PERANCANGAN SISTEM KEAMANAN PADA UBUNTU SERVER TERHADAP *BUFFER OVERFLOW* DAN *FORMAT STRING* MENGGUNAKAN FAIL2BAN DAN APPARMOR

SKRIPSI

WAHYU PRIAMBODO

2207421048

PROGRAM STUDI TEKNIK MULTIMEDIA DAN JARINGAN

JURUSAN TEKNIK INFORMATIKA DAN KOMPUTER

POLITEKNIK NEGERI JAKARTA

2026

PERANCANGAN SISTEM KEAMANAN PADA UBUNTU SERVER TERHADAP *BUFFER OVERFLOW* DAN *FORMAT STRING* MENGGUNAKAN FAIL2BAN DAN APPARMOR

SKRIPSI

Dibuat untuk Melengkapi Syarat-Syarat yang Diperlukan untuk Memperoleh Diploma Empat Politeknik

WAHYU PRIAMBODO

2207421048

PROGRAM STUDI TEKNIK MULTIMEDIA DAN JARINGAN

JURUSAN TEKNIK INFORMATIKA DAN KOMPUTER

POLITEKNIK NEGERI JAKARTA

2026

SURAT PERNYATAAN BEBAS PLAGIARISME

Saya yang bertanda tangan di bawah ini:

Nama	:	Wahyu Priambodo
NIM	:	2207421048
Jurusan / Program Studi	:	Teknik Informatika dan Komputer / Teknik Multimedia dan Jaringan
Judul Skripsi	:	Perancangan Sistem Keamanan pada Ubuntu Server Terhadap *Buffer Overflow* dan *Format String* Menggunakan Fail2Ban dan AppArmor

Menyatakan dengan sebenarnya bahwa skripsi ini benar-benar merupakan hasil karya saya sendiri, bebas dari peniruan terhadap karya dari orang lain. Kutipan pendapat dan tulisan orang lain ditunjuk sesuai dengan cara-cara penulisan karya ilmiah yang berlaku.

Apabila di kemudian hari terbukti atau dapat dibuktikan bahwa dalam skripsi ini terkandung ciri-ciri plagiat dan bentuk-bentuk peniruan lain yang dianggap melanggar peraturan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Depok, 21 Juni 2026

Yang membuat pernyataan

(Wahyu Priambodo)

NIM. 2207421048

LEMBAR PENGESAHAN

Skripsi diajukan oleh :

Nama	:	Wahyu Priambodo
NIM	:	2207421048
Jurusan / Program Studi	:	Teknik Informatika dan Komputer / Teknik Multimedia dan Jaringan
Judul Skripsi	:	Perancangan Sistem Keamanan pada Ubuntu Server Terhadap *Buffer Overflow* dan *Format String* Menggunakan Fail2Ban dan AppArmor

Telah diuji oleh tim penguji dalam sidang skripsi pada hari Kamis, Tanggal 2, Bulan Juli, Tahun 2026 dan dinyatakan **LULUS**.

Disahkan oleh

Pembimbing I	: Iik Muhamad Malik Matin, S.Kom., M.T.	( )
Penguji I	: Defiana Arnaldy, S.Tp., M.Si.	( )
Penguji II	: Ayu Rosyida Zain, S.ST, M.T.	( )
Penguji III	: Ariawan Andi Suhandana, S.Kom., M.T.I.	( )

Mengetahui :

Jurusan Teknik Informatika dan Komputer

Ketua

Dr. Anita Hidayati, S.Kom., M.Kom.

NIP. 197908032003122003

KATA PENGANTAR

Puji dan syukur selalu penulis panjatkan kepada Allah *subhanahu wa ta'ala*, Tuhan Yang Maha Esa, karena atas rahmat dan karunia-Nya, penulis mampu menyelesaikan skripsi ini yang berjudul "Perancangan Sistem Keamanan pada Ubuntu Server Terhadap *Buffer Overflow* dan *Format String* menggunakan Fail2Ban dan AppArmor". Skripsi ini dibuat dengan tujuan untuk memenuhi syarat dalam memperoleh gelar Diploma Empat (D4) Politeknik. Penulis menyadari bahwa terdapat berbagai pihak yang telah mendukung, membantu, dan membimbing selama proses penelitian. Oleh karena itu, izinkan penulis untuk menyampaikan terima kasih yang sebesar-besarnya kepada:

1. Bapak Iik Muhamad Malik Matin, S.Kom., M.T., selaku dosen pembimbing penulis yang telah mendedikasikan waktu, kesabaran, dan ilmunya selama proses penelitian dan menyusun skripsi ini.
2. Komunitas Xyl0n PNJ yang senantiasa mengajak penulis untuk belajar ilmu-ilmu yang baru, terutama di dunia keamanan siber, seperti *Capture The Flag* (CTF) dan Linux *binary exploitation* (*pwn*).
3. Kedua orang tua, saudara, dan kerabat yang telah memberikan doa dan dukungan, baik moral ataupun material, serta motivasi untuk tetap semangat dalam menuntaskan skripsi ini.
4. Sahabat dan teman-teman kelas yang sudah mau berbagi ilmu dan berjuang bersama-sama selama proses mengerjakan skripsi.

Penulis menyadari bahwa masih terdapat keterbatasan pengetahuan dan pengalaman selama proses penelitian, sehingga skripsi ini jauh dari kata sempurna. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun dari pihak-pihak lain supaya skripsi ini bisa lebih baik lagi ke depannya. Akhir kata, penulis berharap agar skripsi ini dapat bermanfaat dan menambah wawasan baru bagi orang lain, terutama *security researcher*.

Depok, 21 Juni 2026

Wahyu Priambodo

SURAT PERNYATAAN PERSETUJUAN PUBLIKASI SKRIPSI UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Politeknik Negeri Jakarta, saya bertanda tangan di bawah ini:

Nama	:	Wahyu Priambodo
NIM	:	2207421048
Jurusan / Program Studi	:	Teknik Informatika dan Komputer / Teknik Multimedia dan Jaringan

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Politeknik Negeri Jakarta Hak Bebas Royalti Non-Eksklusif atas karya ilmiah saya yang berjudul :

Perancangan Sistem Keamanan pada Ubuntu Server Terhadap *Buffer Overflow* dan *Format String* Menggunakan Fail2Ban dan AppArmor

Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Non-Eksklusif ini Politeknik Negeri Jakarta berhak menyimpan, mengalihmediakan/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan skripsi saya tanpa meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Depok, 21 Juni 2026

Yang menyatakan,

(Wahyu Priambodo)

NIM. 2207421048

Perancangan Sistem Keamanan pada Ubuntu Server Terhadap *Buffer Overflow* dan *Format String* Menggunakan Fail2Ban dan AppArmor

ABSTRAK

Kerentanan memori seperti buffer overflow dan format string masih sering ditemukan hingga saat ini. Mitigasi bawaan pada binary di Linux, seperti NX, RELRO, PIE, dan Stack Canary menjadi tidak efektif apabila tidak dikonfigurasi maksimal dan dikombinasikan dengan lemahnya praktik secure coding. Penelitian ini bertujuan untuk merancang sistem keamanan berlapis yang mengombinasikan AppArmor sebagai lapisan pencegah eksploitasi di tingkat sistem operasi dan Fail2Ban sebagai lapisan pemblokiran alamat IP di tingkat jaringan. Rancangan sistem keamanan diimplementasikan pada tiga mesin virtual Ubuntu Server 24.04 dengan proteksi bertingkat, yaitu VM-1 (tanpa proteksi), VM-2 (proteksi sebagian), dan VM-3 (proteksi penuh). Penelitian ini juga memanfaatkan alat monitoring, seperti Grafana, Loki, dan Prometheus. Objek penelitian menggunakan 10 file ELF binary yang diskenariokan memiliki celah dengan variasi proteksi dan alur logika program berbeda-beda. Tahap pengujian dilakukan melalui 4 skenario berbeda, yaitu fungsionalitas, pembatasan dampak oleh AppArmor, pemblokiran alamat IP oleh Fail2Ban, dan pembanjiran koneksi. Hasil pengujian pada binary dengan profil AppArmor longgar masih memungkinkan penyerang memperoleh server secara penuh, karena tidak adanya pembatasan ruang lingkup. Di sisi lain, Fail2Ban mampu memblokir alamat IP berdasarkan pola crash berulang dan pembanjiran koneksi yang melebihi ambang batas. Pengembangan lebih lanjut bisa dengan menambahkan notifikasi alert dan memperdalam pengujian di sisi jaringan.

Kata Kunci: AppArmor, *Buffer Overflow*, CTFd, Fail2Ban, *Format String*, Grafana, *Hardening Server*, Loki, Prometheus, Ubuntu Server

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

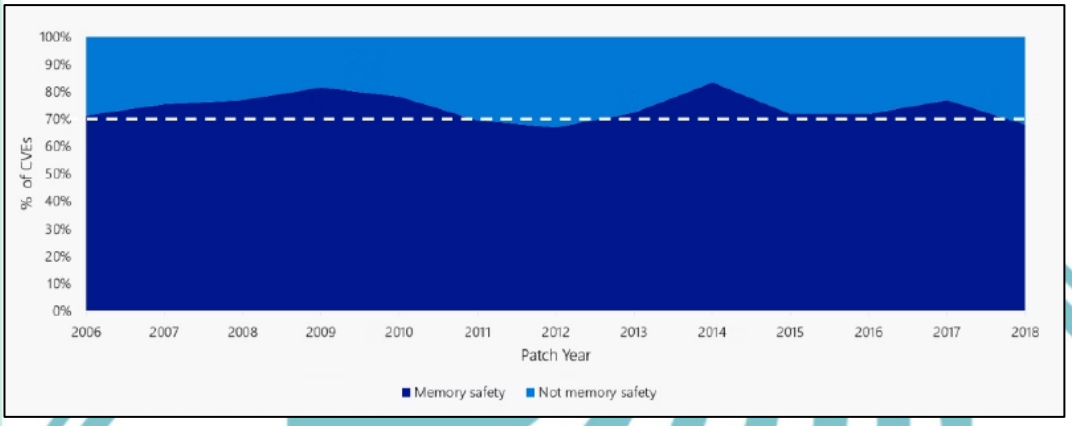
BAB I
PENDAHULUAN

1.1 Latar Belakang Masalah

Setiap aplikasi atau *binary* yang berjalan di sistem operasi berbasis Linux, termasuk Ubuntu Server, umumnya telah dilengkapi mekanisme mitigasi bawaan seperti *No Execute* (NX), *Relocation Read-Only* (RELRO), *Stack Canary* (StackGuard), dan *Position Independent Executable* (PIE) (Butt et al., 2022). Mekanisme mitigasi tersebut berfungsi untuk melindungi program dari berbagai upaya eksploitasi keamanan memori, seperti celah *buffer overflow*, *format string*, dan kerentanan memori lainnya (Butt et al., 2022). Selain itu, terdapat pula mitigasi pada level sistem operasi, yaitu *Address Space Layout Randomization* (ASLR), yang bekerja dengan merandomisasi tata letak ruang alamat proses, sehingga membuat upaya eksploitasi *binary* menjadi semakin sulit (Marco-Gisbert dan Ripoll, 2019). Permasalahan muncul ketika mekanisme mitigasi tersebut tidak dikonfigurasi secara maksimal, sehingga eksploitasi *buffer overflow* dan *format string* tetap berpeluang terjadi. Kondisi ini diperburuk apabila pengembang aplikasi tidak menerapkan praktik *secure coding* atau lemahnya *quality control*, yang menyebabkan mitigasi bawaan pada aplikasi dan ASLR menjadi tidak efektif (Cybersecurity and Infrastructure Security Agency and Federal Bureau of Investigation, 2025). Oleh karena itu, diperlukan mekanisme mitigasi tambahan untuk menutupi keterbatasan mitigasi bawaan pada program di sistem operasi berbasis Linux.

Buffer overflow merupakan celah keamanan yang diakibatkan oleh masukan (*input*) melebihi kapasitas alokasi *buffer* yang telah ditentukan, sehingga dapat menimpa bagian lain di dalam memori (George et al., 2021). Sementara itu, *format string* adalah celah keamanan serius yang dapat dimanfaatkan untuk membocorkan (*read*) maupun mengubah (*write*) data atau alamat di dalam memori (Xu et al., 2024). Kedua celah tersebut menarik untuk dikaji lebih mendalam, karena sama-sama dapat menjadi rantai kerentanan yang berujung pada eksekusi kode arbitrer (*arbitrary code execution*) (Petrean dan Coleşa, 2024).

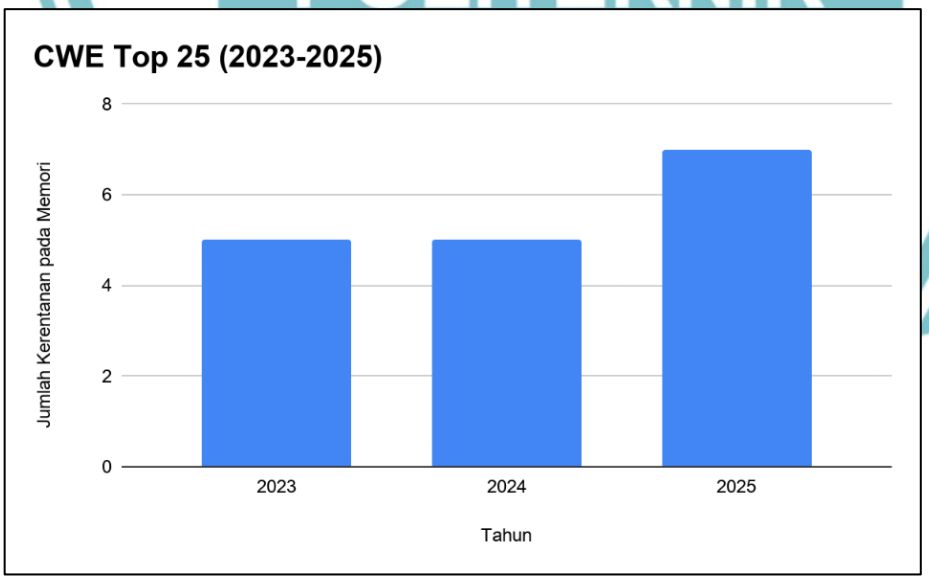
Pernyataan tersebut diperkuat oleh laporan dari Microsoft dan pada tahun 2019 yang menyatakan jika sekitar 70% isu celah keamanan didominasi oleh kerentanan pada memori (Miller, 2019). Gambar 1.1 merepresentasikan laporan dari Microsoft pada tahun 2019.



Gambar 1. 1 Dominasi *Memory Corruption Bugs* di Sistem Microsoft Tahun 2019

Sumber: (Miller, 2019)

Daftar *Top 25 Common Weakness Enumeration* (CWE) periode 2023 - 2025 menunjukkan tren peningkatan eksploitasi terhadap kerentanan memori (MITRE, 2025). Grafik tren kenaikan dapat dilihat pada Gambar 1.2. Dengan demikian, kerentanan pada memori masih menjadi ancaman yang relevan hingga saat ini.



Gambar 1. 2 Tren Kenaikan *Memory Corruption Bugs* pada 2023-2025

Sumber: (MITRE, 2025)

Objek penelitian ini menggunakan total 10 *file ELF binary* yang dirancang memiliki kerentanan *buffer overflow* dan *format string*. Kesepuluh *file binary* tersebut memiliki variasi alur logika program dan proteksi yang berbeda-beda dan harus dilewati oleh penyerang supaya bisa mendapatkan akses ke dalam sistem. Sistem keamanan yang telah dirancang akan diekspos ke penyerang dalam bentuk kompetisi *Capture The Flag* (CTF) yang dapat diakses secara publik untuk pengambilan data pengujian dan evaluasi sistem keamanan (Maulana et al., 2023). Kompetisi CTF akan memberikan kesepuluh *file binary* yang rentan ke penyerang untuk dieksploitasi hingga mendapatkan akses ke dalam sistem dan membaca file *flag*. Penyerang membutuhkan teknik-teknik yang relevan untuk eksploitasi celah *buffer overflow* dan *format string*. Teknik eksploitasi *buffer overflow* dapat berupa *Return Oriented Programming* (ROP), *Return-to-Libc* (*ret2libc*), dan lain sebagainya. Sedangkan, teknik eksploitasi *format string* dapat berupa *Global Offset Table* (GOT) *Overwrite*. Teknik-teknik apapun yang digunakan oleh penyerang dikategorikan valid selama proses pengujian asalkan tidak melakukan pelanggaran seperti *sharing flag* dan merusak platform CTF.

Ubuntu Server dipilih sebagai lingkungan pengujian pada penelitian ini karena terkenal andal dan memiliki reputasi yang stabil dan aman untuk penggunaan standar server (Erawan dan Salman, 2023). Selain itu, menurut Erawan dan Salman (2023) juga menjelaskan bahwa Ubuntu sudah memiliki fitur keamanan bawaan, salah satunya adalah AppArmor, yang dapat membantu membatasi dampak serangan. Versi 24.04 dipilih karena memiliki dukungan pemeliharaan sistem operasi yang panjang atau *Long Term Support* (LTS) hingga 5 tahun. Dukungan tersebut merepresentasikan server produksi yang stabil dan relevan dalam jangka waktu lama.

Studi literatur terhadap penelitian terdahulu dilakukan untuk menemukan celah penelitian (*research gap*). Penelitian Liu et al. (2023) dilakukan dengan tujuan mengembangkan metode deteksi dan pertahanan *buffer overflow* berbasis ekstensi *instruction set* RISC-V. Penelitian Rasyid dan Santoso (2026) kedua dilakukan dengan tujuan menemukan kerentanan *buffer overflow* pada perangkat lunak PDFCook versi 0.4.5 menggunakan teknik *fuzzing* (AFL++) yang dikombinasikan dengan *Address Sanitizer* (ASan). Penelitian Petrean dan Coleşa (2024) dilakukan

dengan tujuan merancang *tool* otomatis (PwnMaster) untuk mendeteksi sekaligus mengeksploitasi kerentanan *buffer overflow* dan *format string* pada *file ELF* menggunakan *symbolic execution* dan *pwntools*. Penelitian Xu dan Wang (2022) dilakukan dengan tujuan membangun sistem *automated exploit generation* (BofAEG) untuk mendeteksi dan menghasilkan eksploitasi *stack buffer overflow* secara otomatis berbasis *symbolic execution* dan analisis dinamis. Penelitian Kim et al. (2024) dilakukan dengan tujuan meningkatkan keamanan aplikasi berbasis bahasa C melalui metode *StackGuard*$^{+}$ sebagai alternatif interoperabel terhadap proteksi berbasis *canary* konvensional. Dari kelima literatur tersebut, belum ada penelitian yang mengkaji pendekatan *hardening* pada Ubuntu Server terhadap eksploitasi *buffer overflow* dan *format string* menggunakan Fail2Ban dan AppArmor.

Selain kelima studi literatur terkait *buffer overflow* dan *format string*, pencarian studi literatur terkait pemanfaatan alat Fail2Ban sebagai metode *Intrusion Prevention System* (IPS) juga dilakukan. Dari hasil studi literatur, Fail2Ban dapat dimanfaatkan untuk memblokir suatu alamat IP yang terindikasi melakukan pelanggaran jika melebihi ambang batas yang telah ditentukan (Balqis & Rahman, 2025). Penerapan Fail2Ban umum digunakan untuk memblokir serangan DDoS dan *brute force login* SSH (Dawamsyach et al., 2023). Studi lain yang dilakukan oleh Park et al. (2021), menunjukkan bahwa Fail2Ban mampu mendeteksi pola *brute force* melalui protokol SSH dan berhasil memblokir alamat IP penyerang. Pada penelitian ini, Fail2Ban juga dapat diterapkan untuk mendeteksi pola *crash* akibat eksploitasi *buffer overflow* atau *format string*. Di mana, pola *crash* suatu proses aplikasi akan tersimpan di *file log* sistem di Linux. Fail2Ban dapat memblokir alamat IP berdasarkan peristiwa *crash* apabila terdapat *log* lain yang menyimpan sumber alamat IP yang melakukan eksploitasi.

Mengingat eksploitasi celah *buffer overflow* dan *format string* dapat berujung pada eksekusi kode arbitrer, penelitian ini dilakukan untuk membatasi dampak dari kondisi tersebut. Penelitian ini merancang mekanisme *hardening* pada Ubuntu Server 24.04 LTS menggunakan AppArmor dan Fail2Ban. AppArmor berperan sebagai *Mandatory Access Control* (MAC) yang membatasi ruang lingkup suatu aplikasi terhadap sistem, dalam penelitian ini berfungsi mengurung (*contain*) proses

yang telah dikompromikan agar tidak dapat mengakses sumber daya di Ubuntu Server secara sembarangan (Alviano dan Sestito, 2025). Sementara itu, Fail2Ban berfungsi memantau *file log* aktivitas suatu layanan dan secara otomatis memblokir alamat IP yang terindikasi melakukan serangan (Ridho et al., 2025).

Pada penelitian ini, AppArmor bertindak sebagai aktor utama yang mencegah upaya eksploitasi *buffer overflow* dan *format string* terhadap akses sumber daya sistem yang tidak diizinkan. Di sisi lain, Fail2Ban bertugas untuk memblokir alamat IP penyerang berdasarkan pola eksploitasi *binary* yang menyebabkan *crash*, seperti spam input karakter hingga melebihi batas input yang telah dialokasikan atau kesalahan dalam skrip eksploitasi. Selain itu, Fail2Ban dapat digunakan untuk memblokir upaya pembanjiran koneksi (*TCP connection flooding*) setelah ambang batas tercapai. Dengan demikian, dampak eksploitasi yang dilakukan oleh penyerang ke Ubuntu Server dapat dibatasi di lapisan sistem operasi dan jaringan.

1.2 Perumusan Masalah

Berdasarkan latar belakang masalah, maka diperoleh rumusan masalah pada penelitian ini sebagai berikut:

1. Bagaimana merancang sistem keamanan pada Ubuntu Server menggunakan Fail2Ban dan AppArmor terhadap eksploitasi *buffer overflow* dan *format sstring*?
2. Bagaimana efektivitas Fail2Ban dan AppArmor dalam membatasi dampak eksploitasi *buffer overflow* dan *format string* di Ubuntu Server?

1.3 Batasan Masalah

Berikut adalah beberapa batasan masalah pada penelitian ini agar cakupan penelitian tidak terlalu luas:

1. Celah keamanan yang diuji terbatas pada *buffer overflow* dan *format string*.
2. Alat dan modul keamanan yang dipakai terbatas pada Fail2Ban dan AppArmor.
3. Lingkungan pengujian yang digunakan terbatas pada Ubuntu Server versi 24.04.

4. Objek penelitian ini menggunakan 10 *file ELF binary* yang dirancang memiliki kerentanan dan memiliki variasi alur logika program dan proteksi yang berbeda-beda untuk pengujian celah keamanan, Terdapat 5 *file ELF binary* yang memiliki kerentanan *buffer overflow* dan 5 *file ELF binary* yang memiliki kerentanan *format string*.
5. Mekanisme keamanan tidak mencakup *patching* terhadap file *ELF binary* yang rentan, melainkan berfokus pada proteksi di lapisan sistem operasi dan jaringan pada Ubuntu Server.

1.4 Tujuan dan Manfaat

Sejalan dengan rumusan masalah, berikut adalah beberapa tujuan yang akan dicapai pada penelitian ini:

1. Merancang sistem keamanan pada Ubuntu Server menggunakan Fail2Ban dan AppArmor guna mencegah eksploitasi celah *buffer overflow* dan *format string*.
2. Mengevaluasi efektivitas Fail2Ban dan AppArmor dalam membatasi dampak eksploitasi *buffer overflow* dan *format string* di Ubuntu Server.

Adapun, manfaat yang akan diperoleh dari penelitian ini antara lain:

1. Bagi peneliti keamanan (*security researcher*)

   Penelitian ini diharapkan dapat menambah referensi mengenai rancangan sistem keamanan dalam meminimalisir upaya eksploitasi kerentanan memori melalui celah *buffer overflow* dan *format string*.

2. Bagi pengembang (*application developer*)

   Penelitian ini diharapkan dapat membantu tim IT dalam menyusun strategi mitigasi terhadap aplikasi yang memiliki *memory corruption bugs*, terutama ketika proses *patching* aplikasi belum dapat dilakukan secara langsung. Selain itu, alat dan modul keamanan dapat digunakan secara gratis, sehingga akan menghemat biaya pengeluaran dalam perancangan sistem keamanan.

1.5 Sistematika Penulisan

Dalam melakukan penelitian ini, berikut adalah sistematika penulisan yang digunakan:

A. BAB I PENDAHULUAN

Bab ini memuat penjelasan mengenai penelitian apa dan mengapa penelitian tersebut harus dilakukan. Bagian ini terdiri dari latar belakang masalah, perumusan masalah, tujuan dan manfaat penelitian, dan batasan masalah terkait penelitian ini.

B. BAB II TINJAUAN PUSTAKA

Bab ini memuat rangkuman penelitian yang berhubungan dengan penelitian ini untuk memperjelas perbedaan dan memperoleh aspek keterbaruan. Kajian literatur yang relevan dengan topik penelitian ini juga ditambahkan untuk memperkuat landasan teori.

C. BAB III METODOLOGI PENELITIAN

Bab ini memuat penjabaran terkait bagaimana penelitian ini akan dilakukan. Metodologi pada penelitian ini meliputi studi literatur, perancangan sistem keamanan, pengujian sistem keamanan, evaluasi hasil pengujian, dan dokumentasi penelitian.

D. BAB IV HASIL DAN PEMBAHASAN

Bab ini memuat hasil dan pembahasan pada penelitian yang telah dilakukan. Parameter pengujian dapat didapatkan pada bab ini setelah metodologi penelitian dilakukan dengan benar dan berurutan.

E. BAB V PENUTUP

Bab ini memuat kesimpulan dan saran pada penelitian yang telah dilakukan. Selain itu, penjelasan pada bagian saran akan ditambahkan apabila masih ada kekurangan yang dapat dikembangkan lebih lanjut.

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan perancangan dan pengujian sistem keamanan pada Ubuntu Server terhadap eksploitasi celah *buffer overflow* dan *format string*, dapat diambil kesimpulan sebagai berikut:

1. AppArmor efektif dalam membatasi dampak lanjutan dari eksploitasi yang telah berhasil, bukan menambal kerentanan *binary* itu sendiri. Hal ini dibuktikan secara kontras antara profil AppArmor longgar dan profil ketat pada VM-2 dan VM-3. Di mana, profil longgar masih mengizinkan penyerang memperoleh *shell* dan membaca *file-file* sistem sensitif. Sedangkan, pada profil ketat, setiap upaya eskalasi seperti eksekusi perintah yang tidak diizinkan, seperti *"whoami"*, *"hostname"*, *"ls"*, dan *"xxd"*, serta pembacaan file-file sensitif berhasil ditolak dan tercatat sebagai peristiwa pelanggaran *(deny-event)*. Dengan demikian, klaim efektivitas AppArmor bersifat berkualifikasi, yaitu efektif sebagai lapisan pencegah di tingkat sistem operasi (Ubuntu Server).
2. Fail2Ban efektif memblokir alamat IP penyerang berdasarkan bukti *crash* berulang di tingkat jaringan. Setiap alamat IP yang melewati ambang batas *(maxretry)*, yaitu 3 alamat IP yang melakukan *crash* berulang serta alamat IP lain yang melakukan pembanjiran IP berhasil diblokir oleh Fail2Ban. Alamat IP yang hanya menimbulkan *crash* di bawah ambang tidak diblokir, didesain menggunakan 3 kali toleransi pelanggaran agar tidak terjadi *false-positive*. Pemblokiran didasarkan pada bukti *crash* yang terjadi akibat input melebihi batas maksimum karakter yang telah dialokasikan atau kesalahan dalam proses eksploitasi. Sinyal *crash* dari hasil pengujian ada beragam, yang mencakup *error Segmentation Fault* (menghasilkan sinyal nomor 11) dan *Stack Smashing Detected* (menghasilkan sinyal nomor 6) yang memicu pemblokiran. Selain itu, terdapat sinyal nomor 14 *(SIGALRM*) yang secara otomatis menginterupsi proses jika tidak ada interaksi selama jangka waktu tertentu (kondisi *idle*).

3. Kombinasi AppArmor dan Fail2Ban membentuk arsitektur pertahanan berlapis yang saling melengkapi sesuai peran masing-masing. Pada pengujian pembanjiran koneksi (*TCP connection flooding*), Fail2Ban memblokir alamat IP pembanjir dalam orde detik, sehingga beban server tetap rendah, yaitu CPU pada puncak 1,32% dan lalu lintas jaringan (*Rx eth0*) pada puncak 9,67 kb/s. Di sisi lain, penggunaan memori tidak terpengaruh secara signifikan. Batasan sistem yang harus diakui secara eksplisit adalah bahwa eksploitasi yang berhasil tanpa menimbulkan *crash* tidak terdeteksi oleh Fail2Ban, dan pada kondisi tersebut pertahanan sepenuhnya bertumpu pada AppArmor sebagai lapisan pencegah.

5.2 Saran

Rancangan sistem keamanan yang telah terbukti efektif masih dapat dikembangkan dalam aspek lainnya. Ada beberapa saran yang dapat dilakukan untuk penelitian selanjutnya, di antaranya:

1. Notifikasi pemblokiran saat ini hanya tercatat pada *log* dan memerlukan pemeriksaan manual. Disarankan menambahkan mekanisme notifikasi real-time, seperti bot Telegram agar tim *Security Operation Center* (SOC) memperoleh peringatan langsung saat terjadi pemblokiran, sehingga respons terhadap serangan dapat dilakukan lebih cepat.
2. Perlu ditambahkan mekanisme deteksi terhadap eksploitasi yang berhasil tanpa menimbulkan *crash*. Misalnya, melalui pemantauan perilaku proses (*runtime behavior monitoring*) atau *audit log* pada level *syscall*, karena eksploitasi jenis ini merupakan celah yang lolos dari deteksi Fail2Ban.
3. Cakupan celah keamanan dalam penelitian ini terbatas pada *buffer overflow* dan *format string*. Misalnya, *heap overflow*, *use-after-free* (UAF), atau *double free*. Tujuannya untuk menilai generalisasi efektivitas pendekatan *containment* dan *enforcement* berlapis ini.
4. Penerapan *secure coding* pada tahap pengembangan *binary* perlu diintegrasikan sebagai lapisan pertahanan tambahan, karena AppArmor dan Fail2Ban membatasi dampak dan memblokir serangan, tetapi tidak menghilangkan akar kerentanan pada kode itu sendiri.

DAFTAR PUSTAKA

Adiyoso, W., & Susetyo, Y. A. (2023). PEMBANGUNAN COMPILER DOMAIN SPECIFIC LANGUAGE SEBAGAI GENERATOR FORM HTML MENGGUNAKAN PYTHON SLY. *Jurnal Indonesia : Manajemen Informatika dan Komunikasi*, *4*(2), 449–461. https://doi.org/10.35870/jimik.v4i2.231

Alviano, M., & Sestito, P. (2025). User Armor: An Extension for AppArmor. *Algorithms*, *18*(4), 185. https://doi.org/10.3390/a18040185

Amrullah, H. I., Amarta, A. N. F., & Rilvani, E. (2025). Fleksibilitas dan Kesederhanaan Arsitektur Sistem Operasi Linux. *Neptunus: Jurnal Ilmu Komputer Dan Teknologi Informasi*, *3*(1), 59–64. https://doi.org/10.61132/neptunus.v3i1.630

Anderson, S. N., Blanco, R., Lampropoulos, L., Pierce, B. C., & Tolmach, A. (2023). Formalizing Stack Safety as a Security Property. *2023 IEEE 36th Computer Security Foundations Symposium (CSF)*, 356–371. https://doi.org/10.1109/CSF57540.2023.00037

Ansar, M. I. (2025a). *Apa itu NGINX? Fungsi, Manfaat, dan Kelebihannya*. https://digitalsolusigrup.co.id/nginx-adalah/

Ansar, M. I. (2025b, September 10). *Apa Itu Grafana? Fungsi, Fitur, dan Manfaat Penggunaannya*. https://digitalsolusigrup.co.id/grafana-adalah/

Balqis, P., & Rahman, R. (2025). Analisis Keamanan Layanan SSH terhadap Brute Force Attack. *Merkurius : Jurnal Riset Sistem Informasi dan Teknik Informatika*, *3*(4), 240–246. https://doi.org/10.61132/merkurius.v3i4.949

Bosman, E., & Bos, H. (2014). Framing Signals—A Return to Portable Shellcode. *2014 IEEE Symposium on Security and Privacy*, 243–258. https://doi.org/10.1109/SP.2014.23

Boyanov, P. (2023). BASIC NETWORK PENETRATION TESTING WITH THE NETWORK TOOL NETCAT IN LINUX-BASED OPERATING

SYSTEMS. *Journal Scientific and Applied Research*, *25*(1), 15–30. https://doi.org/10.46687/jsar.v25i1.377

Butt, M. A., Ajmal, Z., Khan, Z. I., Idrees, M., & Javed, Y. (2022). An In-Depth Survey of Bypassing Buffer Overflow Mitigation Techniques. *Applied Sciences*, *12*(13), 6702. https://doi.org/10.3390/app12136702

Canella, C., Werner, M., Gruss, D., & Schwarz, M. (2021). Automating Seccomp Filter Generation for Linux Applications. *Proceedings of the 2021 on Cloud Computing Security Workshop*, 139–151. https://doi.org/10.1145/3474123.3486762

ÇeliK, E., & Dal, D. (2022). Comparison of the Sizes of Assembly Codes Generated by the GCC Compiler for Different CPU Architectures. *International Conference on Informatics and Computer Science*, 8–13. https://www.researchgate.net/publication/377400794_Comparison_of_the_Sizes_of_Assembly_Codes_Generated_by_the_GCC_Compiler_for_Different_CPU_Architectures

Chen, Z., Pan, B., & Sun, Y. (2019). A Survey of Software Reverse Engineering Applications. *In: Sun, X., Pan, Z., Bertino, E. (Eds) Artificial Intelligence and Security*, *11635*, 235–245. https://doi.org/10.1007/978-3-030-24268-8_22

Creswell, J. W., & Creswell, J. D. (2022). *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches* (6th ed.). SAGE Publishing. https://www.sagepub.com/shop/buy-a-book/research-design-6-270550

Cybersecurity and Infrastructure Security Agency & Federal Bureau of Investigation. (2025, February 12). Secure by Design Alert: Eliminating Buffer Overflow Vulnerabilities. *FACT SHEET*. https://www.cisa.gov/resources-tools/resources/secure-design-alert-eliminating-buffer-overflow-vulnerabilities

Dawamsyach, F., Ruslianto, I., & Ristian, U. (2023). Implementasi IPS (Intrusion Prevention System) Fail2ban Pada Server Terhadap Serangan DDoS dan

Brute Force. *CESS (Journal of Computer Engineering, System and Science)*, *8*(1), 149–161.

Degioanni, L. (2018, June 29). *3 phases of Prometheus adoption.* https://www.sysdig.com/blog/3-phases-of-prometheus-adoption

Dickinson, S. (2019, April 26). An introduction to AppArmor. *Ubuntu Blog*. https://ubuntu.com/blog/an-introduction-to-apparmor

Elradi, M. D. (2025). Prometheus & Grafana: A Metrics-focused Monitoring Stack. *Journal of Computer Allied Intelligence*, *3*(3), 28–39. https://doi.org/10.69996/jcai.2025015

El-Zoghby, A. M., ElSayed, M. S., Jurcut, A., & Azer, M. A. (2025). Overview of the Code-Reuse Attacks Mitigations, and Evaluation using SMAA-2 Approach. *Advances in Knowledge-Based Systems, Data Science, and Cybersecurity*, 108–148.

Erawan, E., & Salman, M. (2023). Penguatan Keamanan Otomatis pada Sistem Operasi Ubuntu berbasis Image Mesin Virtual menggunakan solusi Packer. *Cakrawala Repositori IMWI*, *6*(4), 1089–1097. https://doi.org/10.52851/cakrawala.v6i4.451

Farhannullah, & Hardjianto, M. (2022). Sistem Monitoring Serangan SSH dengan Metode Intrusion Prevention System (IPS) Fail2ban Menggunakan Python Pada Sistem Operasi Linux. *Jurnal Ticom: Technology of Information and Communication*, *11*(1), 33–38. https://doi.org/10.70309/ticom.v11i1.68

Full Stack Python. (n.d.-a). *Terminal Multiplexer*. Terminal Multiplexer. Retrieved June 25, 2026, from https://www.fullstackpython.com/tmux.html

Full Stack Python. (n.d.-b). *Why Use Python?* Full Stack Python. Retrieved June 25, 2026, from https://www.fullstackpython.com/why-use-python.html

Gaidis, A. J., Moreira, J., Sun, K., Milburn, A., Atlidakis, V., & Kemerlis, V. P. (2023). FineIBT: Fine-grain Control-flow Enforcement with Indirect Branch Tracking. *Proceedings of the 26th International Symposium on*

Research in Attacks, Intrusions and Defenses, 527–546. https://doi.org/10.1145/3607199.3607219

GEF Documentation. (n.d.). *GEF - GDB Enhanced Features*. GEF - GDB Enhanced Features. Retrieved June 25, 2026, from https://hugsy.github.io/gef/

George, G., Kotey, J., Ripley, M., Sultana, K. Z., & Codabux, Z. (2021). A Preliminary Study on Common Programming Mistakes that Lead to Buffer Overflow Vulnerability. *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*, 1375–1380. https://doi.org/10.1109/COMPSAC51774.2021.00194

Gherghe, A. L., & Tudose, C. (2025). *A Novel Framework for Evaluating Application Performance in Distributed Systems*. Computer Science and Mathematics. https://doi.org/10.20944/preprints202511.0602.v1

HackerArise. (2023, November 26). Reverse Engineering Malware, Part 03: IDA Pro Introduction [Malware Analysis]. *HackerArise*. https://hackers-arise.com/reverse-engineering-malware-part-3-ida-pro-introduction/

Hadi, M., Azizah, N., & Jaya, F. (2025). Sistem Informasi Perpustakaan Berbasis Website Menggunakan Metode Extreme Programming Di SDI Nurul Manshur. *Jurnal Teknik Mesin, Elektro dan Ilmu Komputer*, *5*(3), 113–132. https://doi.org/10.55606/teknik.v5i3.7776

Hanaputra, R. R., Andzani, N. S., Bintara, M. A., & Rosadi, A. (2023). Implementasi Penggunaan Pwntools Guna Mempermudah Eksploitasi Buffer pada Ubuntu 32-Bit. *Journal on Education*, *5*(3), 5700–5706.

Imtiaz, N., & Williams, L. (2021). *Memory Error Detection in Security Testing* (arXiv:2104.04385). arXiv. https://doi.org/10.48550/arXiv.2104.04385

Kali Linux Documentation. (n.d.-a). *Netcat*. Netcat. Retrieved June 25, 2026, from https://www.kali.org/tools/netcat/

Kali Linux Documentation. (n.d.-b). *Socat*. Socat. Retrieved June 25, 2026, from https://www.kali.org/tools/socat/

Karim, M. M. (2015). *Source Code based Buffer Overflow Detection Technology* [Northwestern Polytechnical University]. https://doi.org/10.13140/RG.2.1.3837.9124

Kim, K., Kim, J., & Lee, S. (2024). StackGuard+: Interoperable alternative to canary-based protection of stack smashing. *Electronics Letters*, *60*(19), e13310. https://doi.org/10.1049/ell2.13310

Kyi, W., Koide, H., & Sakurai, K. (2019). Analyzing the Effect of Moving Target Defense for a Web System. *International Journal of Networking and Computing*, *9*(2), 188–200. https://doi.org/10.15803/ijnc.9.2_188

Lam, R. (2024, April 10). *Introducing an OpenTelemetry Collector distribution with built-in Prometheus pipelines: Grafana Alloy*. https://grafana.com/blog/Grafana Alloy-opentelemetry-collector-with-prometheus-pipelines/

Liu, C., Wu, Y.-J., Wu, J.-Z., & Zhao, C. (2023). A buffer overflow detection and defense method based on RISC-V instruction set extension. *Cybersecurity*, *6*(1), 45. https://doi.org/10.1186/s42400-023-00164-x

Lu, H. J., Matz, M., Girkar, M., Hubicka, J., Jaeger, A., & Mitchell, M. (2021, September 28). *System V Application Binary Interface AMD64 Architecture Processor Supplement (With LP64 and ILP32 Programming Models) Version 1.0* [Technical Specification]. https://cs61.seas.harvard.edu/site/2025/pdf/x86-64-abi-20210928.pdf

Machado, A. (2024a, October 28). *An In-Depth Exploration of GCC: History, Variants, and Internal Architecture* [Technology]. https://machaddr.substack.com/p/an-in-depth-exploration-of-gcc-history

Machado, A. (2024b, November 5). *The Evolution of the GNU C Library: History, Key Contributors, and Impact on the Linux Ecosystem* [Technology]. https://machaddr.substack.com/p/the-evolution-of-the-gnu-c-library

Malabay. (2016). PEMANFAATAN FLOWCHART UNTUK KEBUTUHAN DESKRIPSI PROSES BISNIS. *JURNAL ILMU KOMPUTER (JIK)*, *12*(1), 21–26. https://doi.org/10.47007/komp.v12i1.1579

Marco-Gisbert, H., & Ripoll, I. R. (2019). Address Space Layout Randomization Next Generation. *Applied Sciences*, *9*(14), 2928. https://doi.org/10.3390/app9142928

Maulana, M. H., Ismail, S. J. I., & Rizal, M. F. (2023). Membangun Server CTF (Capture the Flag) Untuk Komunitas Security Di Telkom University. *e-Proceeding of Applied Science*, *9*(5), 2332–2338.

Miller. (2019, June 16). A proactive approach to more secure code. *Security Research and Defense*. https://www.microsoft.com/en-us/msrc/blog/2019/07/a-proactive-approach-to-more-secure-code

MITRE. (2025, December 15). *2025 CWE Top 25 Most Dangerous Software Weaknesses* [Technology]. MITRE Organization. Common Weakness Enumeration. https://cwe.mitre.org/top25/archive/2025/2025_cwe_top25.html#top25list

National Security Agency USA. (2022, November 10). Software Memory Safety. *Cybersecurity Information Sheet*. https://www.nsa.gov/Press-Room/News-Highlights/Article/Article/3215760/nsa-releases-guidance-on-how-to-protect-against-software-memory-safety-issues/

OffSec Tools. (n.d.). *Pwntools*. Retrieved June 25, 2026, from https://offsec.tools/tool/pwntools

Panter, S. K., & Eisty, N. U. (2024). Rusty Linux: Advances in Rust for Linux Kernel Development. *Proceedings of the 18th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, 496–502. https://doi.org/10.1145/3674805.3690756

Park, J., Kim, J., B. Gupta, B., & Park, N. (2021). Network Log-Based SSH Brute-Force Attack Detection Model. *Computers, Materials & Continua*, *68*(1), 887–901. https://doi.org/10.32604/cmc.2021.015172

Peppas, N. (n.d.). How to Install and Configure Fail2ban on Ubuntu Server 16.04 [Technology]. *Liquid Web by Nexcess*. Retrieved June 25, 2026, from https://www.liquidweb.com/blog/install-configure-fail2ban-ubuntu-server-16-04/

Pereira, P. R., & Sanches, J. (2025). Network and Systems Monitoring with Prometheus and Grafana. *Proceedings of 20th Iberian Conference on Information Systems and Technologies (CISTI 2025) - Volume 1*, *1716 LNNS*, 367–378. https://doi.org/10.1007/978-3-032-10929-3_32

Petrean, F., & Coleşa, A. (2024). PwnMaster: Automatic Buffer Overflow and Format String Vulnerability Detection and Exploitation. *2024 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, 1–5. https://doi.org/10.1109/AQTR61889.2024.10554220

Rahman, R., Nurfaida, & Rusmin, M. H. (2026). ANALISIS KEAMANAN FILE SERVER BERBASIS LINUX TERHADAP AKSES TIDAK SAH. *Jurnal Riset Sistem Informasi*, *3*(2), 58–62. https://doi.org/10.69714/38pr8578

Rasyid, A. H., & Santoso, J. D. (2026). Identifikasi Kerentanan Perangkat Lunak Pengolah Dokumen Berbasis Binary menggunakan Metode Whitebox Fuzzing Afl++. *Jurnal Multidisiplin Bhatara*, *3*(1). https://doi.org/https://doi.org/10.59095/jmb.v3i1.265

Ridho, M., Hafizh, A., Dani, I., & Ariyadi, T. (2025). Peningkatan Keamanan SSH Server Berbasis Linux melalui Implementasi Fail2Ban dan Uji Serangan Brute Force. *JURNAL PENELITIAN MULTIDISIPLIN BANGSA*, *1*(12), 2206–2214. https://doi.org/10.59837/jpnmb.v1i12.431

Rieger, G. (n.d.). Socat Man Pages. In *Linux manual pages*. Retrieved https://man7.org/linux/man-pages/man1/socat.1.html

Rohleder, R. (2019). Hands-On Ghidra—A Tutorial about the Software Reverse Engineering Framework. *Proceedings of the 3rd ACM Workshop on Software Protection*, 77–78. https://doi.org/10.1145/3338503.3357725

Sahara, A. D., Sapri, S., & Akbar, A. A. (2024). The Design And Implementation Of Computer Network Monitoring And Security System Using Linux Ubuntu Server. *Jurnal Media Computer Science*, *3*(1), 1–16. https://doi.org/10.37676/jmcs.v3i1.5328

Seyedhamed, G., Tapti, P., Shachee, M., & Michalis, P. (2020). Temporal System Call Specialization for Attack Surface Reduction. *29th USENIX Security Symposium*, 1749–1766. https://www.usenix.org/system/files/sec20-ghavamnia.pdf

Shao, M., Chen, B., Jancheska, S., Dolan-Gavitt, B., Garg, S., Karri, R., & Shafique, M. (2024). *An Empirical Evaluation of LLMs for Solving Offensive Security Challenges* (arXiv:2402.11814). arXiv. https://doi.org/10.48550/arXiv.2402.11814

Sharma, I. (2026, February 26). What is C Programming? Introduction, Features, and Applications [PROGRAMMING LANGUAGES]. *C Programming Introduction*. https://www.guvi.in/blog/what-is-c-programming/

Sublime Text. (n.d.). *Text Editing, Done Right*. Sublime Text. Retrieved https://www.sublimetext.com/

Ubuntu Handbook. (2022, April 21). Ubuntu 22.04 LTS Available to Download, See What's New! [News & Tutorial]. *Ubuntu Handbook*. https://ubuntuhandbook.org/index.php/2022/04/ubuntu-22-04-lts-available/

VM, S. (2024, February 12). *Log Monitoring | Grafana Loki Installation*. https://www.skynats.com/blog/log-monitoring-Grafana Loki-installation/

Xie, S., Zhang, C., Zhou, T., Liu, J., Hong, X., Li, Q., & Peng, X. (2026). *LogCopilot: Automating Log Aggregation Analysis through Large Language Models* (arXiv:2606.17094). arXiv. https://doi.org/10.48550/arXiv.2606.17094

Xie, T., Zhang, Y., Li, J., Liu, H., & Gu, D. (2016). New Exploit Methods against Ptmalloc of GLIBC. *2016 IEEE Trustcom/BigDataSE/ISPA*, 646–653. https://doi.org/10.1109/TrustCom.2016.0121

Xu, S., Jiang, Z., Wang, Y., & Xie, P. (2024). FormatAEG: A framework for bypassing ASLR defense and automated exploitation of format string vulnerability. *The Computer Journal*, *67*(11), 3056–3066. https://doi.org/10.1093/comjnl/bxae069

Xu, S., & Wang, Y. (2022). BofAEG: Automated Stack Buffer Overflow Vulnerability Detection and Exploit Generation Based on Symbolic Execution and Dynamic Analysis. *Security and Communication Networks*, *2022*, 1–9. https://doi.org/10.1155/2022/1251987

Yanto, R. (2024). Implementasi Ubuntu Web Server dengan Service Nginx pada STMIK Dharmapala Riau. *Journal of Enterpreneurship & Technopreneur (JoET)*, *1*(1), 26–38.

Yoon, H., & Lee, M. (2022). SGXDump: A Repeatable Code-Reuse Attack for Extracting SGX Enclave Memory. *Applied Sciences*, *12*(15), 7655. https://doi.org/10.3390/app12157655

Zhang, H., Wang, J., Wang, Y., Li, M., Song, J., & Liu, Z. (2023). ICVTest: A Practical Black-Box Penetration Testing Framework for Evaluating Cybersecurity of Intelligent Connected Vehicles. *Applied Sciences*, *14*(1), 204. https://doi.org/10.3390/app14010204

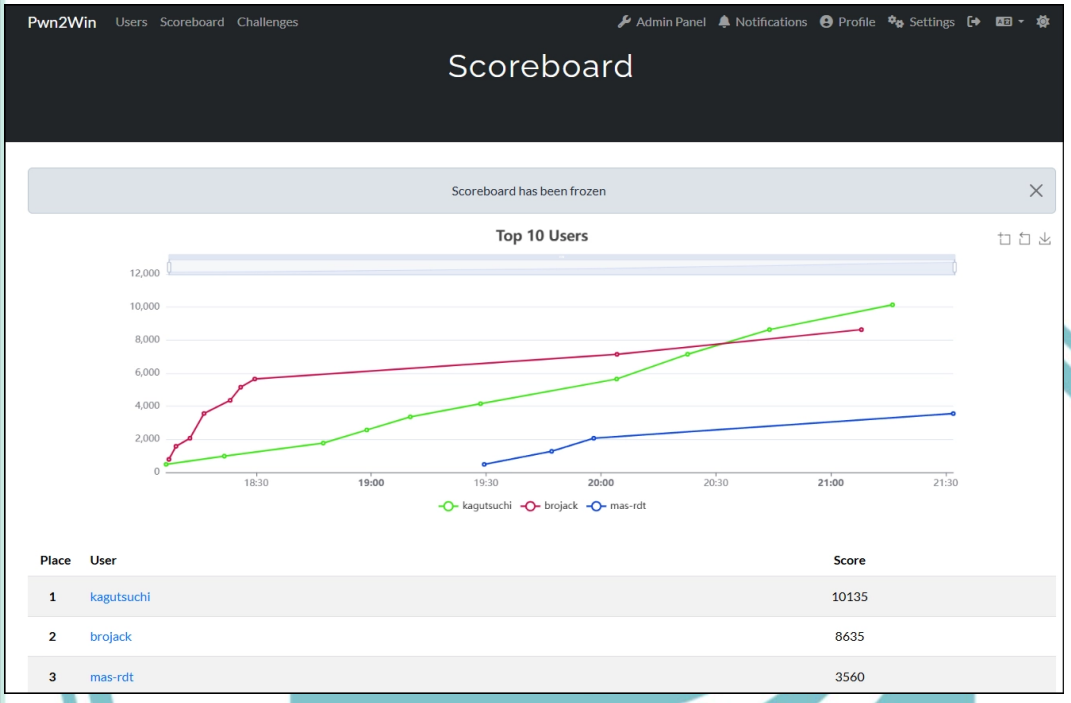
DAFTAR RIWAYAT HIDUP PENULIS

Wahyu Priambodo

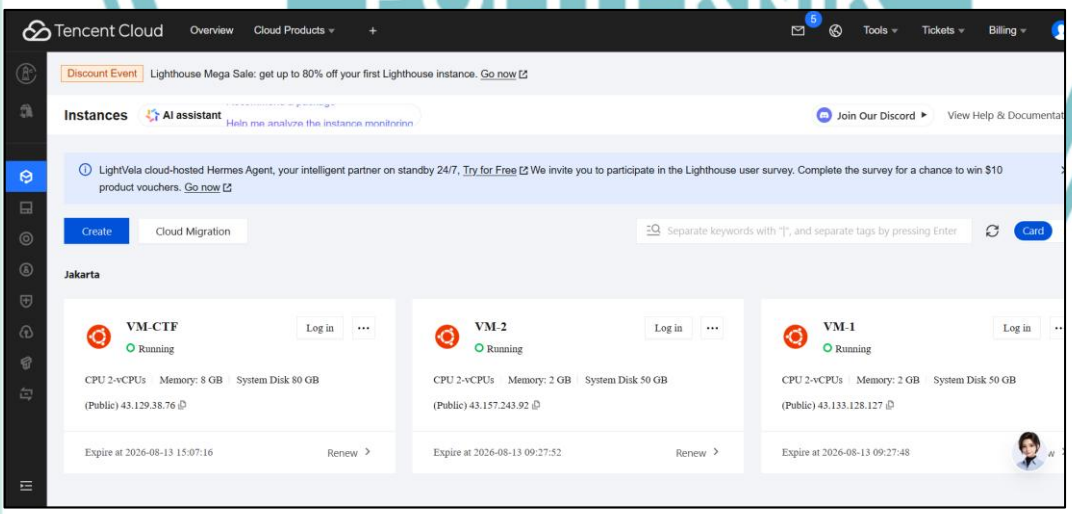
Lahir pada tahun 2003 dan merupakan anak ke-2 dari 2 bersaudara. Penulis menyelesaikan pendidikan dasar di SD Negeri 01 Bukit Duri, pendidikan menengah pertama di SMP Negeri 26 Jakarta, dan pendidikan menengah atas di SMK Negeri 1 Jakarta dengan jurusan Sistem Informatika, Jaringan, dan Aplikasi (SIJA) selama 4 tahun. Pada tahun 2022, penulis melanjutkan pendidikan di Politeknik Negeri Jakarta (PNJ), Jurusan Teknik Informatika dan Komputer dan Program Studi Teknik Multimedia dan Jaringan. Penulis memiliki ketertarikan untuk mendalami ilmu komputer, khususnya di bidang jaringan komputer dan keamanan siber.

LAMPIRAN

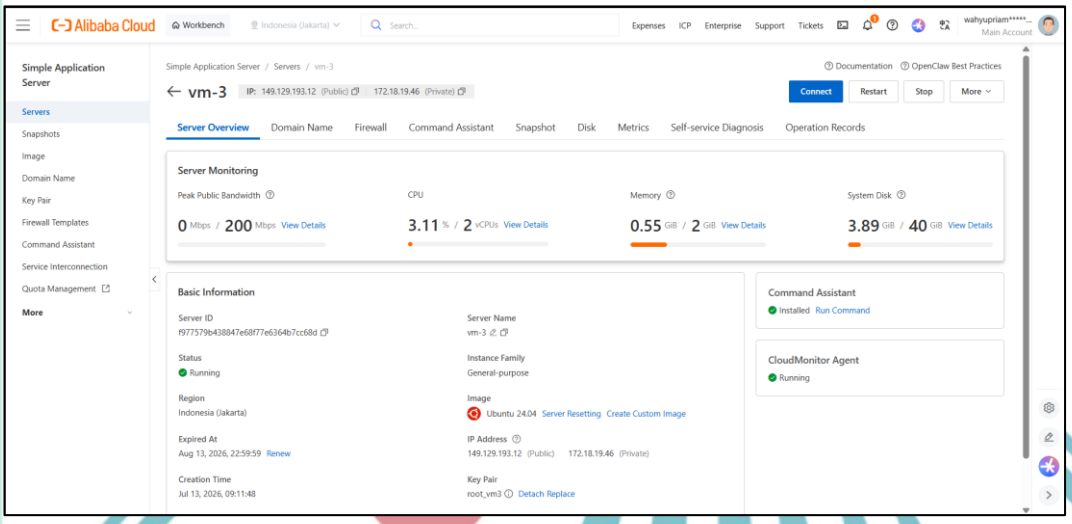
Lampiran 1. Bukti Pelaksanaan Ajang CTF Internal untuk Pengambilan Data



Lampiran 2. Bukti VM-CTF, VM-1, dan VM-2 yang Disewa dari *Provider – Tencent Cloud*



Lampiran 3. Bukti VM-3 yang Disewa dari *Provider – Alibaba Cloud*



Lampiran 4. Bukti Grafana Berhasil Memvisualisasikan Sumber Daya Server Lain Secara *Real-time*

Lampiran 5. Akses Platform CTF Melalui Internet

Ajang CTF yang telah dilaksanakan pada tanggal 15 Juli 2026 dari mulai pukul 17:00 WIB hingga 23:59 WIB diikuti oleh 3 peserta yang beraksi selayaknya penyerang. Di mana, CTF diselenggarakan menggunakan platform CTFd yang tersedia secara gratis dan menyediakan 10 *file binary* (objek penelitian) kepada penyerang untuk dieksploitasi. Adapun platform CTF dapat diakses melalui server utama yang sudah dikaitkan dengan nama domain dan koneksi HTTPS. Akses platform CTF melalui nama *domain* berikut: https://ctf.pwngame.site

Lampiran 4. Repositori Lab Penelitian (GitHub)

File-file objek penelitian, skrip eksploitasi, *service*, dan konfigurasi terhadap komponen-komponen yang diperlukan sudah penulis sertakan pada repositori GitHub. Repositori GitHub penulis untuk lab penelitian ini tertera pada tautan berikut: https://github.com/whyuhurtz/research-lab/