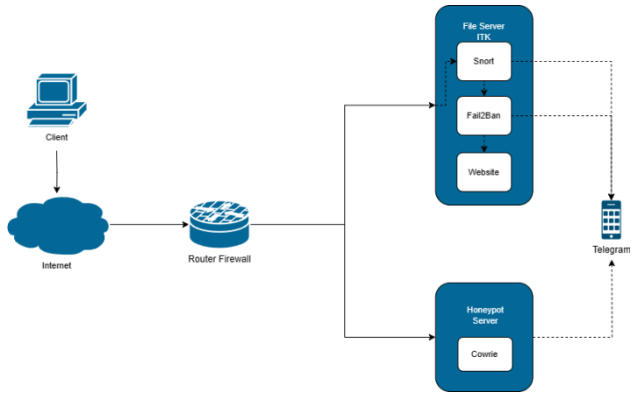
Gambar 4 menjelaskan alur permintaan *client* pada sistem jaringan Institusi XYZ. Permintaan login dikirim melalui *router* dan diperiksa oleh server IDS untuk perizinan dan deteksi tindakan mencurigakan. IDS mencatat ke log, dan notifikasi dikirim melalui Telegram jika terdeteksi tindakan mencurigakan. Jika tidak mencurigakan, permintaan dilanjutkan ke server Institusi XYZ. Rancangan topologi pertama ringan bagi server utama Institusi XYZ, meminimalkan dampak lonjakan permintaan. Meskipun konfigurasi mungkin kompleks, risiko *false alarm* dapat diminimalisir dengan keberadaan cowrie sebagai *honeypot*.



Gambar 5 Usulan Topologi Sistem 2

Topologi pada Gambar 5 menempatkan fail2ban pada server Institusi XYZ bersamaan dengan *website dummy*, berfungsi sebagai lapisan pelindung pada web server. Prosesnya melibatkan pengiriman permintaan login dari *client* ke server Institusi XYZ melalui *router*, diperiksa oleh IDS, dan mencatat tindakan mencurigakan. IDS yang mendeteksi tindakan mencurigakan akan mengirim notifikasi melalui Telegram. Jika fail2ban di server Institusi XYZ mendeteksi tindakan mencurigakan pada *website dummy*, notifikasi juga dikirimkan melalui Telegram.

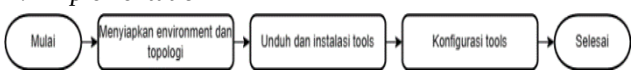
Usulan kedua mencapai keseimbangan dan efisiensi dengan Snort bekerja optimal di layer *network* dan fail2ban melindungi *website dummy* tanpa memberatkan server Institusi XYZ. Namun, perlu konfigurasi fail2ban yang teliti untuk mengatasi kekurangan seperti tidak terdeteksinya serangan dan kesulitan mengidentifikasi *false alarm*.



Gambar 6 Usulan Topologi Sistem 3

Topologi ketiga pada Gambar 6 melibatkan pemasangan IDS snort dan fail2ban pada *file* server Institusi XYZ untuk melindungi *file* server secara langsung. Prosesnya melibatkan permintaan *client* kepada server Institusi XYZ, yang kemudian diperiksa oleh snort di layer *network* server Institusi XYZ. Jika snort mendeteksi tindakan mencurigakan, notifikasi akan dikirimkan melalui Telegram. Selanjutnya, fail2ban di layer aplikasi akan memeriksa permintaan yang tidak terdeteksi oleh snort, memberikan notifikasi jika tindakan mencurigakan terbaca dalam *file* log fail2ban. Meskipun topologi ini dianggap meningkatkan keamanan, potensi lonjakan permintaan dari *client* dapat mempengaruhi kinerja server utama Institusi XYZ dan menyebabkan kegagalan server.

E. Implementation



Gambar 7 Tahap Implementasi

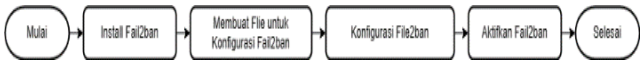
Gambar 7 dimulai dengan menyiapkan *environment* dan menyusun topologi sesuai dengan tahap *design*. Langkah berikutnya melibatkan pengunduhan, instalasi *tools*, serta konfigurasi komponen sistem IDS seperti snort dan fail2ban. Setelah itu, dilakukan instalasi cowrie dan pemasangan bot Telegram. Pada konfigurasi *tools* ini akan dilakukan menggunakan alur sebagai berikut.

- Alur Konfigurasi Snort

Gambar 8 Alur Implementasi Snort

Gambar 8 Alur Implementasi Snort dipasang dan dikonfigurasi pada sistem dengan *rules* yang sesuai, dilanjutkan dengan konfigurasi skema jaringan untuk pengujian sebelum diaktifkan.

- Alur Konfigurasi Fail2Ban



Gambar 9 Alur Implementasi Fail2Ban

Gambar 9 Alur Implementasi Fail2ban akan diinstalasi di sistem Institusi XYZ, diikuti pembuatan dan konfigurasi *file* penyimpanan, serta konfigurasi pada jail.conf untuk memilih servis dan *file* log yang akan dideteksi, sebelum diaktifkan dan siap untuk pengujian.

- Konfigurasi Cowrie

Gambar 10 Alur Implementasi Cowrie

Gambar 10 Alur Implementasi Cowrie diinstal pada private server terpisah dari sistem Institusi XYZ, dikonfigurasi sesuai kebutuhan pengujian, dan siap untuk diuji setelah proses instalasi dan konfigurasi selesai.

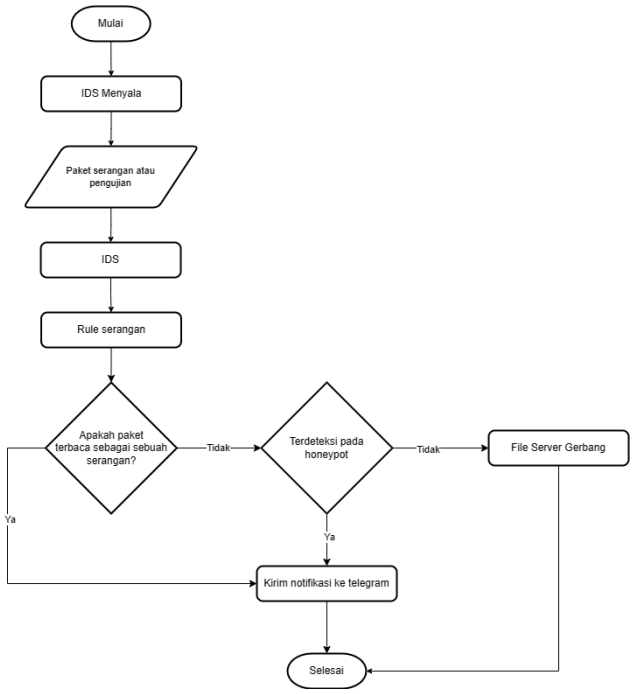
- Konfigurasi Telegram

Gambar 11 Alur Implementasi Bot Telegram

Gambar 11 menunjukkan Instalasi Telegram di *smartphone*, pembuatan dan konfigurasi bot menggunakan BotFather, serta penggunaan bot tersebut untuk menerima log data dan memberikan notifikasi *real-time* dilakukan dengan skrip Python yang disesuaikan.

F. Enforcement

Gambar 12 menunjukkan pengujian IDS dimulai dengan konfigurasi dan verifikasi kinerja. Kemudian, uji coba serangan dilakukan menggunakan *brute force attack* atau percobaan login, dengan notifikasi langsung melalui Telegram jika terdeteksi oleh IDS atau *honeypot*. Proses ini diulang tiga kali dengan skenario topologi yang berbeda.



Gambar 12 Tahap *Enforcement*

G. *Enhancement*

Pada tahap enhancement dalam metode SPDLC, dilakukan analisis sistem yang sudah diuji, memberikan rekomendasi perbaikan, dan membandingkan efektivitas tiga pengujian dengan topologi berbeda.

III. HASIL DAN PEMBAHASAN

Hasil dan pembahasan mencakup hasil dari penerapan tahapan-tahapan yang digunakan dalam pengimplementasian snort dan fail2ban untuk proteksi serangan *brute force* dengan notifikasi telegram sebagai alarm.

A. *Identification*

TABEL 2
PEMETAAN NARASUMBER DENGAN INDEKS KAMI 4.2

Pertanyaan	Jawaban
Apakah *website* Institusi XYZ merupakan aset yang penting bagi berjalannya sistem lain di Institusi XYZ?	Iya sangat penting sekali, karena menyimpan data terkait akademik. Dan karena akademik merupakan inti bisnis di Institusi XYZ.
Apakah *website* Institusi XYZ berisikan informasi yang bersifat sensitif atau tidak ?	Iya, karena apabila informasi tersebar maka akan menimbulkan masalah.
Apakah jika terdapat perubahan atau gangguan yang terjadi pada *website* Institusi XYZ akan mempengaruhi kinerja dari *website* lain yang ada di Institusi XYZ?	Iya, karena berhubungan langsung dengan sistem lain yang ada di Institusi XYZ.
Seberapa sering digunakannya SSH *service* dalam akses sistem server Institusi XYZ?	Jarang, karena biasanya dicek melalui webmin.
Apakah sebelumnya pernah terjadi gangguan keamanan yang berdampak pada data pada server Institusi XYZ ?	Belum ada untuk Institusi XYZ, namun ada kasus kebocoran data karena kesalahan dari *user*.
Berapa lama waktu yang dibutuhkan untuk mengetahui serangan yang terjadi dan durasi yang diperlukan untuk memulihkan sistem server Institusi XYZ?	Untuk mengetahuinya diperlukan 2 hari ketika *user* melaporkan bahwa tidak dapat mengakses. Untuk pemulihannya dibutuhkan waktu kurang dari 1 hari.
Apabila terjadi suatu gangguan atau serangan, apakah terdapat sistem notifikasi yang digunakan untuk mengetahui serangan tersebut ? seperti log dan lainnya ?	Belum ada, masih mengecek lewat log manual saja.

Dari hasil wawancara Tabel 1, dapat disimpulkan bahwa Gerbang Institusi XYZ memiliki nilai CIA tinggi dan karenanya perlu peningkatan keamanan, seperti penerapan sistem deteksi snort dan fail2ban, serta notifikasi alarm melalui aplikasi Telegram untuk mengurangi risiko serangan yang dapat berdampak besar pada sistem informasi Institusi XYZ.

B. *Analysis*

Pada analisis, dilakukan evaluasi spesifikasi sistem yang mencakup perangkat keras dan perangkat lunak, seperti tercantum dalam Tabel 3

.

TABEL 3
SPESIFIKASI SYSTEM

Sistem	Keterangan
Intrusion Detetction System	Bertindak untuk memberikan keterangan apabila telah terdeteksi serangan.
Intrusion Prevention System	Bertindak untuk memblokir IP yang melakukan akses mencurigakan.
Honeypot System	Bertindak untuk menampung serangan *brute force* yang diterima.
Client Server	Bertindak sebagai server utama untuk menampung *website dummy* yang berisikan *login page* dan *home page*.

Client Attacker	Bertindak sebagai penyerang untuk melakukan serangan *brute force* kepada *client server*.
Alarm System	Bertindak sebagai penerima keterangan sistem dan mengirimkan notifikasi *alarm*.

TABEL 4
PERANGKAT LUNAK

Perangkat Lunak	Keterangan
Sistem Operasi	Sistem operasi yang digunakan ada 3 yaitu *Ubuntu, Debian, dan Kali Linux*.
Snort	Sebagai IDS yang akan bekerja pada *layer network*, yang akan mendeteksi serangan.
Fail2ban	Sebagai IPS yang akan bekerja pada *layer application*, yang akan mendeteksi dan melakukan *ban* terhadap penyerang.
Cowrie	Sebagai *Honeypot* yang akan menerima serangan yang dialihkan.
Hydra	Perangkat lunak pada kali linux yang akan digunakan untuk mengirimkan serangan.
Iptables	Perangkat lunak yang akan digunakan untuk melakukan *forwarding* dari *server* ke *honeypot*.
Oracle VM Virtual Box	Perangkat lunak yang akan digunakan untuk menginstalasi sistem operasi secara virtual.
Telegram	Sebagai penerima dan pengirim notifikasi selama dilakukannya pengujian dan sebagai alarm.

TABEL 5
PERANGKAT KERAS

Perangkat Keras	Spesifikasi	Keterangan
Laptop/PC	- Intel® Core™ i5-9300H 4 Cores 8 Threads - 16 GB RAM - Storage 1 TB + 256 GB - NVIDIA GTX 1050 3 GB - Windows OS	Sebagai perangkat keras yang digunakan untuk menjalankan seluruh simulasi sistem yang dibuat.
Smartphone	- Mediatek MT6769H Helio G88 - 6 GB RAM - Storage 128 GB - Android 13	Sebagai perangkat keras yang digunakan untuk menjalankan aplikasi telegram sebagai *alarm system*.

Setelah menganalisis kebutuhan sistem, diperlukan perangkat lunak dan perangkat keras yang tercatat dalam Tabel 4 dan Tabel 5.

C. Implementation 1

Pada tahap implementasi, sistem disusun sesuai desain topologi dengan instalasi dan konfigurasi sistem operasi pada VirtualBox, sehingga sistem siap dijalankan setelah konfigurasi berhasil.

TABEL 6
SPESIFIKASI VIRTUALBOX

Nama Sistem	Sistem Operasi	Spesifikasi
Server IDS	Ubuntu Server 22.04	○ 1 Core ○ 1 GB Ram ○ 16 MB VRAM ○ 10 GB Storage
Server *Honeypot*	Ubuntu Server 22.04	○ 1 Core ○ 1 GB Ram ○ 16 MB VRAM ○ 10 GB Storage
File Server Institusi XYZ	Debian 12.2	○ 2 Cores ○ 2 GB Ram ○ 16 MB VRAM ○ 10 GB Storage
Client Attacker	Kali Linux 2023.3	○ 2 Core ○ 4 GB Ram ○ 128MB VRAM

Nama Sistem	Sistem Operasi	Spesifikasi
		○ 50 GB Storage

Setelah instalasi sistem operasi berhasil, konfigurasi IP dilakukan untuk menghubungkan dan menjalankan semua sistem operasi sesuai dengan topologi 1 yang telah dirancang.

- Konfigurasi IP *Address* dan *routing* pada topologi dimulai:
  - Konfigurasi IP
  - Atur IP *Address* di Server IDS
  - Atur **iptables net.ipv4.port.forward = yes**
  - Cek status dengan *command* "**sysctl-p**"
  - Tambahkan rule nat pada iptables
  - Atur sistem lain yang terhubung dengan server IDS dengan *interface internal network*
  - Atur IP *Address* dan Gateway yang menggunakan IP dari server IDS.

- Instalasi dan Konfigurasi Snort
  - Install snort dengan "**apt-get install snort**"
  - Buka *file* konfigurasi snort.conf atau *copy* dan ubah nama *file* terlebih dahulu jika ingin mengatur snort khusus untuk memantau *interface* tertentu seperti pada topologi ini menjadi ***snort.enp0s8.conf*** karena hanya digunakan untuk memantau *request* dari *interface* tersebut (*client attacker)*
  - Didalam *file* konfigurasi yang telah dibuat atur *ipvar* HOME_NET dengan *any*
  - Atur *rules* di ***snort/rules/local/rules***
  - *Restart* snort dengan *command* "***systemctl restart snort***"
  - Lakukan pengujian pada *rules*
  - *Running* snort
  - Buat *file bash* dengan nama ***bot-tele.sh*** untuk memantau log snort dan mengirim pesan ke telegram
  - Running snort dan *file* bash bersamaan dengan menggunakan package screen.
- Instalasi dan Konfigurasi NFS
  - Install NFS
  - Konfigurasi folder yang ingin di-*share* agar *nfs client* atau pada topologi ini yaitu server IDS dapet mengakses folder yang ada pada *file server* Institusi XYZ
  - Restart NFS
  - Berikan akses pada folder agar folder dapat diakses *client*
  - *Install* NFS
  - Cek koneksi dari server IDs ke *file* server Institusi XYZ
  - Buat folder kemudian *mount folder* dengan *file* pada folder yang ada di NFS *file* server Institusi XYZ

- Instalasi dan Konfigurasi Fail2Ban
  - Install fail2ban dengan *command* "***apt-get install fail2ban"***
  - Buka dan ubah *file* jail.local dengan *command* "***nano /etc/fail2ban/jail.local***" dan tambahkan *rules*.
  - Buat dan tambahkan *file filter web-login* pada folder filter.d dengan *command* "***nano /etc/fail2ban/filter.d/web-login.conf***"
  - Tambahkan *file action* telegram untuk mengirim notifikasi telegram ketika ada serangan pada folder action.d dengan *command* "***nano /etc/fail2ban/action.d/telegram.conf***"
  - Tambahkan *file bash* yang nantinya akan dijalankan ketika ada serangan pada folder *scripts* dengan *command* "***nano /etc/fail2ban/scripts/telegram_notif.sh***)"
  - *Restart* dan aktifkan fail2ban dengan *command* **"*systemctl restart fail2ban*", "*systemctl enable fail2ban***"
  - Cek status fail2ban dengan *command* berikut, **"*fail2ban-client status [nama_rule = optional]*"**
  - Untuk melakukan *Unbanned* ip yang sudah di-*banned* dapat menggunakan *command* berikut, **"*file2ban-client set [nama_rule] unbanip [ip_add]*"**

- Instalasi dan Konfigurasi Cowrie
  - *Update* dan *upgrade* ubuntu dengan *command*, "***apt-get update***" & "***apt-get upgrade***"
  - Tambahkan user cowrie
  - Berikan akses sudo ke *username* cowrie dengan menambahkan kode pada *file* sudoers.
  - *Install package*
  - Masuk sebagai *user* cowrie dan unduh *repository* cowrie dari *github*
  - Membuat virtualenv untuk *honeypot* karena cowrie berjalan pada virtual *environment* python
  - Memasang *package* cowrie pada *honeypot* yang *package* tersebut dapat dilihat pada *file* requirement.txt
  - Keluar dari mode cowrie-env
  - Menduplikat *file* cowrie.cfg.dist menjadi cowrie.cfg
  - Menambahkan iptables untuk *redirect port* 2222 ke *port* 22
  - Jalankan Cowrie

- Instalasi dan Konfigurasi SSH *service*
  - *Install* SSH dengan "***apt-get install openssh-server***"
  - Nyalakan *password authentication* agar server bisa di-*remote* dengan SSH
  - Aktifkan *password* untuk *login root* agar bisa *remote* dengan *root*
  - *Restart* ssh *server.*

D. Enforcement 1

Setelah implementasi, tahap *enforcement* melibatkan pengujian dengan serangan *brute force attack* dari Kali

Linux dan percobaan sederhana untuk mengidentifikasi *false alarm* serta *no alarm*, dengan detail pengujian tertera pada Tabel 7.

TABEL 7
JENIS SERANGAN

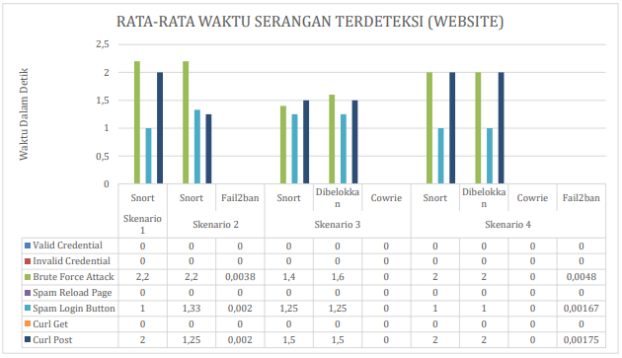
Target Serangan	Bentuk Serangan	Keterangan
Website	*Valid Credential*	Melakukan login dengan username dan password yang benar pada *website dummy* secara cepat.
	Invalid Credential	Melakukan login dengan username dan password yang salah pada *website dummy* secara cepat.
	Brute Force Attack	Mengirimkan *packet* serangan menggunakan hydra melalui kali linux.
	Spam Reload Page	Melakukan reload page pada login page secara cepat.
	Spam Login Page	Melakukan spam pada tombol login pada login page secara cepat.
	Curl Get	Mengirimkan perintah get melalui kali linux ke *website*
	Curl Post	Mengirimkan perintah post melalui kali linux ke *website*
SSH	*Valid Credential*	Melakukan login dengan username dan password yang benar pada SSH secara cepat.
	Invalid Credential	Melakukan login dengan username dan password yang salah pada SSH secara cepat.
	Brute Force Attack	Mengirimkan *packet* serangan menggunakan hydra melalui kali linux.
	Manual SSH Access Spam	Melakukan akses ssh secara cepat.

TABEL 8
SKENARIO TESTING

Skenario *Tools*	*Tools* yang Digunakan
Skenario 1	Snort
Skenario 2	Snort dan Fail2ban
Skenario 3	Snort dan Cowrie
Skenario 4	Snort, Fail2ban, dan Cowrie

E. Enhancement 1

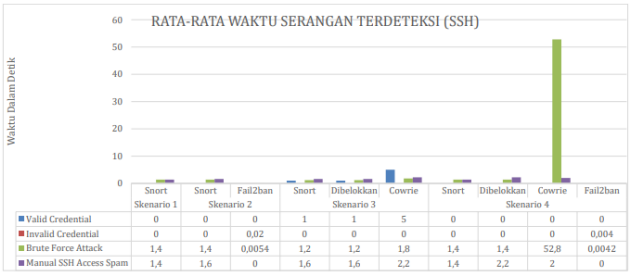
- *Website*



Gambar 13 Rata-rata waktu serangan terdeteksi (*WEBSITE*)

Dari Gambar 13, snort memiliki performa stabil dalam mendeteksi serangan dengan rata-rata waktu 1-3 detik, walaupun terjadi kasus *no alarm* pada serangan tertentu. Fail2ban efektif menutupi kelemahan snort dengan mendeteksi dan memblokir serangan dalam waktu kurang dari 1 detik. Meskipun demikian, terdapat dua kasus di mana kombinasi snort dan fail2ban tidak dapat mendeteksi serangan. Performa cowrie sebagai *honeypot* tidak dapat dinilai dalam pengujian ini, tetapi cowrie dapat menerima serangan yang dibelokkan oleh snort dalam waktu 1-2 detik tanpa adanya *false alarm*.

- SSH



Gambar 14 Rata-rata waktu serangan terdeteksi (SSH)

Gambar 14 pengujian menunjukkan bahwa snort dalam topologi 1 memiliki performa baik dengan rata-rata waktu deteksi serangan antara 1,2 sampai 2,2 detik, meskipun terjadi satu *false alarm* pada skenario 3. Cowrie sebagai *honeypot* juga efektif mendeteksi serangan yang dibelokkan oleh snort. Namun, kelemahan terlihat pada fail2ban yang memerlukan

waktu lebih lama karena ketergantungan pada log nfs, menyebabkan lebih banyak *no alarm* pada pengujian *website*. Meskipun begitu, fail2ban mampu menutupi kekurangan snort dalam beberapa situasi.

F. Implementation 2

Pada tahap implementasi ini, sistem akan disusun sesuai desain topologi 2 dengan langkah awal instalasi sistem operasi pada virtualbox, yang melibatkan proses instalasi dan konfigurasi serupa dengan topologi 1, namun dengan perbedaan penempatan fail2ban pada *file* server Institusi XYZ bersamaan dengan *website*.

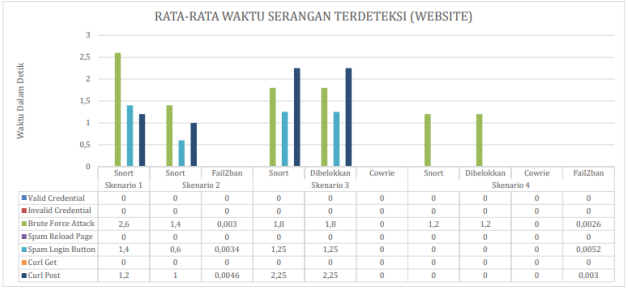
- Instalasi dan Konfigurasi Fail2Ban
  - Install fail2ban dengan *command* berikut. "***apt-get install fail2ban***"
  - Ubah jail.local (/etc/fail2ban/jail.local)
  - Tambahkan filter web-login pada folder filter.d yanmg diakses menggunakan *command*"***/etc/fail2ban/filter.d/web-login.conf***"
  - Tambahkan *file action* telegram untuk mengirim notif telegram ketika ada serangan pada folder action.d (/etc/fail2ban/action.d/telegram.conf)
  - Tambahkan *file* bash yang akan dijalankan ketika ada serangan pada folder action.d (/etc/fail2ban/scripts/telegram_notif.sh)
  - Restart dan aktifkan fail2ban dengan *command* "***systemctl restart fail2ban***", "***systemctl enable fail2ban***"
  - Cek status fail2ban dengan *command* "***fail2ban-client status [nama_rule = optional]***"
  - *Ubanned* ip yang sudah dibanned dengan *command* dibawah. "***file2ban-client set [nama_rule] unbanip [ip_add]***"

G. Enforcement 2

Pada tahapan *Enforcement* 2 ini dimana tahap ini akan dilakukan pengujian yang sama dengan yang dilakukan pada tahapan *Enforcement* 1. Dengan target, bentuk serangan serta scenario yang sama.

H. Enhancement 2

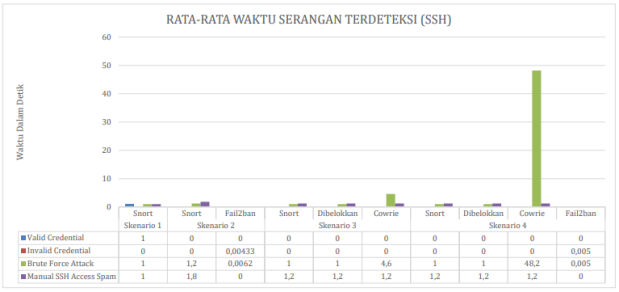
- *Website*



Gambar 15 Rata-rata waktu serangan terdeteksi (*WEBSITE*)

Pada Gambar 15 Snort, fail2ban, dan *honeypot* bekerja efisien tanpa tugas yang tumpang tindih, dengan performa stabil. Snort memiliki waktu deteksi serangan 1-3 detik, fail2ban mendeteksi dan memblokir dalam waktu kurang dari 1 detik, serta *honeypot* menerima serangan dibelokkan dengan waktu pembelokan 1-2 detik. Pada pengujian kombinasi ketiga *tools*, hasil maksimal tercapai, namun snort sering no alarm ketika dikombinasikan dengan fail2ban karena IP penyerang sudah di-banned oleh fail2ban sebelum snort mendeteksi serangan.

- SSH



Gambar 16 Rata-rata waktu serangan terdeteksi (SSH)

Pada pengujian skenario 1 Gambar 16, snort menunjukkan performa yang buruk dengan 1 *false alarm*, sedangkan fail2ban pada topologi 2 sering tidak memberikan *alarm* pada uji *invalid credential* karena pengaturan *default* ssh *service*. Namun, pada skenario 4, setelah semua *tools* diaktifkan, performa secara keseluruhan sangat baik, khususnya pada uji *brute force attack* dan manual SSH *access spam*, tanpa adanya *false alarm*. Topologi 2 memiliki kelebihan dalam performa yang stabil dan efisien, namun demikian, penempatan *tools* yang berbeda mempengaruhi kinerja masing-masing.

I. Implementation 3

Pada tahap *Implementation* 3, sistem akan disusun sesuai desain topologi yang membutuhkan instalasi sistem operasi pada virtualbox dengan spesifikasi yang tertera pada Tabel 9. Tahap ini berbeda karena hanya melibatkan 3 sistem, berbeda dengan dua topologi sebelumnya.

TABEL 9
SPESIFIKASI VIRTUALBOX TOPOLOGI 3

Nama Sistem	Sistem Operasi	Spesifikasi
Server Honeypot	Ubuntu Server 22.04	1 Core 1 GB Ram 16 MB VRAM 10 GB Storage
File Server Institusi XYZ	Debian 12.2	2 Cores 2 GB Ram 16 MB VRAM 10 GB Storage
Client Attacker	Kali Linux 2023.3	2 Core 4 GB Ram 128 MB VRAM 50 GB Storage

- Konfigurasi IP *Address* dan routing pada topologi dimulai:
  - Atur interface *file* server Institusi XYZ sebagai *router*
  - Atur IP *Address* di *file* server Institusi XYZ

- IP *Address client attacker* dan *server honeypot* sama seperti topologi sebelumnya

- Instalasi dan Konfigurasi Snort:
  - Konfigurasi snort pada topologi 3 hampir sama dengan topologi 1 & 2 namun ada sedikit perubahan pada *rules* untuk mendeteksi *brute force website*.

```
# ALERT ON ANY TCP CONNECTION ATTEMPT
alert tcp any any -> 192.168.10.5 80 (MSG: "[WEB] Serangan Brute Force"; content:"POST
alert tcp any any -> 192.168.10.5 22 (MSG: "[SSH] Serangan Brute Force"; flags: S+; de
```

Gambar 17 Snort *Rules* Topologi

```
# ALERT ON ANY TCP CONNECTION ATTEMPT
:"POST /login"; detection_filter:track by_src, count 30, seconds 60; sid:1000001; rev
alert tcp any any -> 192.168.10.5 22 (MSG: "[SSH] Serangan Brute Force"; flags: S+; de
```

Gambar 18 Snort *Rules* Topologi

Gambar 17 dan Gambar 18 menunjukkan *rules* yang ditanamkan pada snort yang berisi *port* yang dipantau dan juga keterangan serangan dari masing-masing *port*, kemudian ditanamkan pula perintah untuk menyimpan keterangan pada log.

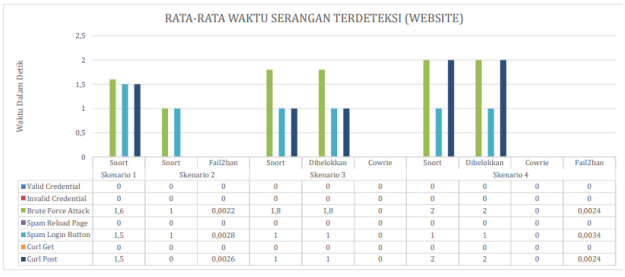
- Running snort

J. *Enforcement 3*

Pada tahapan *Enforcement* 3 ini dimana tahap ini akan dilakukan pengujian yang sama dengan yang dilakukan pada tahapan *Enforcement* 1 dan *Enforcement* 2. Dengan target, bentuk serangan serta skenario yang sama.

K. *Enhancement 3*

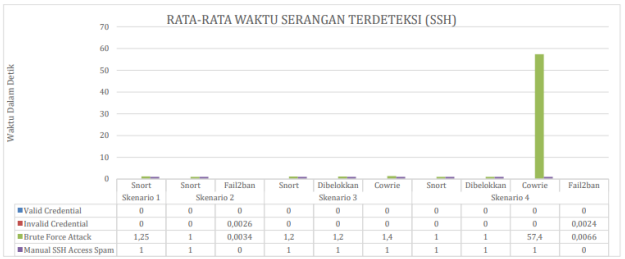
- *Website*



Gambar 19 Rata-rata waktu serangan terdeteksi (*WEBSITE*)

Pada Gambar 19 penempatan snort pada topologi ini memiliki performa yang buruk dalam mendeteksi sebagian besar serangan, terutama pada uji coba *spam login button* dan *curl post*. Meskipun snort dapat mendeteksi *brute force attack*, performanya tetap rendah dan seringkali gagal mendeteksi sebagian besar serangan, bahkan ketika diaktifkan bersamaan dengan fail2ban. Di sisi lain, fail2ban menunjukkan performa yang baik dengan mendeteksi, memblokir serangan, dan memberikan *alert* tanpa kasus *false alarm* atau *no alarm*, dengan rata-rata waktu *respons* kurang dari 1 detik. Meskipun *honeypot* pada topologi ini tidak dievaluasi secara khusus, ia masih mampu menerima serangan yang dibelokkan oleh snort dengan rata-rata waktu pembelokan serangan 1-2 detik.

- SSH



Gambar 20 Rata-rata waktu serangan terdeteksi (SSH)

Dari hasil pengujian Gambar 20, Snort menunjukkan performa buruk terutama dalam mendeteksi serangan SSH *Access Spam*, namun memiliki hasil yang kurang lebih sama dengan topologi lain pada pengujian *brute force attack*. Topologi 3 menunjukkan kelebihan pada performa fail2ban dalam mendeteksi dan memblokir serangan *website*, meskipun terdapat kekurangan terkait penempatan *tools* Snort yang mempengaruhi performa secara keseluruhan. Setelah evaluasi, topologi 2 dipilih sebagai yang terbaik karena memiliki akurasi 100% pada skenario 4 dan rata-rata waktu deteksi lebih cepat dibanding topologi lain, meskipun konfigurasinya belum tepat.

Mendukung atau tidaknya terkait hasil dengan tujuan pada penelitian ini, maka berikut disajikan korelasi antara hasil dan tujuan penelitian.

Tujuan 1: Mengetahui Kinerja Snort dan Fail2Ban sebagai IDS dalam Mengamankan Sebuah Sistem Informasi Hasil penelitian menunjukkan bahwa kombinasi *Snort* dan Fail2Ban efektif dalam mendeteksi dan merespon serangan *brute force*. *Snort*, sebagai IDS, berhasil mendeteksi 100% serangan *brute force* yang dilakukan selama pengujian. Fail2Ban, yang berfungsi sebagai IPS, mampu memblokir IP yang mencurigakan berdasarkan log yang dihasilkan oleh *Snort*. Semua serangan *brute force* yang dilakukan pada jaringan berhasil dideteksi oleh *Snort*. Hal ini menunjukkan bahwa *Snort* memiliki aturan (*rules*) yang efisien dalam mengenali pola serangan brute force. Sementara itu Fail2Ban mampu memblokir IP penyerang setelah beberapa percobaan *login* yang gagal terdeteksi oleh *Snort*. Ini menunjukkan bahwa integrasi antara *Snort* dan *Fail2Ban* berjalan dengan baik. Kinerja *Snort* dan *Fail2Ban* dalam mendeteksi dan memblokir serangan *brute force* sangat baik, mendukung tujuan penelitian untuk mengetahui efektivitas tools tersebut sebagai IDS untuk implementasi pengamanan sistem informasi. Hasil ini mengindikasikan bahwa kombinasi keduanya dapat digunakan sebagai alternatif solusi keamanan jaringan, terutama untuk mendeteksi dan mencegah serangan *brute force*.

Tujuan 2: Mengetahui Apakah Cowrie sebagai Honeypot Dapat Membantu IDS dalam Meminimalisir Terjadinya False Alarm dan No Alarm. Cowrie, yang berfungsi sebagai *honeypot*, memainkan peran penting dalam mengurangi *false alarm* dan *no alarm* pada sistem IDS. Dengan mengarahkan serangan yang mencurigakan

ke *honeypot*, *Cowrie* mampu mencatat aktivitas penyerang tanpa mengganggu sistem utama. *Cowrie* berhasil mengalihkan serangan *brute force* yang dilakukan oleh bot, dan memungkinkan pencatatan aktivitas serangan secara detail tanpa menimbulkan *false alarm* pada sistem utama.

Pencatatan Aktivitas: Semua aktivitas yang dilakukan oleh penyerang di *honeypot* tercatat dengan baik, memberikan data berharga untuk analisis lebih lanjut. Dengan berfungsinya *Cowrie* sebagai *honeypot*, sistem IDS dapat fokus pada deteksi serangan tanpa terbebani oleh *false alarm*. Ini menunjukkan bahwa *Cowrie* efektif dalam membantu IDS meminimalisir *false alarm* dan *no alarm*, mendukung tujuan penelitian untuk mengevaluasi peran *honeypot* dalam sistem keamanan.

Tujuan 3: Mengetahui Apakah IDS Dapat Menjadi Solusi Alternatif Apabila Belum Memiliki Firewall. Hasil penelitian menunjukkan bahwa IDS dapat menjadi solusi alternatif yang efektif apabila *firewall* belum tersedia atau memiliki keterbatasan. Sistem yang dikembangkan, terdiri dari *Snort*, *Fail2Ban*, dan *Cowrie*, mampu mendeteksi dan merespon serangan dengan baik tanpa adanya *firewall*. Sistem IDS yang dikembangkan mampu mendeteksi dan memblokir serangan *brute force* secara efektif meskipun tanpa firewall. Respons Cepat: Notifikasi melalui Telegram memungkinkan respons cepat dari administrator jaringan ketika terjadi serangan. Hasil ini menunjukkan bahwa IDS dapat berfungsi sebagai alternatif solusi keamanan jaringan yang efektif, terutama dalam kondisi di mana *firewall* tidak tersedia. Implementasi *Snort*, *Fail2Ban*, dan *Cowrie* dapat memberikan lapisan keamanan tambahan yang signifikan.

Tujuan 4: Mengetahui Susunan Topologi yang Paling Efektif dalam Pengimplementasian *Tools* untuk Menangani Serangan *Brute Force*. Pengujian dilakukan dengan beberapa topologi yang berbeda untuk menilai efektivitas penempatan *Snort*, *Fail2Ban*, dan *Cowrie* dalam menangani serangan *brute force*. Topologi yang paling efektif ditemukan dengan menempatkan *Snort* di depan server utama, Fail2Ban di layer aplikasi, dan *Cowrie* sebagai *honeypot* di belakang *Snort*. Topologi Optimal yang didapat yaitu Topologi yang menempatkan *Snort* di depan *server*, *Fail2Ban* di layer aplikasi, dan *Cowrie* sebagai *honeypot* di belakang *Snort* menunjukkan performa terbaik dalam mendeteksi dan mengalihkan serangan. Sementara itu untuk respons dan pencatatan, sistem mampu merespon serangan secara *real-time* dan mencatat aktivitas serangan dengan baik, memberikan data berharga untuk analisis lebih lanjut. Hasil ini menunjukkan bahwa penempatan *Snort*, *Fail2Ba*n, dan *Cowrie* dalam topologi yang tepat sangat penting untuk efektivitas sistem keamanan. Topologi yang optimal memberikan perlindungan yang lebih baik dan memungkinkan respons cepat terhadap serangan.

IV. KESIMPULAN

Penelitian ini bertujuan untuk mengevaluasi efektivitas kombinasi *Snort* dan *Fail2Ba*n sebagai *Intrusion Detection System* (IDS) dan *Cowrie* sebagai *honeypot* dalam menghadapi serangan *brute force*. Temuan utama dari penelitian ini adalah sebagai berikut:

1. Efektivitas *Snort* dan *Fail2Ban*: Kombinasi *Snort* dan *Fail2Ban* berhasil mendeteksi dan memblokir serangan brute force dengan tingkat deteksi dan respon yang tinggi. *Snort* efektif dalam mendeteksi serangan pada level jaringan, sementara Fail2Ban efektif dalam memblokir IP yang mencurigakan berdasarkan log yang dihasilkan oleh *Snort*.
2. Peran *Cowrie* sebagai *Honeypot*: *Cowrie* terbukti efektif dalam mengalihkan serangan *brute force* dan mencatat aktivitas penyerang tanpa menimbulkan *false alarm* pada sistem utama. Hal ini membantu dalam meminimalisir *false alarm* dan *no alarm* yang sering terjadi pada IDS.
3. IDS sebagai Solusi Alternatif: IDS yang terdiri dari *Snort*, *Fail2Ban*, dan *Cowrie* dapat berfungsi sebagai solusi alternatif yang efektif ketika *firewall* tidak tersedia atau memiliki keterbatasan. Sistem ini mampu mendeteksi dan merespon serangan dengan baik tanpa memerlukan *firewall*.
4. Topologi Optimal: Penempatan *Snort* di depan server utama, *Fail2Ban* di layer aplikasi, dan *Cowrie* sebagai *honeypot* di belakang *Snort* merupakan susunan topologi yang paling efektif. Topologi ini memberikan perlindungan yang optimal dan memungkinkan respons cepat terhadap serangan.

Penelitian ini menunjukkan bahwa kombinasi *Snort*, *Fail2Ban*, dan *Cowrie* dapat digunakan secara efektif untuk meningkatkan keamanan jaringan terhadap serangan *brute force*. Implementasi sistem ini dapat diterapkan pada institusi atau perusahaan yang membutuhkan solusi keamanan jaringan yang andal dan efisien, terutama ketika *firewall* yang berkualitas tidak tersedia.

Penelitian ini memberikan panduan praktis untuk konfigurasi dan implementasi IDS yang efektif, serta menekankan pentingnya penempatan topologi yang tepat untuk memaksimalkan performa sistem keamanan jaringan.

DAFTAR PUSTAKA

[1] A. D. Andayani dan O. C. Briliyant, "Penilaian Kapabilitas Tata Kelola Keamanan Teknologi Informasi dan Rekomendasi Perbaikan Menggunakan COBIT 5," *Info Kripto*, vol. 15, no. 1, hal. 1–10, 2021, doi: 10.56706/ik.v15i1.17.

[2] J. Shahid, M. K. Hameed, I. T. Javed, K. N. Qureshi, M. Ali, dan N. Crespi, "A Comparative Study of Web Application Security Parameters: Current Trends and Future Directions," *Appl. Sci.*, vol. 12, no. 8, 2022, doi: 10.3390/app12084077.

[3] A. R. Riswaya, A. Sasongko, A. Maulana, S. M. Indonesia, dan U. L. Bandung, "Evaluasi Tata Kelola Keamanan Teknologi Informasi Menggunakan Indeks KAMI Untuk Persiapan Standar SNI ISO/IEC 27001 (Studi Kasus: STMIK Mardira Indonesia)," *J. Comput. Bisnis*, vol. 14, no. 1, hal. 10–18, 2020.

[4] L. Reznik, "Firewall Design and Implementation," in *Intelligent Security Systems*, Wiley, 2021, hal. 57–108. doi: 10.1002/9781119771579.ch2.

[5] Z. Ahmad, A. S. Khan, C. W. Shiang, J. Abdullah, dan F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 1, 2021, doi: 10.1002/ett.4150.

[6] A. Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman, dan A. Alazab, "Hybrid Intrusion Detection System Based on the Stacking Ensemble of C5 Decision Tree Classifier and One Class Support Vector Machine," *Electronics*, vol. 9, no. 1, hal.

173, 2020, doi: 10.3390/electronics9010173.
[7] T. Ernawati dan F. F. F. Rachmat, "Keamanan Jaringan dengan Cowrie Honeypot dan Snort Inline-Mode sebagai Intrusion Prevention System," *J. RESTI (Rekayasa Sist. dan Teknol. Informasi)*, vol. 5, no. 1, hal. 180–186, 2021, doi: 10.29207/resti.v5i1.2825.
[8] A. Mutaqin, "Rancang Bangun Sistem Monitoring Keamanan Jaringan Prodi Teknik Informatika Melalui SMS Alert dengan Snort," *J. Sist. dan Teknol. Inf.*, vol. 1, 2016.
[9] R. Vansuri *et al.*, "Universitas Bhayangkara Jakarta Raya, Indonesia, achmad.fauzi@dsn.ubharajaya.ac.id 3. Universitas Bhayangkara Jakarta Raya, Indonesia, ery.teguh@ubharajaya.ac.id 4," vol. 2, no. 1, doi: 10.38035/jim.v2i1.
[10] T. Tuyikeze dan D. Pottas, "An Information Security Policy Development Life Cycle." 2010.
[11] L. A. Wahsheh dan J. Alves-Foss, "Security Policy Development: Towards a Life-Cycle and Logic-Based Verification Model," *Am. J. Appl. Sci.*, vol. 5, no. 9, hal. 1117–1126, 2008, doi: 10.3844/ajassp.2008.1117.1126.
[12] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, dan K.-Y. Tung, "Intrusion detection system: A comprehensive review," *J. Netw. Comput. Appl.*, vol. 36, no. 1, hal. 16–24, 2013, doi: 10.1016/j.jnca.2012.09.004.
[13] B. Caswell, J. Beale, dan A. Baker, *Snort Intrusion Detection and Prevention Toolkit*. Elsevier, 2007. doi: 10.1016/B978-1-59749-099-3.X5000-9.
[14] R. Ramadhan, J. Latuny, dan S. J. Litiloly, "Perancangan Pengamanan Server Apache Menggunakan Firewall Iptables Dan Fail2ban," *J. ISOMETRI*, vol. 1, no. 1, hal. 9–15, 2022, doi: 10.30598/isometri.2022.1.1.9-15.
[15] S. H. J. Marzuki dan M. Azmi, "Layanan Informasi Pembayaran Biaya Kuliah Berbasis SMS Gateway," *Tek. Teknol. Inf. dan Multimed.*, vol. 2, no. 1, hal. 24–28, 2021, doi: 10.46764/teknimedia.v2i1.32.
[16] U. Ubaidillah, T. Taryo, dan A. Hindasyah, "Analisis dan Implementasi Honeypot Honeyd Sebagai Low Interaction Terhadap Serangan Distributed Denial Of Service (DDOS) dan Malware," *JTIM J. Teknol. Inf. dan Multimed.*, vol. 5, no. 3, hal. 208–217, 2023, doi: 10.35746/jtim.v5i3.405.