Network Security Monitoring System via Whatsapp using Snort on Ubuntu

Jefri Simanjuntak[1*], Syahril Rizal[2]

Abstract

This research addresses critical network security vulnerabilities identified at the Palembang City Office of Culture and Tourism (Dinas Kebudayaan Dan Pariwisata Kota Palembang), an agency responsible for managing regional tourism and cultural assets. The necessity for heightened security became apparent following a successful Distributed Denial of Service (DDOS) attack against the agency's operational website. The investigation diagnosed the incident's root causes as high network vulnerability stemming from reliance on standard, inadequately secured infrastructure, coupled with insufficient human resource (HR) capacity and the complete absence of effective legacy prevention systems [1]. The solution implemented involves the construction of a Network Security Monitoring System utilizing the open-source Snort application, configured as an Intrusion Detection System (IDS) deployed on an Ubuntu platform. The core technical innovation of this system lies in its capacity for real-time alert delivery, leveraging the ubiquitous Whatsapp instant messaging application to notify the network administrator immediately upon the detection of an intrusion [1]. Validation testing confirms the system's efficacy in detecting specific, high-risk attack vectors, including SSH brute force login attempts, Telnet access probes, File Transfer Protocol (FTP) access attempts, and high-volume DDOS flooding [1]. The successful deployment demonstrates a viable, low-cost, and responsive strategy for public sector entities to transition from a passive security approach to a proactive, real-time security monitoring posture.

Keywords

Network Security,
Vulnerability Assessment
Distributed Denial of Service (DDoS),Intrusion Detection System (IDS)
Snort (Open-Source IDS),Ubuntu Platform

Article History
Received 11 Descember 2024
Accepted 25 February 2025

How to Cite
Simanjuntak, J., & Rizal, S. (2025). Network Security Monitoring System via Whatsapp using Snort on Ubuntu. ***Jurnal Jaringan Komputer dan Keamanan,*** *6*(1), 9-21.

[1*] Universitas Bina Darma, Indonesia, Corresponding email: j.simanjuntak241098@gmail.com
[1*] Universitas Bina Darma, Indonesia, email: syahril.rizal@binadarma.ac.id

Introduction

A computer network is formally defined as an integrated system comprised of multiple computing devices designed to enable resource sharing, facilitate communication (such as email and instant messaging), and provide access to information resources (such as web browsers). For a computer network to achieve its operational objectives, a service-oriented architecture is typically employed, wherein various components are designed to request and provide specific services. The entity requesting or receiving a service is traditionally termed the client, while the entity providing or transmitting the service is termed the server. This design paradigm, universally recognized as the client-server system, underpins virtually all contemporary computer network applications.

Network security represents a critical and continuous process focused on preventing, detecting, and monitoring unauthorized usage within a computer network environment. The fundamental purpose of implementing network security is to proactively anticipate and mitigate risks faced by the infrastructure. These risks encompass both physical and logical threats that, whether directly or indirectly, possess the potential to interfere with the ongoing, essential activities conducted within the network.

The Palembang City Tourism Office network exhibited significant security deficits prior to this research intervention. The infrastructure relied upon a rather standard internet network configuration, where connectivity was established directly using a default, standard modem router provided by the telecommunications operator (Telkom). This prevailing condition a standard network coupled with a complete lack of dedicated network security measures resulted in a high-vulnerability environment. This vulnerability was acutely demonstrated when the office experienced a successful DDOS attack targeting its official website.

A subsequent analysis identified key systemic deficiencies contributing to this security failure. These included inadequate human resource capacity and the absence of preventative systems capable of effectively detecting or mitigating sophisticated hacking attacks. The combination of vulnerable network architecture and insufficient administrative oversight necessitated the rapid deployment of a specialized security solution. The agency explicitly required a system capable of detecting, identifying, and instantaneously communicating any attack attempts directed against its public website and internal network infrastructure.

The solution proposed and implemented to address the severe vulnerabilities at the Palembang City Tourism Office involved the construction of a comprehensive network security monitoring system built around Snort. Snort is an industry-standard, lightweight Network Intrusion Detection System (NIDS) utilized for passive network monitoring. Its functionality is predicated upon a sophisticated rules system, which can be custom-configured to meet specific organizational needs, enabling the software to detect and log a wide variety of malicious activities or unusual traffic patterns occurring across the network.

The motivation for integrating an IDS stems from the observation that many network attacks are only discovered post-event, often after strange or disruptive occurrences have already impacted network stability or data integrity. To resolve this delay, the system was designed to monitor server activity in real-time, utilizing Snort to record attacks and subsequently employing a custom mechanism to transmit attack notifications directly via Whatsapp to the network administrator's mobile device.

The choice to integrate Snort alerts with Whatsapp provides a crucial operational advantage over conventional security monitoring systems. The initial diagnosis identified deficient human resource capacity as a contributing factor to the security lapses. In environments where administrators may not be physically present or actively monitoring server logs at all times, relying on passive log storage is inherently risky. The implementation of real-time mobile notification effectively bridges this operational gap. By ensuring that alerts are delivered instantaneously to the administrator's personal communication device, the system transforms detection into an immediate, actionable security event, thereby mitigating the severe risk associated with delayed human intervention and improving the overall administrative response time.

Methodology

Diagnosis

The research process commenced with an initial diagnosis phase conducted directly at the Palembang City Tourism Office. This diagnostic assessment confirmed the lack of any dedicated network security measures. The existing network architecture utilized a simple setup where the internet connection provided by Telkom was directly routed via a switch to the individual computing devices, confirming a minimal and unsecured configuration. The diagnosis included a physical inventory of all active network devices and the execution of a preparatory network scanning protocol utilizing the NMAP application to map the current vulnerability landscape.

Topology

The network arrangement deployed at the Palembang City Tourism and Culture Office is visually represented in Figure 1 of the original document [1].

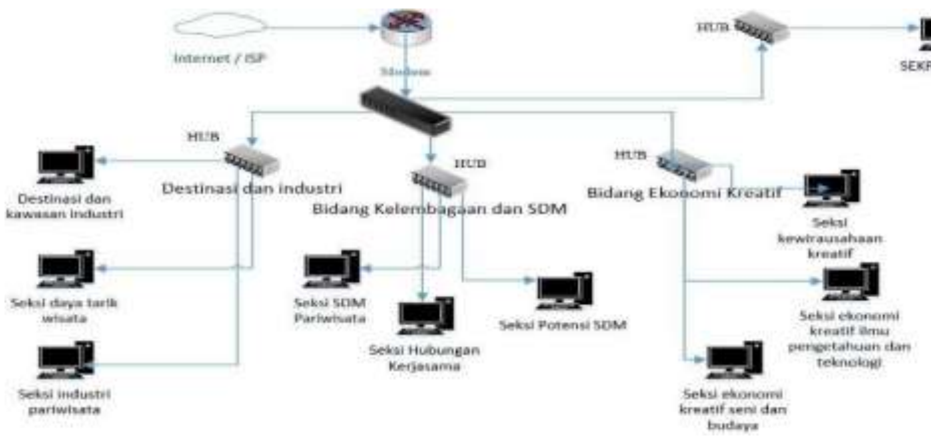

Internet / ISP
Modem
HUB
SEKRETARIAT
HUB
Destinasi dan industri
HUB
Bidang Kelembagaan dan SDM
HUB
Bidang Ekonomi Kreatif
Destinasi dan kawasan industri
Seksi daya tarik wisata
Seksi industri pariwisata
Seksi SDM Pariwisata
Seksi Hubungan Kerjasama
Seksi Potensi SDM
Seksi kewirausahaan kreatif
Seksi ekonomi kreatif ilmu pengetahuan dan teknologi
Seksi ekonomi kreatif seni dan budaya

Figure 1. Network Topology

This topology confirms a centralized structure connecting various functional divisions, including the Secretariat, Destination and Industrial Management, Creative Economy, and

Institutional and HR divisions. The key hardware components and their functions within the infrastructure are critical to understanding the environment requiring protection:

Device	**Function in the Infrastructure**
Modem or Router	Functions as the primary gateway responsible for managing the internet connection [1]
Switch	Facilitates Layer 2 connectivity, helping to connect network segments and distribute internet access to each connected computer [1]
HUB	Used for distributing network connectivity originating from the switch to simplify Local Area Network (LAN) installation within various office rooms [1]
Access Point	Provides wireless connectivity services, acting as a Wi-Fi or Hotspot [1]
PC Computers	The primary endpoint devices where ongoing work processes are executed [1]

The operational criticality of this network is underscored by the high-value applications it supports. These include the financial reporting applications e-monev.bappenas.go.id (used for APBN fund reporting) and skid.djpk.kemenkeu.go.id (used for DAK non-physical reporting). Furthermore, essential editing applications such as Canva, Photoshop, and CorelDraw are utilized by the creative economy division, emphasizing the necessity of maintaining high network stability and integrity.

Problem Identification

Based on the diagnostic activities, the computer network at the Palembang City Tourism and Culture Office was determined to possess several significant security weaknesses. A structured analysis linked these weaknesses to their underlying causes and projected operational consequences.

Table 1: Identified Network Weaknesses, Causes, and Consequences

Weakness	**Cause**	**Consequence**
Absence of network security	Failure to implement a firewall on the network [1]	The network is highly susceptible to compromise and remote takeover by unauthorized attackers [1]
Existence of open ports	Failure to actively close unused or vulnerable access ports [1]	Open ports enable attackers to launch various targeted assaults against network-connected hosts [1]
Lack of bandwidth management	Failure to allocate or segment bandwidth usage across different	Results in network instability and performance degradation across the

	organizational divisions [1]	entire infrastructure [1]
Absence of early detection mechanisms	Lack of a dedicated supporting application for network security monitoring [1]	Prevents the administrator from receiving timely notification when the network is actively under attack [1]

The systematic process of problem identification included formal network scanning (Figure 2, Scanning Jaringan), utilizing the NMAP application to empirically confirm vulnerabilities [1]. The specific command executed was a comprehensive scan: nmap -p 1-65535 -T4 -A -v 192.168.80.1. The NMAP output confirmed the existence of several open TCP ports, which are inherent security loopholes: ports 21 (FTP), 80 (HTTP), 1723 (PPTP), 2880, and 8291 [1]. The presence of these open ports, particularly for protocols like FTP, makes the network vulnerable to intrusion for direct remote access, unauthorized data exfiltration, or the infiltration of malware [1]. The identification of these specific, exploitable vectors—such as the open port 21 used by FTP—empirically justified the need for targeted Snort rules to detect and alert on unauthorized connection attempts to these services.

Action Plan

To realize the security posture outlined in the solution, a structured technical action plan was followed:

a. IP Address Configuration: Initial configuration of the Internet Protocol (IP) Address settings on the dedicated server.
b. Snort Installation: Installation of the core Snort application.
c. Database Implementation: Integration and implementation of the database backend required for Snort to store and manage log data.
d. Rule Configuration: Configuration and deployment of specialized Snort rules tailored to the detection requirements of the local network.
e. Trigger Mechanism Development: Downloading and installing the MySQL User Defined Functions (UDF) library. This library is essential for enabling database triggers, which are required to execute the external scripting mechanism responsible for sending the alert notifications via Whatsapp.
f. Web Interface Deployment: Downloading and installing Acidbase, a PHP-based analysis console, to provide a web interface for alert visualization and management.
g. System Validation: Execution of comprehensive attack testing procedures against the deployed network security system.

Solution

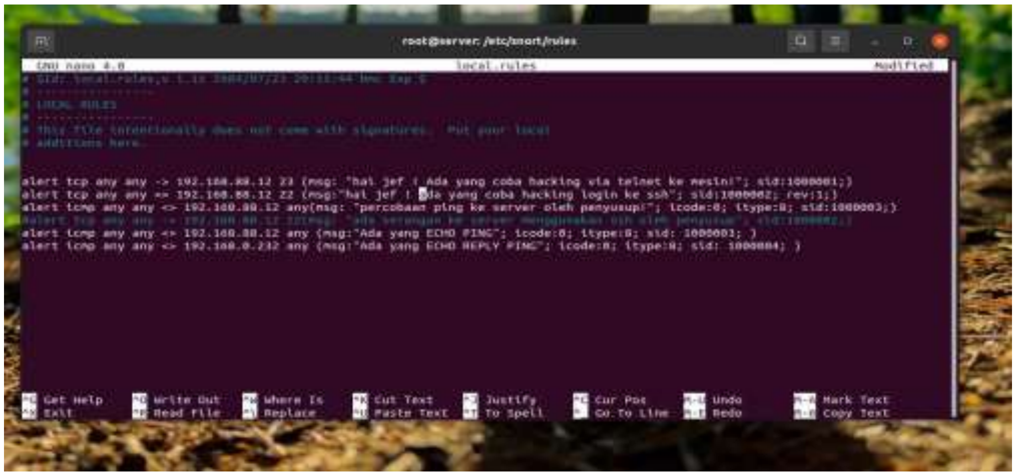
The proposed solution for the Palembang City Tourism Office involved the establishment of a robust network security framework. This framework centers on building a network security monitoring system using Snort as an IDS for early attack detection, coupled with the delivery of immediate notifications via the Whatsapp application. The successful deployment of this solution was expected to yield several tangible benefits: The network administrator would receive instantaneous notification during active attack periods, enabling a rapid security response. The administrator would be empowered to initiate timely remedial action upon receiving information regarding network anomalies or compromise. The IDS implementation was anticipated to include an explicit or implicit capability to manage and close exposed ports, utilizing the integrated firewall features associated with the IDS setup.

Results

Following the initial system implementation and deployment, continuous evaluation was performed. This process revealed the necessity of refining and augmenting the Snort rule set to effectively secure the network against specific organizational threats. The evaluation prioritized the addition of custom rules that would trigger immediate alerts for the administrator upon detection of targeted attack types.

Addition of New Rules in Snort

The customization of detection logic was achieved by editing the local rule file located at /etc/snort/rules/local.rules using the GNU nano text editor (Figure 3).



```
  GNU nano 4.8                          local.rules                          Modified
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# ----------------
# LOCAL RULES
# ----------------
# This file intentionally does not come with signatures.  Put your local
# additions here.


alert tcp any any -> 192.168.88.12 23 (msg: "hai jef ! Ada yang coba hacking via telnet ke mesin!"; sid:1000001;)
alert tcp any any -> 192.168.88.12 22 (msg:"hai jef ! Ada yang coba hacking login ke ssh"; sid:1000002; rev:1;)
alert icmp any any <> 192.168.88.12 any(msg: "percobaan ping ke server oleh penyusup!"; icode:0; itype:8; sid:1000003;)
#alert tcp any any -> 192.168.88.12 22(msg: "ada serangan ke server menggunakan ssh oleh penyusup"; sid:1000002;)
alert icmp any any <> 192.168.88.12 any (msg:"Ada yang ECHO PING"; icode:0; itype:8; sid: 1000003; )
alert icmp any any <> 192.168.0.232 any (msg:"Ada yang ECHO REPLY PING"; icode:0; itype:0; sid: 1000004; )
```

Figure 3. Adding Snort Rules

This rule augmentation focused on common infiltration and resource exhaustion vectors. The following primary detection rules were integrated:

Attack Vector	Snort Protocol Rule	Target IP/Port	Custom Alert Message (msg)	Custom SID
Telnet Access	alert tcp any any -> 192.168.88.12 23	Port 23	"halo jef! Ada yang coba hacking via telnet ke mesin"	1000001

SSH Login Attempt	alert tcp any any -> 192.168.88.12 22	Port 22	"halo jef! Ada yang coba hacking login ke ssh"	1000002
ICMP Ping/DDOS Probe	alert icmp any any -> 192.168.88.12 any	Any	"percobaan ping ke server oleh penyusup"	1000003
ICMP ECHO Reply (DDOS)	alert icmp any any -> 192.168.88.12 any	Any	"Ada yang ECHO REPLY PING"	1000004

The assignment of custom Snort IDs (SID) beginning at 1000001 ensures that these organization-specific rules are distinct and do not clash with standard vendor-supplied rule sets, confirming robust configuration management. Furthermore, the deliberate inclusion of personalized language, such as "halo jef!", within the alert messages is a key design choice aimed at improving operational efficiency. By prioritizing clear, human-centric messaging over rigid technical formality, the system significantly reduces the cognitive load on the network administrator, thereby accelerating the triage and response time to an active security event.

Re-testing of Security Attacks Related to Network Security Evaluation

A comprehensive suite of security tests was re-executed to validate the effectiveness and responsiveness of the newly configured Snort ruleset. SSH Security Testing (Figure 4 login SSH): An attempt was made to access the server (IP 192.168.8.6) using the Secure Shell (SSH) protocol via the terminal command line interface (CMD).

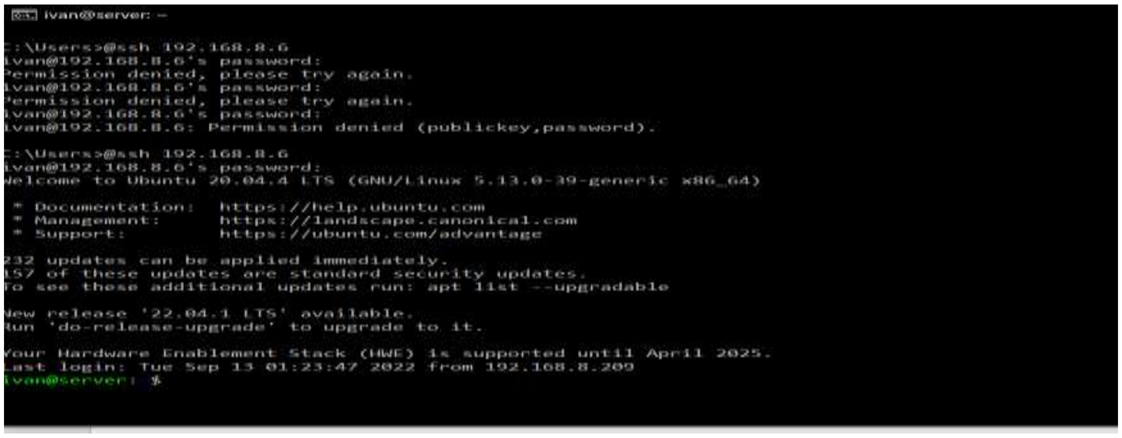
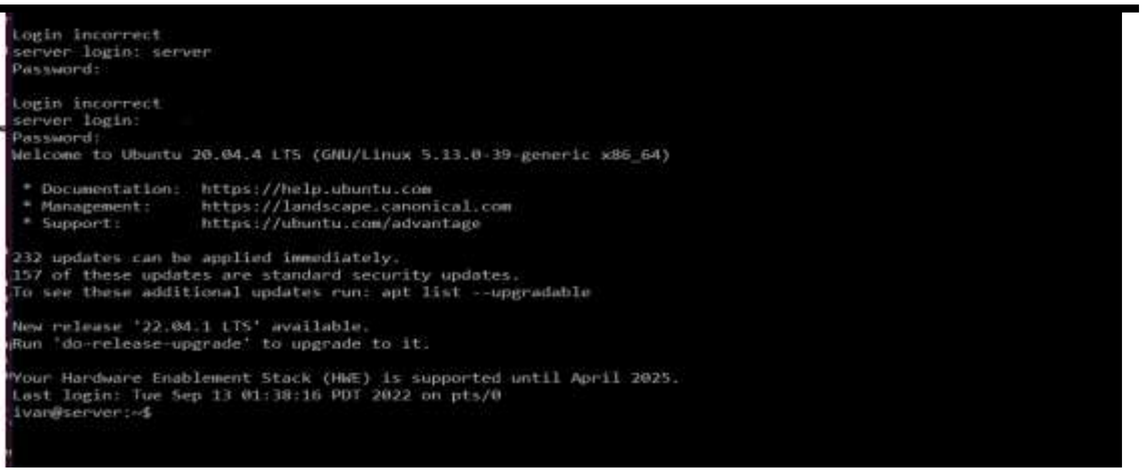

```
ivan@server: ~

:\Users>@ssh 192.168.8.6
ivan@192.168.8.6's password:
Permission denied, please try again.
ivan@192.168.8.6's password:
Permission denied, please try again.
ivan@192.168.8.6's password:
ivan@192.168.8.6: Permission denied (publickey,password).

:\Users>@ssh 192.168.8.6
ivan@192.168.8.6's password:
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.13.0-39-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

232 updates can be applied immediately.
157 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

New release '22.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Tue Sep 13 01:23:47 2022 from 192.168.8.209
ivan@server: $
```

Figure 4. Testing Using SSH Login

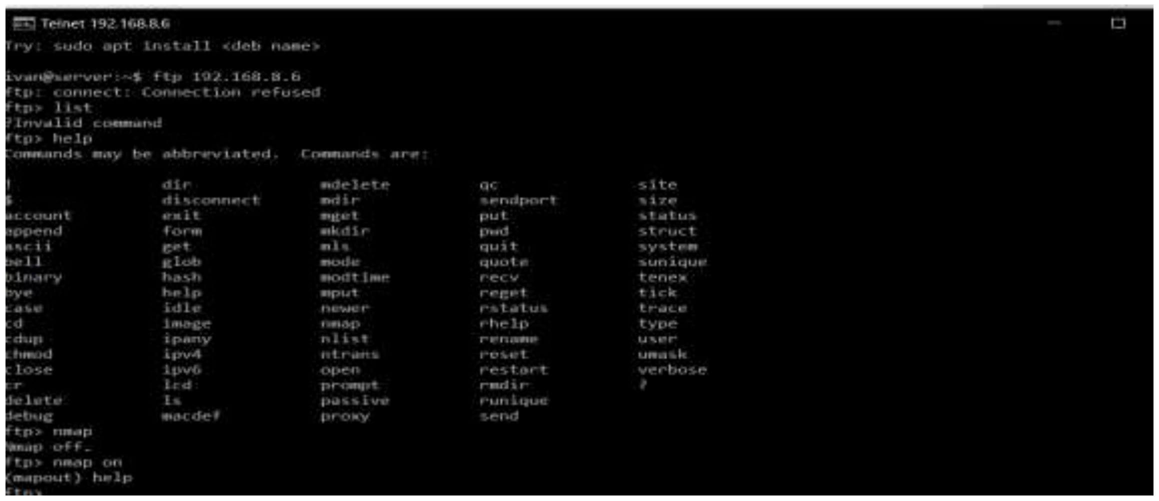
This test simulated a brute force login attempt, assessing the server's response and, crucially, confirming that the IDS triggered the corresponding alert upon detection of the access attempt. Telnet Security Testing (Figure 5): Testing was performed by attempting connection to port 23, the standard port for the Telnet protocol.



```
Login incorrect
server login: server
Password:

Login incorrect
server login:
Password:
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.13.0-39-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

232 updates can be applied immediately.
157 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

New release '22.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Tue Sep 13 01:38:16 PDT 2022 on pts/0
ivan@server:~$
```

Figure 5. Testing via Telnet

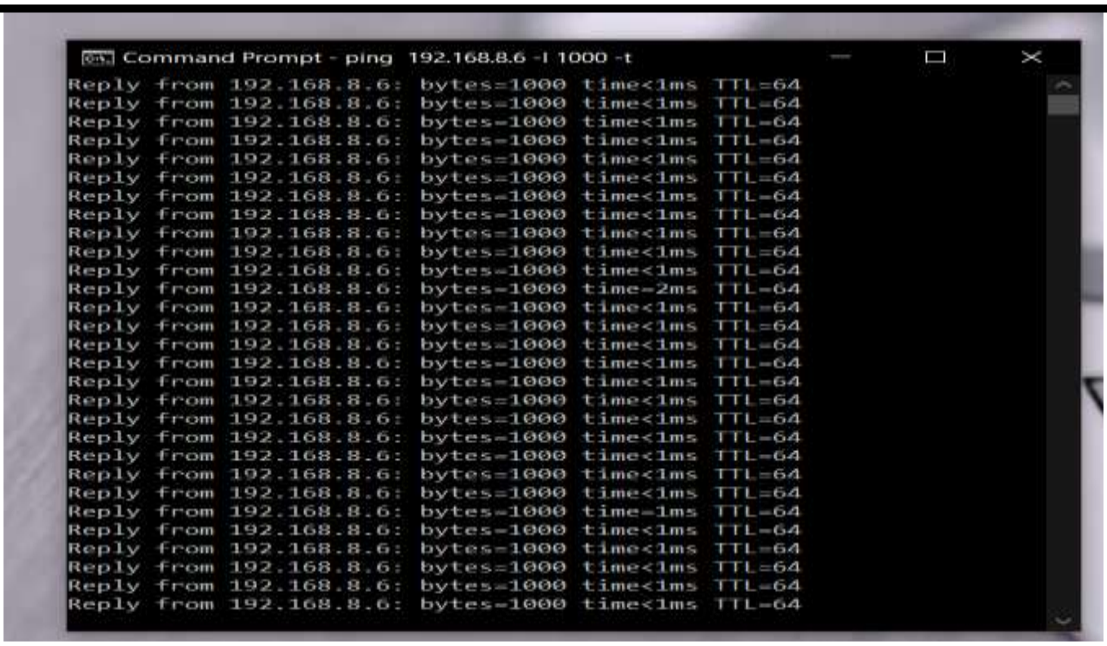
Given that Telnet utilizes an unencrypted channel, it is a high-risk vector. The objective was to confirm that the new rule provided immediate notification upon any connection probe to this vulnerability, enabling the administrator to block the access attempt or manually close the port. FTP Security Testing (Figure 6): The test simulated unauthorized access via the File Transfer Protocol (FTP), which targets port 21, often utilized for shared data access.



```
Telnet 192.168.8.6
Try: sudo apt install <deb name>

ivan@server:~$ ftp 192.168.8.6
ftp: connect: Connection refused
ftp> list
?Invalid command
ftp> help
Commands may be abbreviated.  Commands are:

!               dir             mdelete         qc              site
$               disconnect      mdir            sendport        size
account         exit            mget            put             status
append          form            mkdir           pwd             struct
ascii           get             mls             quit            system
bell            glob            mode            quote           sunique
binary          hash            modtime         recv            tenex
bye             help            mput            reget           tick
case            idle            newer           rstatus         trace
cd              image           nmap            rhelp           type
cdup            ipany           nlist           rename          user
chmod           ipv4            ntrans          reset           umask
close           ipv6            open            restart         verbose
cr              lcd             prompt          rmdir           ?
delete          ls              passive         runique
debug           macdef          proxy           send
ftp> nmap
Nmap off.
ftp> nmap on
(mapout) help
ftp>
```

Figure 6. Testing FTP

Although the connection was refused by the server, the test was designed to verify the successful generation of a Snort alert for the FTP connection attempt, protecting against unauthorized data exfiltration or infiltration attempts. DDOS Attack Testing (Figure 7): A simulation of a Distributed Denial of Service (DDOS) attack was conducted. This involved continuous, high-volume ping flooding against the server using the command ping 192.168.8.6 -l 1000 -t (using a large packet size of 1000 bytes).



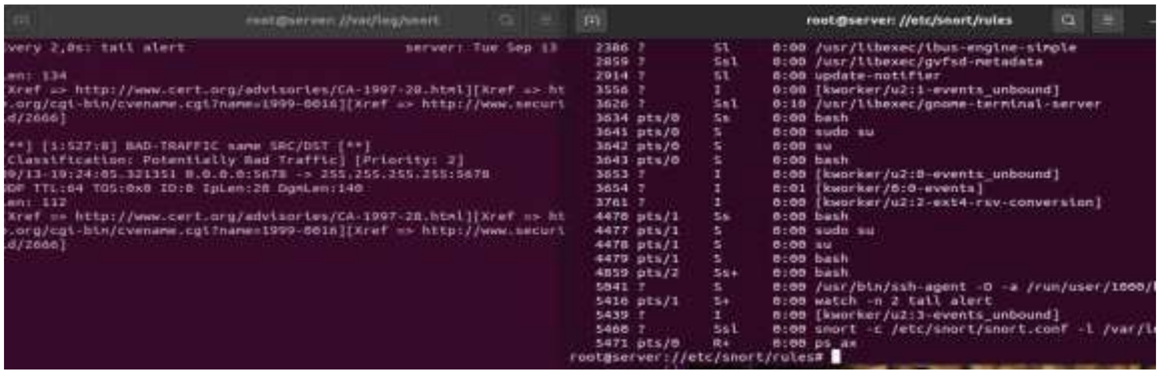
```
Command Prompt - ping  192.168.8.6 -l 1000 -t
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time=2ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time=1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
Reply from 192.168.8.6: bytes=1000 time<1ms TTL=64
```

Figure 7. DDoS Attack

This test directly addresses the historical security failure of the Palembang City Tourism Office, validating the system's ability to detect bulk network activity that can lead to server resource exhaustion and service unavailability.

Implementation Results

The implemented system successfully demonstrated continuous real-time monitoring. When operating in alert mode, the server displayed live traffic data and active alerts in the server log (Figure 8).



```
root@server: //var/log/snort
very 2,0s: tail alert                                 server: Tue Sep 13

en: 134
Xref => http://www.cert.org/advisories/CA-1997-28.html][Xref => ht
.org/cgi-bin/cvename.cgi?name=1999-0016][Xref => http://www.securi
d/2666]

**] [1:527:8] BAD-TRAFFIC same SRC/DST [**]
Classification: Potentially Bad Traffic] [Priority: 2]
9/13-19:24:05.321351 0.0.0.0:5678 -> 255.255.255.255:5678
DP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:140
en: 112
Xref => http://www.cert.org/advisories/CA-1997-28.html][Xref => ht
.org/cgi-bin/cvename.cgi?name=1999-0016][Xref => http://www.securi
d/2666]
```

```
root@server: //etc/snort/rules
 2386 ?        Sl     0:00 /usr/libexec/ibus-engine-simple
 2859 ?        Ssl    0:00 /usr/libexec/gvfsd-metadata
 2914 ?        Sl     0:00 update-notifier
 3556 ?        I      0:00 [kworker/u2:1-events_unbound]
 3626 ?        Ssl    0:10 /usr/libexec/gnome-terminal-server
 3634 pts/0    Ss     0:00 bash
 3641 pts/0    S      0:00 sudo su
 3642 pts/0    S      0:00 su
 3643 pts/0    S      0:00 bash
 3653 ?        I      0:00 [kworker/u2:0-events_unbound]
 3654 ?        I      0:01 [kworker/0:0-events]
 3761 ?        I      0:00 [kworker/u2:2-ext4-rsv-conversion]
 4476 pts/1    Ss     0:00 bash
 4477 pts/1    S      0:00 sudo su
 4478 pts/1    S      0:00 su
 4479 pts/1    S      0:00 bash
 4859 pts/2    Ss+    0:00 bash
 5041 ?        S      0:00 /usr/bin/ssh-agent -D -a /run/user/1000/
 5416 pts/1    S+     0:00 watch -n 2 tail alert
 5439 ?        I      0:00 [kworker/u2:3-events_unbound]
 5466 ?        Ssl    0:00 snort -c /etc/snort/snort.conf -l /var/l
 5471 pts/0    R+     0:00 ps ax
root@server://etc/snort/rules#
```

Figure 8. Network Monitoring

This successful log stream confirms that the foundational Snort process is operational and accurately capturing and filtering network packets according to the custom rules.

Detailed Attack Detection Analysis

The Snort alert logs confirmed that each simulated attack successfully triggered the defined custom rule: SSH Evaluation the system successfully logged an alert for the SSH attempt targeting port 22 on the server IP 192.168.88.12. The log displayed the custom message: "hello jef! Ada penyusup via ssh ke mesin," confirming immediate detection of the attack vector. Telnet Evaluation An alert was triggered for the Telnet connection attempt on port 23. The server log confirmed the detection and displayed the custom message: "halo jef! Ada penyusup via telnet ke mesin." This provided crucial real-time situational awareness, allowing the administrator to potentially block the source IP or mitigate the vulnerability associated with port 23. FTP Evaluation Despite the connection refusal, the system successfully triggered an alert for the FTP attempt on port 21, displaying the message: "Ada yang coba FTP ke mesini." This verifies the IDS capability to detect unauthorized data transfer probes. DDOS Evaluation the DDOS simulation, involving continuous large-packet ping flooding, was instantly detected. The system triggered the "ICMP PING" alert and classified the activity as "Misc activity" with Priority 3. Crucially, the system accurately identified and logged the attacking source IP address, providing the necessary data for automated or manual blocking. The ability of the rule to detect such a high volume of packets is vital, as persistent flooding could exhaust system resources (RAM, Processor, Hardisk), leading to a system crash and server downtime.

Detailed Notification Analysis (Whatsapp Alerts)

The most significant operational success was the reliable, instantaneous transmission of security alerts to the administrator's mobile device via Whatsapp, validating the real-time response capability of the system.

Table 2: Summary of Intrusion Detection System (IDS) Attack Detection and Whatsapp Notification Results

Attack Type	**Targeted Port**	**Custom Alert Message**	**Source IP/Server IP**	**Notification Status (Figure)**
Telnet Login Access	23	"halo jef! Ada penyusup via telnet ke mesin"	192.168.88.1 -> 192.168.88.12	Successfully delivered (Figure 13)
SSH Brute Force Login	22	"halo jef! Ada penyusup via ssh ke mesin"	192.168.88.1 -> 192.168.88.12	Successfully delivered (Figure 14)
FTP (File Transfer Protocol) Access	21	"halo jef! Ada penyusup via FTP ke mesin"	192.168.88.1 -> 192.168.88.12	Successfully delivered (Figure 15)
DDOS/Ping Flood Attack	ICMP Type 8	"ICMP PING"	192.168.88.1 -> 192.168.88.12	Successfully delivered (Figure

16)

Telnet Attack Notification The Whatsapp notification successfully presented the customized text alert ("halo jef! Ada penyusup via telnet ke mesin"), confirmed the high priority (Priority 0 in the application layer), and specified the attack target port (23) and the specific attacker and server IP addresses. SSH Attack Notification A similar alert was generated for the SSH test, displaying the custom message ("halo jef! Ada penyusup via ssh ke mesin") and identifying the attack on port 22, ensuring the administrator knew immediately that a secured protocol login attempt was underway. FTP Attack Notification The notification for the FTP attack contained the custom message ("halo jef! Ada penyusup via FTP ke mesin") and confirmed the attack on port 21, highlighting the potential data breach risk. DDOS Attack Notification The notification for the high-volume DDOS attack was delivered, categorizing the event as "ICMP PING" and classifying it with Priority 3. Crucially, the message included the attacker's source IP address (192.168.88.1), along with the destination server IP (192.168.88.12) and the technical details (byte type 8, indicating ping activity).

The successful transmission of the attacker's source IP address within the real-time Whatsapp alert is an achievement with paramount operational significance. This specific data point transforms the notification from a general warning into an immediate, actionable intelligence report. The administrator is now equipped to execute swift countermeasures, such as blacklisting the malicious source IP address on the network's router or firewall, without the critical delay involved in logging into the server and manually sifting through log files. The system's demonstrated ability to process and instantaneously transmit high-volume alerts (such as those generated during the DDOS test) confirms the low latency and high reliability of the custom notification integration script.

Conclusion and Recommendations

The research successfully designed, implemented, and validated the deployment of Snort as a cost-effective and operationally efficient Intrusion Detection System tailored for the specific security requirements of the Palembang City Tourism Office network. The study yielded the following key scientific conclusions regarding network security management: High-Fidelity Detection: The implemented Intrusion Detection System effectively achieved its primary objective, enabling the network administrator to ascertain immediately if any server within the network is under attack, conditional upon the efficacy of the established, customized rule set. Instantaneous Alerting Capability: The seamless integration with the Whatsapp application establishes a highly functional, real-time alert mechanism, which surpasses the limitations inherent in traditional, passive log monitoring. This mechanism is crucial for mitigating risks in environments with limited dedicated human resources. Vulnerability Coverage: The designed IDS ruleset demonstrated comprehensive detection effectiveness across key documented threat vectors, including successful validation against Port Scanning (as rules were derived from NMAP results), unauthorized FTP Login attempts, SSH Brute Force attempts, and high-volume DDOS Attacks. The successful adoption of this open-source, community-supported solution (Snort, Ubuntu) carries a substantial economic implication for public sector information technology management. It provides empirical evidence that achieving a sophisticated, real-time security monitoring and response capability does not necessitate prohibitive investment in proprietary hardware or expensive commercial security software. This model offers a scalable and financially pragmatic security upgrade path for other government agencies operating under similar resource constraints.

Based on the empirical findings and the current system architecture, the following future research and development recommendations are proposed to enhance the system's resilience, functionality, and level of automation : Rule Coverage Expansion: Continuous refinement and development of the Snort ruleset are imperative. Future work should focus on expanding the detection signature base to cover a broader array of sophisticated and emerging attack vectors, such as advanced persistent threats (APTs) and zero-day exploits, thereby significantly enhancing the system's overall threat intelligence capabilities. Transition to Intrusion Prevention System (IPS): The system's architecture should be technically evolved from a purely reactive IDS function (detection and notification) to a full Intrusion Prevention System (IPS). This upgrade is necessary to enable the platform to automatically mitigate or block harmful attacks upon detection—for example, by automatically updating firewall rules—thereby transitioning the security posture from proactive awareness to automated, active defense. Development of Interactive Communication Features: Future iterations should incorporate features that permit the administrator to communicate with the system via the same Whatsapp notification medium. This functionality would facilitate operational tasks such as querying real-time system status, remotely checking network health metrics, or securely triggering basic mitigation actions, ultimately leading to a more autonomous and sophisticated security management platform.

References

All about Palembang Tourism Destination.” https://www.palembang-tourism.com/ (Accessed: April 26, 2020).

Mutaqin, A. F. (2016). Rancang Bangun Sistem Monitoring Keamanan Jaringan Prodi Teknik Informatika Melalui SMS Alert dengan Snort. Journal Name, 1(1), 6.

Mutaqin - 2016 - Rancang Bangun Sistem Monitoring Keamanan Jaringan.pdf.

Action Research / Penelitian Tindakan,” CHANDRA, July 05, 2008. https://chandrax.wordpress.com/2008/07/05/action-research-penelitian-tindakan/ (Accessed: May 03, 2020).

Afrizal, M. R., Amri, J., & Sari, J. (2025). Network Security Analysis Using Snort With Intrusion Detection System (IDS) Method for Computer Network Security Optimization. Jurnal Teknologi Rekayasa Informasi dan Komputer**, Vol. 8(2), pp. 22–29.

Purba, E., & Efendi, W. (2021). Implementation of Intrusion Detection System (IDS) and Snort Community Rules to Detect Types of Network Attacks. International Journal of Computer Applications (IJCA), Vol. 183(42).

Fatah, B., & Utomo, I. C. (2023). Implementation of IDS Using Snort with Barnyard2 Visualization for Network Monitoring. International Journal of Computer and Information System (IJCIS), Vol. 4(4).

AbdulRaheem, M., Oladipo, I. D., Imoize, A. L., Awotunde, J. B., Lee, C. C., Balogun, G. B., & Adeoti, J. O. (2023). Machine Learning Assisted Snort and Zeek in Detecting DDoS Attacks in Software-Defined Networking. International Journal of Information Technology, Vol. 16, pp. 1627–1643.

Bashah, N. S. K., Simbas, T. S., Janom, N., & Aris, S. R. S. (2023). Proactive DDoS Attack Detection in Software-Defined Networks with Snort Rule-Based Algorithms. International Journal of Advanced Technology and Engineering Exploration, Vol. 10(105).

Alosimy, H., Alzaidi, J., Alajmani, S., & Soh, B. (2025). An Algorithm for Detecting Brute Force Attacks on FTP and SSH Services Utilizing Deep Learning with Probabilistic Neural Networks. International Journal of Recent Technology and Engineering (IJRTE), Vol. 13(6).

Alotibi, N., & Alshammari, M. (2023). Deep Learning-Based Intrusion Detection: A Novel Approach for Identifying Brute-Force Attacks on FTP and SSH Protocol. International Journal of Advanced Computer Science and Applications (IJACSA), Vol. 14(6).

Kamarudin Shah, M. F., Md-Arshad, M., Abdul Samad, A., & Ghaleb, F. A. (2023). Comparing FTP and SSH Password Brute Force Attack Detection Using k-Nearest Neighbour and Decision Tree in Cloud Computing.International Journal of Innovative Computing (IJIC), Vol. 13(1).

Tandi, R. S., Siaulhak, A., & Jumarniati, J. (2025). Implementasi Intrusion Detection System (IDS) Snort Sebagai Sistem Keamanan Menggunakan WhatsApp dan Telegram Sebagai Media Notifikasi. Jurnal Ilmu Komputer Revolusioner, Vol. 9(6).

Magnusson, L., Iqbal, S., Elm, P., & Dalipi, F. (2025). Information Security Governance in the Public Sector: Investigations, Approaches, Measures, and Trends. International Journal of Information Security, Vol. 24, Article 177.