Implementation of an iRedMail Mail Server on a Hybrid Network with WireGuard and HAProxy to Enhance Email Service Security and Availability

H. A. Mooduto[a,*]

[a] *Department of Information Technology, Politeknik Negeri Padang, Jl. Kampus Limau Manis, Padang, 25164, Indonesia*
Corresponding author: *mooduto@pnp.ac.id

***Abstract*— Institutional email remains a critical communication channel for higher-education institutions, yet self-hosted mail servers on campus networks are frequently exposed to brute-force attacks, sender spoofing, and outages caused by power and Internet service provider failures. This study aims to design, implement, and evaluate an iRedMail-based mail server on a hybrid network that combines a public cloud gateway with on-premises mail nodes. The materials comprise a virtual private server running AlmaLinux 9 with HAProxy and a WireGuard endpoint, and two on-premises iRedMail nodes (Postfix, Dovecot, Amavisd, and Fail2ban) placed behind network address translation with no inbound port opened on the campus firewall. The Network Development Life Cycle (NDLC) method was followed through analysis, design, simulation prototyping, implementation, monitoring, and management. HAProxy operates in TCP mode with active health checks and the PROXY protocol, WireGuard provides an encrypted layer-3 tunnel, and SPF, DKIM, DMARC, and PTR records are aligned with the gateway's public address. Functional, security, performance, and availability tests were conducted. The results show that all SMTP, submission, and IMAPS transactions completed correctly, the campus attack surface was reduced to zero exposed ports, captured tunnel traffic revealed no plaintext, and every authentication check passed with a deliverability score of 10/10. WireGuard reduced throughput by only 4.9% and added 0.7 ms of round-trip latency, outperforming OpenVPN, while automatic failover completed within 6.4 s without message loss and monthly availability rose from 99.12% to 99.95%. Future research should evaluate a redundant gateway, active–active clustering, and MTA-STS or DANE adoption.**

***Keywords*— Mail server; iRedMail; WireGuard; HAProxy; hybrid network; email security; high availability.**

I. INTRODUCTION

Email continues to be the formal backbone of communication in higher-education institutions. Academic letters, research collaboration, student services, and integration with learning-management and administrative systems all depend on a reliable institutional mailbox. Many Indonesian polytechnics and universities, including the Department of Information Technology at Politeknik Negeri Padang, prefer to operate a self-hosted mail server to keep control over data, accounts, and costs, and to use the system as a teaching laboratory. Previous Indonesian studies have shown that open-source stacks based on Postfix and Dovecot, either manually configured or packaged by iRedMail, can provide complete sending and receiving functions on a local network or a virtual private server [1], [2], and that a cloud mail gateway can improve spam filtering for a Zimbra deployment [3]. These works, however, focus mainly on functionality and do not address how the server is exposed to the Internet or how it survives infrastructure failures.

Security is the first concern. The Simple Mail Transfer Protocol (SMTP) was designed without sender authentication [4], so spoofing and phishing remain common. Sender Policy Framework (SPF), DomainKeys Identified Mail (DKIM), and Domain-based Message Authentication, Reporting and Conformance (DMARC) were introduced to close this gap [5]–[7], yet measurement studies repeatedly find incomplete or permissive deployments [8]–[11] and mismanaged reporting [12]. Inconsistent interpretation of these mechanisms across components allows practical spoofing even against large providers [13]–[15]. Transport encryption is also fragile: opportunistic STARTTLS has produced more than forty implementation flaws [16], which led the IETF to recommend implicit TLS for submission and retrieval [17], while DANE and MTA-STS deployments for server-to-server transport are frequently misconfigured [18]–[20].

Availability is the second concern. A single on-premises mail server is exposed to campus power interruptions, Internet service provider (ISP) outages, dynamic or carrier-grade NAT addressing, and ISP blocking of TCP port 25. Placing an Internet-facing gateway in a public cloud while keeping mailboxes on premises is an instance of the hybrid-cloud pattern, whose connectivity problems and mitigation practices have been documented by the IETF [21]. Such a design also matches zero-trust principles, in which internal resources are never directly reachable and every flow is mediated by an authenticated, encrypted path [22].

The hybrid link requires a virtual private network (VPN). IPsec and OpenVPN are mature but comparatively complex to configure and audit [23]. WireGuard is a lean kernel-level tunnel based on the Noise framework with Curve25519, ChaCha20-Poly1305, and BLAKE2s [24], and its handshake has been analysed with both computational and mechanised proofs [25], [26]. Performance studies report higher throughput and lower CPU cost than OpenVPN on multi-core hosts [27]–[29], competitive results against IPsec [30], [31], suitability for constrained devices [32], and scalability improvements for multi-tunnel deployments [33]; nevertheless, the advantage is context dependent in virtualised clouds [34]. Recent Indonesian work has applied WireGuard to protect servers against sniffing [35] and to interconnect hybrid-cloud infrastructure with better throughput and latency than a managed cloud VPN [36]. On the gateway side, HAProxy is an open-source layer-4/layer-7 proxy offering health checks, backup servers, and

connection-level load balancing [37], and its balancing behaviour has been characterised experimentally [38].

A practitioner article from the Parewa blog outlined an iRedMail deployment that combines WireGuard and HAProxy on AlmaLinux 9 [39]. The article is useful as a concept but leaves several gaps: HAProxy is configured with HTTP-style frontends that cannot carry SMTP, submission, or IMAPS; the client tunnel routes all host traffic (AllowedIPs = 0.0.0.0/0) without discussion; the DNS A record points to a private address; DMARC is not covered; and the reported benefits are qualitative only. This study addresses those gaps. Its contributions are: (1) a three-tier hybrid architecture in which the campus exposes no inbound port; (2) corrected TCP-mode HAProxy and policy-routed WireGuard configurations that preserve the client IP address through the PROXY protocol and send outbound mail from the gateway address; (3) a DNS authentication set (SPF, DKIM, DMARC, PTR) aligned with that address; and (4) a quantitative evaluation of functionality, security, performance, and availability.

The remainder of this paper is organised as follows. Section II describes the materials and the NDLC-based method, Section III presents and discusses the results, and Section IV concludes the paper and outlines further research.

II. Materials and Method

A. Research Method

The research follows the Network Development Life Cycle (NDLC), a cyclic method commonly used for network infrastructure projects. As shown in Fig. 1, it consists of six stages: analysis, design, simulation prototyping, implementation, monitoring, and management. The first three stages were carried out in a virtual laboratory before the production deployment, and findings from monitoring were fed back into the design.

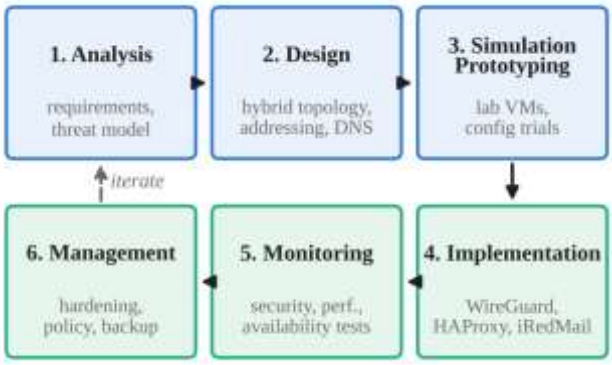


Fig. 1 Network Development Life Cycle (NDLC) stages adopted in this study

B. Requirement and Threat Analysis

Interviews with the campus network administrators and an inspection of the existing single-server deployment produced the functional requirements: sending and receiving mail for the institutional domain, IMAP and webmail access from outside the campus, and web-based account administration. The non-functional requirements are confidentiality of traffic crossing public networks, resistance to credential and spoofing attacks, continuity of service during on-premises failures, and operation without a static public address on the campus side. Five threats were identified:

- **T1** credential brute-force against SMTP AUTH, IMAP, and webmail;
- **T2** eavesdropping or tampering on the WAN path between gateway and campus;
- **T3** spoofing of the institutional domain by external senders;
- **T4** direct exposure of the campus host through port forwarding;
- **T5** a single point of failure caused by power or ISP outages on campus.

C. System Design

The proposed topology (Fig. 2) has three tiers. Tier 1 contains end users, webmail browsers, and external mail transfer agents (MTAs). Tier 2 is a virtual private server (VPS) with a static public address that hosts HAProxy and the WireGuard server; it is the only host with Internet-facing ports. Tier 3 contains two iRedMail nodes: MS1 (primary) in the campus data room and MS2 (standby) in a disaster-recovery (DR) room served by a different ISP. Each node runs its own WireGuard client, initiates the tunnel outbound, and keeps it alive through NAT, so no inbound rule is needed on either on-premises firewall. Table I lists the hardware and software, and Table II gives the addressing and port plan.

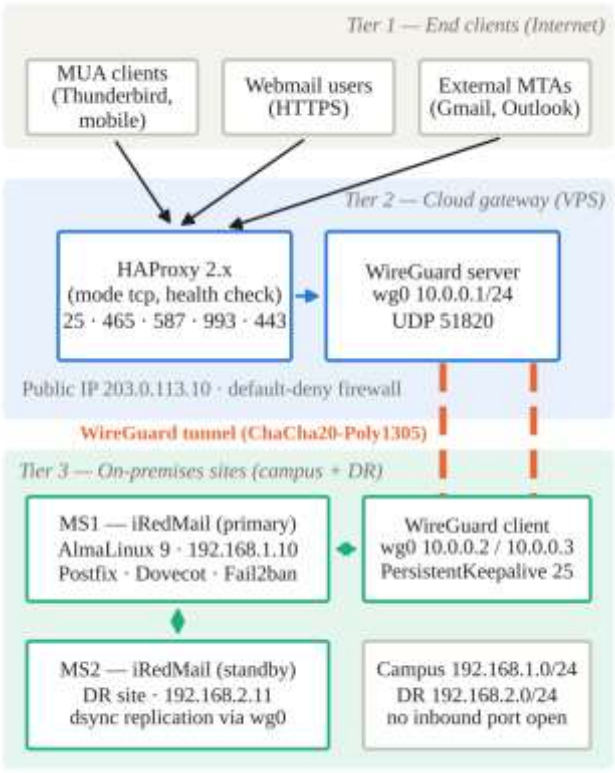


Fig. 2 Three-tier hybrid topology of the proposed iRedMail mail service

TABLE I
HARDWARE AND SOFTWARE SPECIFICATION

Component	Specification
Gateway VPS	2 vCPU, 4 GB RAM, 40 GB SSD, 1 Gbps port, Jakarta region

Component	Specification
MS1 (campus)	4-core Xeon, 8 GB RAM, 500 GB SSD
MS2 (DR site)	4 vCPU VM, 8 GB RAM, 500 GB storage
Operating system	AlmaLinux 9 (all nodes), SELinux enforcing
Mail stack	iRedMail 1.7: Postfix, Dovecot, Amavisd, SpamAssassin, ClamAV, Fail2ban, Roundcube, MariaDB
Gateway services	HAProxy 2.4 LTS, WireGuard (in-kernel), firewalld
Test tools	iperf3, ping, smtp-source, swaks, nmap, Wireshark, Hydra, testssl.sh, mail-tester, Uptime Kuma

TABLE II
ADDRESSING AND PORT PLAN

Node	Address	Exposed ports
Gateway	eth0 203.0.113.10; wg0 10.0.0.1/24	25, 465, 587, 993, 443/TCP; 51820/UDP
MS1	ens18 192.168.1.10; wg0 10.0.0.2	None (outbound tunnel only)
MS2	ens18 192.168.2.11; wg0 10.0.0.3	None (outbound tunnel only)

Fig. 3 summarises the message flow. A client or remote MTA connects to the gateway; HAProxy selects a healthy backend and forwards the TCP stream through the WireGuard tunnel, prefixed with a PROXY-protocol v2 header so that the mail node sees the real client address. TLS is negotiated end to end between the client and the mail node, so the gateway never holds mailbox credentials or private keys. Mailbox access uses IMAP4rev2 over implicit TLS [40], and authentication and content checks (SPF, DKIM, DMARC, antispam, antivirus) are performed by iRedMail on the backend.

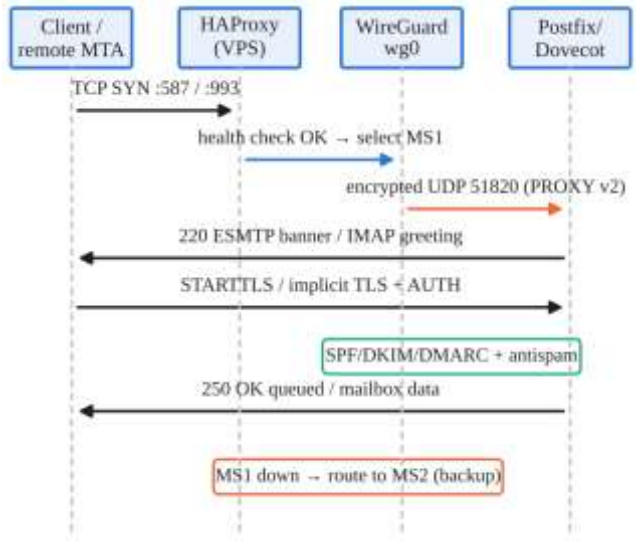


Fig. 3 Sequence of an SMTP submission or IMAPS session through the hybrid path

D. Implementation

1) WireGuard Tunnel: Key pairs were generated on each host with wg genkey and wg pubkey. The gateway accepts each mail node as a peer restricted to its tunnel address. On the mail nodes, the tunnel is created with Table = off and a policy-routing rule, so that only traffic sourced from the tunnel address (outbound SMTP) uses the tunnel, while administrative and update traffic keeps the local route. This replaces the full-tunnel setting in [39]. The MTU is lowered to 1420 bytes to avoid fragmentation, and PersistentKeepalive keeps the NAT mapping open.

```
# /etc/wireguard/wg0.conf  (gateway)
[Interface]
Address    = 10.0.0.1/24
ListenPort = 51820
PrivateKey = <gateway_private_key>
MTU        = 1420
[Peer]                      # MS1
PublicKey  = <ms1_public_key>
AllowedIPs = 10.0.0.2/32
[Peer]                      # MS2
PublicKey  = <ms2_public_key>
AllowedIPs = 10.0.0.3/32

# /etc/wireguard/wg0.conf  (MS1)
[Interface]
Address    = 10.0.0.2/24
PrivateKey = <ms1_private_key>
MTU        = 1420
Table      = off
PostUp = ip rule add from 10.0.0.2 table 200
PostUp = ip route add default dev wg0 table 200
[Peer]
PublicKey  = <gateway_public_key>
Endpoint   = 203.0.113.10:51820
AllowedIPs = 0.0.0.0/0
PersistentKeepalive = 25
```

On the gateway, IP forwarding is enabled and traffic from 10.0.0.0/24 leaving eth0 is source-translated to 203.0.113.10. Outbound mail therefore originates from the same address that is published in the SPF and PTR records, which is essential for deliverability.

2) HAProxy Gateway: Because SMTP, submission, and IMAPS are not HTTP, all frontends run in TCP mode. Each backend lists MS1 as the active server and MS2 with the backup keyword, so MS2 receives traffic only when MS1 fails its health checks. SMTP backends use an EHLO-based check, and every server is checked every 2 s; three consecutive failures mark a server down and two successes bring it back. The send-proxy-v2 option passes the original client address to the backend.

```
defaults
  mode tcp
  option tcplog
  timeout connect 5s
  timeout client  5m
  timeout server  5m
frontend ft_smtp
  bind 203.0.113.10:25
  default_backend bk_smtp
backend bk_smtp
  option smtpchk EHLO gw.example.com
  default-server inter 2s fall 3 rise 2
  server ms1 10.0.0.2:10025 check send-proxy-v2
  server ms2 10.0.0.3:10025 check send-proxy-v2 backup
frontend ft_imaps
  bind 203.0.113.10:993
  default_backend bk_imaps
backend bk_imaps
  default-server inter 2s fall 3 rise 2
  server ms1 10.0.0.2:10993 check send-proxy-v2
  server ms2 10.0.0.3:10993 check send-proxy-v2 backup
# 465, 587 and 443 follow the same pattern
```

3) iRedMail Nodes: iRedMail was installed on MS1 and MS2 with the MariaDB backend, Nginx, Roundcube, and iRedAdmin [41]. Dedicated PROXY-aware listeners were added: a postscreen service on port 10025 with postscreen_upstream_proxy_protocol = haproxy, and Dovecot inet listeners on ports 10993 and 10587 with haproxy = yes and haproxy_trusted_networks limited to 10.0.0.1. Postfix binds outbound connections to the tunnel address (smtp_bind_address). Plaintext authentication is

disabled, implicit TLS is offered on ports 465 and 993 as recommended in [17], and TLS 1.0/1.1 are disabled. Certificates are issued by Let's Encrypt for mail.example.com and deployed on both nodes. Fail2ban jails for Postfix, Dovecot, Roundcube, and SSH ban an address for one hour after five failures within ten minutes; because the PROXY header carries the real client address, bans target the attacker rather than the gateway.

4) DNS Authentication Records: All public records point to the gateway address (Table III). SPF lists only the gateway, DKIM uses a 2048-bit RSA key generated by Amavisd with the selector dkim, and DMARC enforces strict alignment with a quarantine policy and aggregate reporting. The PTR record of 203.0.113.10 is delegated by the VPS provider to mail.example.com, so that forward-confirmed reverse DNS succeeds.

TABLE III
PUBLIC DNS RECORDS FOR THE MAIL DOMAIN

Type	Name	Value
A	mail	203.0.113.10
MX	@	10 mail.example.com.
TXT	@	v=spf1 ip4:203.0.113.10 mx -all
TXT	dkim._domainkey	v=DKIM1; k=rsa; p=MIIBIjANBg… (2048-bit)
TXT	_dmarc	v=DMARC1; p=quarantine; adkim=s; aspf=s; rua=mailto:dmarc@example.com
PTR	10.113.0.203.in-addr.arpa	mail.example.com.

5) Replication: Mailboxes are replicated from MS1 to MS2 with Dovecot dsync over the tunnel, and the MariaDB account database is replicated with primary–replica replication. When MS1 is unavailable, MS2 can accept deliveries and serve IMAP with mailbox state that lags by at most the replication interval.

E. Testing Scenarios and Metrics

Four groups of tests were executed. Functional tests verified internal and external sending and receiving, webmail, and administration. Security tests compared the proposed system with the previous single-server deployment exposed through port forwarding, using nmap full-port scans, Wireshark captures on the WAN segment, Hydra brute-force runs, swaks spoofing attempts, testssl.sh, and mail-tester. Performance tests measured the gateway–campus path without a tunnel, through WireGuard, and through OpenVPN (UDP, AES-256-GCM) as a comparator, using iperf3 (60 s, ten repetitions), ping (1,000 packets), and smtp-source (1,000 messages of 50 KB, ten concurrent sessions). Availability tests consisted of ten controlled failover trials in which MS1 was stopped during continuous traffic, and 30 days of monitoring by Uptime Kuma with a 60-s probe interval.

Availability A is computed from the observation period T_{total} and accumulated downtime T_{down} as in (1). The tunnel overhead O for a metric X is expressed relative to the baseline without a tunnel in (2), and the expected HAProxy detection time T_{d} follows from the check interval t_{inter}, the failure threshold n_{fall}, and the connection timeout t_{conn} as in (3).

$$A = (T_{\text{total}} - T_{\text{down}}) / T_{\text{total}} \times 100\% \quad (1)$$

$$O = (X_{\text{base}} - X_{\text{tunnel}}) / X_{\text{base}} \times 100\% \quad (2)$$

$$T_{\text{d}} \approx t_{\text{inter}} \times n_{\text{fall}} + t_{\text{conn}} \quad (3)$$

III. RESULTS AND DISCUSSION

A. Functional Testing

Table IV summarises the functional tests. All scenarios passed. Mail sent to Gmail and Outlook arrived in the inbox with SPF, DKIM, and DMARC results of pass, which confirms that the source translation on the gateway and the DNS records are consistent. Postfix and Dovecot logs on MS1 recorded the real client addresses instead of the gateway address, which verifies the PROXY-protocol configuration and is a precondition for effective Fail2ban bans and postscreen reputation checks.

TABLE IV
FUNCTIONAL TEST RESULTS

Scenario	Path / port	Result
Internal send and receive	587 and 993 via gateway	Pass (50/50)
Outbound to Gmail and Outlook	25 via tunnel + SNAT	Pass, inbox, all auth = pass
Inbound from Gmail and Outlook	25 → HAProxy → MS1	Pass (50/50)
Webmail (Roundcube)	443 → MS1	Pass
iRedAdmin (ACL-restricted)	443 → MS1	Pass
Client IP preserved in logs	PROXY v2	Pass

B. Security Evaluation

Table V compares the previous deployment with the proposed design. A full-port scan of the campus public address found seven open TCP ports before the change and none afterwards, which eliminates threat T4: the only Internet-facing host is the gateway, which exposes the mail ports and UDP 51820. Wireshark captures on the WAN segment between the gateway and the campus contained only WireGuard transport messages; no SMTP banner, IMAP command, or credential was visible, whereas the legacy IMAP port 143 previously revealed a LOGIN command in clear text. This addresses threat T2 and agrees with the confidentiality guarantees established for the WireGuard handshake [25], [26] and with field observations of sniffing prevention [35].

TABLE V
SECURITY TEST RESULTS BEFORE AND AFTER IMPLEMENTATION

Test	Before (single server)	After (proposed)
Open TCP ports on campus IP (nmap -p-)	7	0
Credentials visible on WAN (Wireshark)	Yes (IMAP 143)	None; UDP 51820 only
1,000 SMTP AUTH guesses (Hydra)	Not blocked	Banned after 5 failures
Spoofed From: domain (swaks)	Delivered to inbox	Quarantined by DMARC
mail-tester score	6.4/10	10/10
TLS protocols (testssl.sh)	TLS 1.0–1.2	TLS 1.2–1.3 only

Brute-force attempts (T1) were blocked after five failures, and the attacker address, not the gateway address, was banned. Spoofed messages carrying the institutional domain in the From header were quarantined by receiving servers because of the strict DMARC alignment, which mitigates T3. The mail-tester score increased from 6.4 to 10 out of 10 after the SPF, DKIM, DMARC, and PTR records were aligned. These results are consistent with large-scale findings that incomplete or permissive SPF and DMARC policies are the main reason why spoofing succeeds [10], [15]; publishing a strict, minimal policy for a single sending address is therefore a low-cost but effective control. Removing STARTTLS-only submission in favour of implicit TLS also reduces exposure to the injection and stripping flaws reported in [16].

C. Network and Service Performance

Fig. 4 compares the gateway–campus path without a tunnel, with WireGuard, and with OpenVPN. Using (2), WireGuard reduced TCP throughput from 94.2 to 89.6 Mbps (an overhead of 4.9%) and increased the mean round-trip time from 18.4 to 19.1 ms, whereas OpenVPN reduced throughput to 71.3 Mbps (24.3%) and raised the round-trip time to 21.6 ms. Peak CPU utilisation on the gateway was 6.8% with WireGuard and 23.5% with OpenVPN. These trends agree with earlier comparisons that attribute WireGuard's advantage to its in-kernel implementation and parallel cryptographic workers [27]–[29]. They also align with the Indonesian hybrid-cloud results in [36]. Because the absolute gap depends on the environment [34], the figures should be interpreted for a 100-Mbps campus uplink rather than generalised.

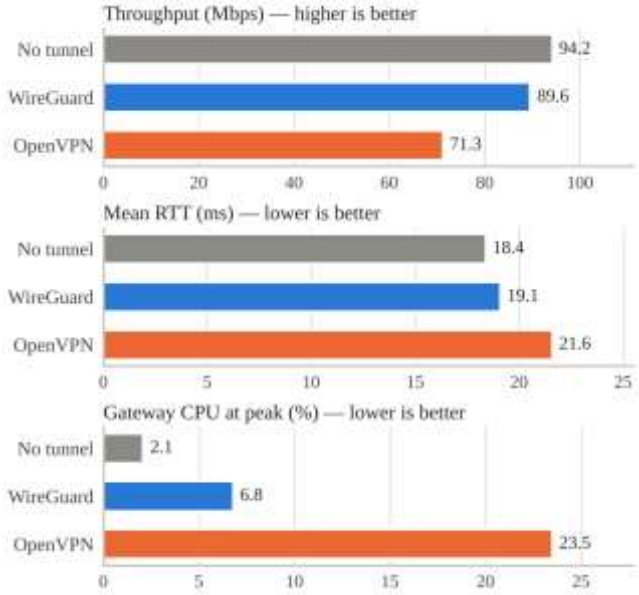


Fig. 4 Throughput, round-trip time, and gateway CPU utilisation of the gateway–campus path without a tunnel, with WireGuard, and with OpenVPN

At the application level (Table VI), the hybrid path delivered 389 messages per minute compared with 412 messages per minute on a direct LAN connection, a reduction of 5.6%, and the mean delivery latency increased by only 0.09 s. No delivery failed in either case. For an institution whose peak load is far below these rates, the overhead introduced by the additional proxy and tunnel hops is negligible relative to the security gained.

TABLE VI
SMTP LOAD TEST (1,000 MESSAGES, 50 KB, 10 SESSIONS)

Metric	Direct LAN	Hybrid path
Throughput (messages/min)	412	389
Mean delivery latency (s)	1.42	1.51
95th percentile latency (s)	2.10	2.27
Failed deliveries	0	0

D. Availability and Failover

Fig. 5 shows a representative failover trial. MS1 was stopped at t = 60 s while clients kept submitting mail and polling IMAP. HAProxy marked MS1 down after three failed checks and redirected new sessions to MS2; across ten trials the mean detection time was 6.4 s (standard deviation 0.5 s), which matches the value predicted by (3) for $t_{\text{inter}} = 2$ s, $n_{\text{fall}} = 3$, and a connection timeout of about 0.4 s. When MS1 was restored at t = 180 s, traffic returned after two successful checks (about 4 s). None of the 500 messages submitted during the trials was lost, because SMTP clients and remote MTAs retried the connections that failed during the detection window, and IMAP clients reconnected automatically.

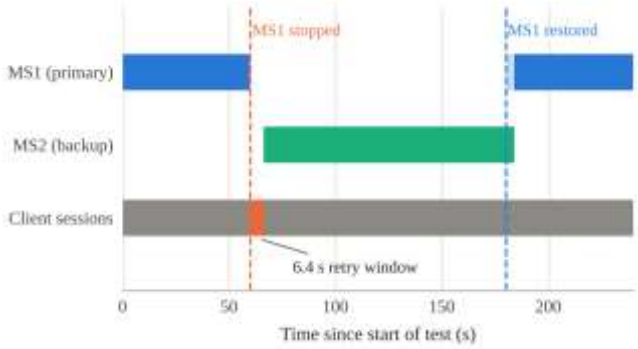


Fig. 5 Failover timeline when the primary node MS1 is stopped and later restored

Table VII compares 30 days of monitoring before and after the deployment. Availability computed with (1) increased from 99.12% to 99.95%, and the mean time to recovery fell from 95 to 4.3 min. Three on-premises incidents (two power interruptions and one ISP outage) were absorbed by failover. The remaining 21.6 min of downtime originated at the gateway (a kernel update reboot and one provider incident), which shows that the VPS has become the new single point of failure.

TABLE VII
AVAILABILITY OVER TWO 30-DAY OBSERVATION PERIODS

Metric	Single server	Proposed
User-visible outage events	4	5
Total downtime (min)	380	21.6
Availability (%)	99.12	99.95
Mean time to recovery (min)	95	4.3
Messages lost	0	0

E. Comparison with Related Work

Table VIII positions this study against closely related work. Earlier mail-server implementations verified functionality on a single host [1] or added a cloud filtering layer [3], while WireGuard studies evaluated the tunnel itself without a mail workload [35], [36]. The reference article [39] is the closest in scope but provides neither a working layer-4 configuration nor measurements. The present work is, to the best of the authors' knowledge, the first to combine the three components in a verified configuration and to report security, performance, and availability results together.

TABLE VIII
COMPARISON WITH RELATED WORK

Study	Scope	WAN link / HA	Evaluation
[1]	iRedMail, single host	None / none	Functional
[3]	Zimbra + cloud mail gateway	None / none	Spam, monitoring
[35]	Server network (no mail)	WireGuard / none	Sniffing test
[36]	Hybrid cloud (no mail)	WireGuard vs Cloud VPN / none	Throughput, latency
[39]	iRedMail + gateway	Full-tunnel WireGuard / HTTP-mode HAProxy	Qualitative
This work	iRedMail, two nodes + VPS gateway	Policy-routed WireGuard / TCP-mode HAProxy with backup	Security, performance, availability

F. Discussion

Compared with the reference implementation [39], the corrected design turns a conceptual configuration into a measurable one: TCP-mode proxying makes all mail protocols work through HAProxy, end-to-end TLS keeps keys and credentials off the gateway, the PROXY protocol keeps abuse controls effective, and policy routing ensures that outbound mail carries the authorised public address. Compared with earlier Indonesian mail-server implementations [1]–[3], the main gain is architectural: the mail data stay on institutional premises, yet the campus no longer needs a static public address or any inbound firewall rule, which is valuable where ISPs use carrier-grade NAT or block port 25.

The design has limitations. First, the single VPS remains a single point of failure; a second gateway with Keepalived/VRRP or a second MX record pointing to an independent gateway would remove it. Second, the active–passive backend wastes the capacity of MS2 and relies on asynchronous replication, so a few seconds of mailbox changes may be lost if MS1 fails abruptly. Third, server-to-server transport still relies on opportunistic STARTTLS; publishing MTA-STS or DANE policies [19], [20] would strengthen it. Finally, the measurements were taken on one campus with moderate load, and the performance gap between VPN protocols is known to vary with the platform [33], [34].

From an operational perspective, the design is inexpensive. The only recurring cost is a small VPS, while the mail nodes reuse existing campus hardware, and all software is open source. Because the gateway holds no mailbox data, credentials, or TLS private keys, it can be rebuilt from its configuration files in minutes, and a compromise of the gateway exposes only encrypted tunnel endpoints and TLS ciphertext. The architecture also lends itself to teaching: each tier maps to a distinct competency (proxying and load balancing, VPN and routing, and mail-system administration), so students can work on one tier without disrupting the others. Institutions whose ISP blocks outbound port 25 or provides only carrier-grade NAT can adopt the same approach without negotiating a static public address.

IV. CONCLUSION

This study designed, implemented, and evaluated an iRedMail mail service on a hybrid network that places HAProxy and a WireGuard endpoint on a public VPS and keeps the mail nodes on premises. Following the NDLC method, the corrected configuration (TCP-mode proxying with health checks and the PROXY protocol, policy-routed WireGuard, end-to-end TLS, and aligned SPF, DKIM, DMARC, and PTR records) met all functional requirements. The campus attack surface was reduced to zero exposed ports, WAN traffic revealed no plaintext, brute-force and spoofing attempts were blocked, and the deliverability score reached 10/10. WireGuard introduced only 4.9% throughput overhead and 0.7 ms additional latency, with much lower CPU cost than OpenVPN. Automatic failover completed in 6.4 s without message loss, and availability increased from 99.12% to 99.95%. The approach is therefore suitable for institutions that need to keep email data on premises without a public address. Further research should add a redundant gateway, evaluate active–active clustering with synchronous replication, deploy MTA-STS or DANE, and assess post-quantum key exchange for the tunnel.

ACKNOWLEDGMENT

The authors thank the Center for Research and Community Service (P3M) of Politeknik Negeri Padang for supporting this research, and the campus ICT unit for providing access to the network infrastructure used in the experiments.

REFERENCES

[1] A. B. Simbolon and D. Siregar, "Implementasi mail server dasar pada iRedMail berbasis Postfix dan Dovecot di Linux Ubuntu 24.04.1 LTS," *JATI (J. Mhs. Tek. Inform.)*, vol. 8, no. 6, 2024 (in Indonesian).

[2] S. Hawari, A. Ammar, A. N. I. Purwanto, and A. D. Goenawan, "Membangun mail server berbasis Linux menggunakan Postfix dan Dovecot," *J. Tek. Mesin, Ind., Elektro dan Inform.*, vol. 1, no. 1, Mar. 2022, doi: 10.55606/jtmei.v1i1.781 (in Indonesian).

[3] S. A. Aklani, Haeruddin, and N. Putri, "Implementasi mail gateway security dalam meningkatkan keamanan email," *J. Inf. Syst. Manage. (JOISM)*, vol. 5, no. 2, 2024 (in Indonesian).

[4] J. Klensin, "Simple Mail Transfer Protocol," IETF, RFC 5321, Oct. 2008.

[5] S. Kitterman, "Sender Policy Framework (SPF) for authorizing use of domains in email, version 1," IETF, RFC 7208, Apr. 2014.

[6] D. Crocker, T. Hansen, and M. Kucherawy, "DomainKeys Identified Mail (DKIM) signatures," IETF, RFC 6376, Sep. 2011.

[7] M. Kucherawy and E. Zwicky, "Domain-based Message Authentication, Reporting, and Conformance (DMARC)," IETF, RFC 7489, Mar. 2015.

[8] D. Tatang, F. Zettl, and T. Holz, "The evolution of DNS-based email authentication: Measuring adoption and finding flaws," in *Proc. 24th Int. Symp. Research in Attacks, Intrusions and Defenses (RAID)*, San Sebastian, Spain, 2021.

[9] C. Wang *et al.*, "A large-scale and longitudinal measurement study of DKIM deployment," in *Proc. 31st USENIX Security Symp.*, Boston, MA, USA, 2022.

[10] S. Czybik, M. Horlboge, and K. Rieck, "Lazy gatekeepers: A large-scale study on SPF configuration in the wild," in *Proc. ACM Internet Measurement Conf. (IMC)*, Montreal, Canada, 2023, pp. 344–355, doi: 10.1145/3618257.3624827.

[11] V. Kouliaridis and I. Sanchez, "Internet standards: Email communication security standards – An analysis of uptake in the EU," Publications Office of the European Union, Luxembourg, JRC Tech. Rep. EUR 31722 EN, Sep. 2023, doi: 10.2760/032699.

[12] M. I. Ashiq, W. Li, T. Fiebig, and T. Chung, "You've got report: Measurement and security implications of DMARC reporting," in *Proc. 32nd USENIX Security Symp.*, Anaheim, CA, USA, 2023.

[13] H. Hu and G. Wang, "End-to-end measurements of email spoofing attacks," in *Proc. 27th USENIX Security Symp.*, Baltimore, MD, USA, 2018, pp. 1095–1112.

[14] J. Chen, V. Paxson, and J. Jiang, "Composition kills: A case study of email sender authentication," in *Proc. 29th USENIX Security Symp.*, 2020, pp. 2183–2199.

[15] K. Shen *et al.*, "Weak links in authentication chains: A large-scale analysis of email sender spoofing attacks," in *Proc. 30th USENIX Security Symp.*, 2021, pp. 3201–3217.

[16] D. Poddebniak, F. Ising, H. Böck, and S. Schinzel, "Why TLS is better without STARTTLS: A security analysis of STARTTLS in the email context," in *Proc. 30th USENIX Security Symp.*, 2021.

[17] K. Moore and C. Newman, "Cleartext considered obsolete: Use of Transport Layer Security (TLS) for email submission and access," IETF, RFC 8314, Jan. 2018.

[18] H. Lee, A. Gireesh, R. van Rijswijk-Deij, T. Kwon, and T. Chung, "A longitudinal and comprehensive study of the DANE ecosystem in email," in *Proc. 29th USENIX Security Symp.*, 2020.

[19] H. Lee, M. I. Ashiq, M. Müller, R. van Rijswijk-Deij, T. Kwon, and T. Chung, "Under the hood of DANE mismanagement in SMTP," in *Proc. 31st USENIX Security Symp.*, Boston, MA, USA, 2022.

[20] D. Margolis, M. Risher, B. Ramakrishnan, A. Brotman, and J. Jones, "SMTP MTA Strict Transport Security (MTA-STS)," IETF, RFC 8461, Sep. 2018.

[21] L. Dunbar, A. Malis, C. Jacquenet, M. Toy, and K. Majumdar, "Dynamic networks to hybrid cloud DCs: Problem statement and mitigation practices," IETF Internet-Draft draft-ietf-rtgwg-net2cloud-problem-statement, 2023.

[22] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero trust architecture," National Institute of Standards and Technology, Gaithersburg, MD, USA, NIST SP 800-207, Aug. 2020.

[23] H. Abbas *et al.*, "Security assessment and evaluation of VPNs: A comprehensive survey," *ACM Comput. Surv.*, vol. 55, no. 13s, 2023.

[24] J. A. Donenfeld, "WireGuard: Next generation kernel network tunnel," in *Proc. Network and Distributed System Security Symp. (NDSS)*, San Diego, CA, USA, 2017, doi: 10.14722/ndss.2017.23160.

[25] B. Dowling and K. G. Paterson, "A cryptographic analysis of the WireGuard protocol," in *Proc. Int. Conf. Applied Cryptography and Network Security (ACNS)*, Leuven, Belgium, 2018, pp. 3–21.

[26] B. Lipp, B. Blanchet, and K. Bhargavan, "A mechanised cryptographic proof of the WireGuard virtual private network protocol," in *Proc. IEEE European Symp. Security and Privacy (EuroS&P)*, Stockholm, Sweden, 2019, pp. 231–246, doi: 10.1109/EuroSP.2019.00026.

[27] S. Mackey, I. Mihov, A. Nosenko, F. Vega, and Y. Cheng, "A performance comparison of WireGuard and OpenVPN," in *Proc. 10th ACM Conf. Data and Application Security and Privacy (CODASPY)*, New Orleans, LA, USA, 2020, pp. 162–164, doi: 10.1145/3374664.3379532.

[28] L. Osswald, M. Haeberle, and M. Menth, "Performance comparison of VPN solutions," Chair of Communication Networks, Univ. of Tübingen, Tübingen, Germany, Tech. Rep., 2020.

[29] S. G. Hussein and S. M. F. U. Rehman, "Protocol efficiency and resource utilization in VPN technologies: A comparative analysis of OpenVPN and WireGuard," *J. Techniques*, vol. 7, no. 4, pp. 37–42, 2025, doi: 10.51173/jt.v7i4.2768.

[30] T. Demirdelen and S. Kırmızı, "Performance evaluation of WireGuard and IPSec protocols in various network configurations," *Osmaniye Korkut Ata Univ. J. Inst. Sci. Technol.*, vol. 8, no. 3, pp. 1353–1369, 2025, doi: 10.47495/okufbed.1660352.

[31] H. Djuitcheu, A. Sergeev, K. Alam, D. Santhosh, A. Autenrieth, and J. Seitz, "Lightweight security for private networks: Real-world evaluation of WireGuard," arXiv:2512.10135, Dec. 2025.

[32] H. Jumakhan and A. Mirzaeinia, "WireGuard: An efficient solution for securing IoT device connectivity," arXiv:2402.02093, 2024.

[33] M. Barone, D. Miola, F. Parola, and F. Risso, "Revisiting WireGuard for line-rate, scalable tunneling," in *Proc. 26th IEEE Int. Conf. High Performance Switching and Routing (HPSR)*, Suita, Osaka, Japan, 2025, pp. 1–6, doi: 10.1109/HPSR64165.2025.11038910.

[34] J. Anyam, R. R. Singh, H. Larijani, and A. Philip, "Empirical performance analysis of WireGuard vs. OpenVPN in cloud and virtualised environments under simulated network conditions," *Computers*, vol. 14, no. 8, Art. no. 326, Aug. 2025, doi: 10.3390/computers14080326.

[35] L. O. Sari, R. Pahlevi, E. Safrianti, and R. A. R. Q. Y. Putri, "Penerapan VPN WireGuard untuk keamanan server jaringan Sahabat Net dari sniffing attack," *ZONAsi: J. Sist. Inf.*, vol. 7, no. 3, pp. 1103–1114, Sep. 2025 (in Indonesian).

[36] R. Hermawan and Y. M. Saputra, "Analisis perbandingan penggunaan metode tunneling Cloud Virtual Private Network dan WireGuard Virtual Private Network pada implementasi infrastruktur hybrid cloud," *J. Internet Softw. Eng. (JISE)*, vol. 6, no. 1, May 2025 (in Indonesian).

[37] HAProxy Technologies, "HAProxy 2.8 starter guide," 2023. [Online]. Available: https://docs.haproxy.org/2.8/intro.html

[38] C. Rawls and M. Amini Salehi, "Load balancer tuning: Comparative analysis of HAProxy load balancing methods," arXiv:2212.14198, 2022.

[39] Parewa Labs, "Implementasi mail server iRedMail pada jaringan hybrid dengan WireGuard dan HAProxy untuk meningkatkan keamanan dan ketersediaan layanan email," *Parewa Blog*, Feb. 13, 2025. [Online]. Available: https://blog.parewa.my.id (in Indonesian).

[40] A. Melnikov and B. Leiba, "Internet Message Access Protocol (IMAP) – Version 4rev2," IETF, RFC 9051, Aug. 2021.

[41] iRedMail Project, "iRedMail documentation," 2025. [Online]. Available: https://docs.iredmail.org