RANCANG BANGUN *MAIL SERVER* DAN IMPLEMENTASI *HONEYPOT* SEBAGAI KEAMANAN JARINGAN PADA *SERVER* BERBASIS LINUX

SKRIPSI

Dibuat Untuk Melengkapi Syarat-Syarat yang Diperlukan

untuk Memperoleh Diploma Empat Politeknik

ALDY WIDANTO

1807422021

PROGRAM STUDI TEKNIK MULTIMEDIA DAN JARINGAN

JURUSAN TEKNIK INFORMATIKA DAN KOMPUTER

POLITEKNIK NEGERI JAKARTA

2025

SURAT PERNYATAAN BEBAS PLAGIARISME

Saya yang bertanda tangan di bawah ini:

Nama	:	Aldy Widanto
NIM	:	1807422021
Jurusan/Program Studi	:	Teknik Informatika dan Komputer/Teknik Multimedia dan Jaringan
Judul Skripsi	:	RANCANG BANGUN *MAIL SERVER* DAN IMPLEMENTASI *HONEYPOT* SEBAGAI KEAMANAN JARINGAN PADA SERVER BERBASIS LINUX

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, 30 Juli 2025

Yang membuat pernyataan

(Aldy Widanto)

NIM. 1807422021

LEMBAR PENGESAHAN

Skripsi diajukan oleh :

Nama	:	Aldy Widanto
NIM	:	1807422021
Program Studi	:	Teknik Multimedia dan Jaringan
Judul Skripsi	:	RANCANG BANGUN MAIL SERVER DAN IMPLEMENTASI *HONEYPOT* SEBAGAI KEAMANAN JARINGAN PADA SERVER BERBASIS LINUX

Telah diuji oleh tim penguji dalam Sidang Skripsi pada hari Selasa,
Tanggal 15, Bulan Juli, Tahun 2025 dan dinyatakan
LULUS/~~TIDAK LULUS.~~

Disahkan oleh

Pembimbing I : Iik Muhamad Malik Matin, S.Kom., M.T. ( )

Penguji I : Defiana Arnaldy, S.Tp., M.Si. ( )

Penguji II : Dr. Indra Hermawan, S.Kom., M.Kom. ( )

Penguji III : Asep Kurniawan, S.Pd., M.Kom. ( )

Mengetahui:
Jurusan Teknik Informatika dan Komputer
Ketua

Dr. Anita Hidayati, S.Kom., M.Kom.
NIP. 197908032003122003

KATA PENGANTAR

Puji syukur penulis panjatkan ke hadirat Tuhan Yang Maha Esa atas segala limpahan rahmat, karunia, dan kemudahan yang telah diberikan, sehingga penulisan skripsi ini dapat diselesaikan dengan baik. Laporan skripsi ini disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana Terapan (Diploma 4) di Politeknik Negeri Jakarta.

Dalam kesempatan ini, penulis ingin menyampaikan penghargaan dan terima kasih yang sebesar-besarnya kepada berbagai pihak yang telah memberikan dukungan, bimbingan, serta motivasi selama proses penyusunan skripsi ini, di antaranya:

1. Kepada Bapak **Iik Muhamad Malik Matin, S.Kom., M.T.** selaku dosen pembimbing yang telah memberikan arahan, bimbingan, serta saran-saran yang sangat berarti dalam setiap tahapan penulisan skripsi ini.
2. Kepada Bapak **Ronald E. Samosir** dan Ibu **Sri Wiyatmi** selaku orang tua tercinta yang telah memberikan kepercayaan penuh, dukungan moral, dan fasilitas yang diperlukan selama studi dan penyusunan skripsi ini.
3. Kepada **Jonathan Elloy Simanjuntak S.T.** dan **Gratia Brenda Gerung S.Si.** yang telah memberikan dukungan serta bantuan, baik secara moril maupun materil, sehingga penulis dapat terus bersemangat dalam menyelesaikan tugas akhir ini.

Penulis menyadari bahwa skripsi ini masih jauh dari sempurna, baik dari segi isi maupun penyajiannya. Oleh karena itu, kritik dan saran yang membangun sangat penulis harapkan demi perbaikan di masa mendatang.

Akhir kata, penulis berharap semoga skripsi ini dapat memberikan manfaat dan menjadi kontribusi yang berarti bagi pengembangan ilmu pengetahuan, khususnya di bidang yang relevan.

SURAT PERNYATAAN PERSETUJUAN PUBLIKASI SKRIPSI UNTUK KEPENTINGAN AKADEMIS

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

Nama	:	Aldy Widanto
NIM	:	1807422021
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 :

RANCANG BANGUN *MAIL SERVER* DAN IMPLEMENTASI *HONEYPOT* SEBAGAI KEAMANAN JARINGAN PADA SERVER BERBASIS LINUX

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 pemilik Hak Cipta.

Demikian pernyatan ini saya buat dengan sebenarnya.

Depok, 30 Juli 2025

Yang Menyatakan

(Aldy Widanto)

NIM. 1807422021

RANCANG BANGUN *MAIL SERVER* DAN IMPLEMENTASI *HONEYPOT* SEBAGAI KEAMANAN JARINGAN PADA SERVER BERBASIS LINUX

Abstrak

Mail server merupakan salah satu komponen vital dalam komunikasi digital, namun juga rentan terhadap berbagai ancaman siber, terutama serangan *bruteforce* pada protokol SSH. Penelitian ini bertujuan merancang dan mengimplementasikan sistem keamanan jaringan berbasis *honeypot* Cowrie pada *mail server* berbasis iRedMail dengan sistem operasi Debian 12. Metode yang digunakan adalah pendekatan eksperimen kuantitatif, dimana simulasi serangan SSH *bruteforce* dilakukan melalui jaringan lokal pada lingkungan mesin virtual yang telah dikonfigurasi secara tersegmentasi. *Honeypot* Cowrie diintegrasikan untuk menangkap, merekam, dan mengklasifikasikan seluruh aktivitas mencurigakan, termasuk percobaan *login* tidak sah dan perintah pasca-kompromi. Hasil pengujian menunjukkan bahwa seluruh upaya serangan SSH berhasil dialihkan dan dicatat secara rinci oleh *honeypot* tanpa mengganggu kinerja layanan *email* utama. Analisis data *log* memperlihatkan pola perilaku penyerang yang sistematis, mulai dari enumerasi sistem hingga upaya *privilege escalation*. Simpulan penelitian ini menegaskan bahwa implementasi *honeypot* Cowrie efektif sebagai deteksi dini dan dokumentasi aktivitas serangan pada *mail server* berbasis Linux, serta dapat diadopsi sebagai model proteksi tambahan untuk memperkuat infrastruktur keamanan jaringan.

Kata Kunci: *Mail Server*, *Honeypot*, Cowrie, iRedMail, Keamanan Jaringan, SSH *Bruteforce*, Deteksi Dini, Log Analisis, Linux

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang Masalah

Email merupakan salah satu layanan yang paling banyak digunakan dalam komunikasi melalui internet. Namun, layanan *email* tidak sepenuhnya aman sebagai media komunikasi, dikarenakan terdapat beberapa celah keamanan pada protokol yang digunakannya. Celah tersebut memungkinkan penyerang *(attacker)* untuk mengeksploitasi layanan *email*, seperti mengirimkan *email* palsu *(email spoofing)*, atau bahkan melumpuhkan layanan *email* secara sengaja melalui serangan yang bertujuan meminta tebusan *(ransomware)*. Dalam dunia industri maupun perusahaan, *email* menjadi media komunikasi utama, baik dalam koordinasi antar karyawan maupun dalam berkomunikasi dengan perusahaan lainnya. Oleh karena itu, sangat disarankan bagi perusahaan untuk memiliki *mail server* pribadi dibandingkan bergantung sepenuhnya pada layanan *email* publik. Penggunaan *mail server* pribadi *(private mail server)* memberikan beberapa keunggulan, di antaranya adalah tingkat keamanan data yang lebih terjamin serta efisiensi waktu dalam proses pengiriman dan penerimaan *email*. Hal ini dikarenakan trafik data pada *mail server* pribadi umumnya lebih rendah dibandingkan *mail server* publik, sehingga mempercepat pertukaran informasi serta mengurangi risiko kemacetan data *(traffic congestion)* (Larasati et al., 2021).

Akan tetapi, pembangunan suatu *mail server* harus didukung dengan implementasi sistem keamanan jaringan yang efektif. Hal ini bertujuan untuk mengantisipasi berbagai jenis ancaman atau serangan siber yang bertujuan mencuri data sensitif perusahaan maupun memperlambat kinerja *mail server* tersebut. Salah satu bentuk serangan yang umum terjadi pada *mail server* adalah serangan *Denial of Service* (DoS). Serangan ini diketahui terus mengalami peningkatan signifikan dalam beberapa tahun terakhir. Menurut laporan dari Cloudflare yang dikutip oleh Sam Cook dalam ulasannya di situs *comparitech.com,* intensitas serangan DoS mengalami peningkatan hampir sepertiga antara tahun 2020 hingga 2021. Lebih lanjut, tercatat pula adanya lonjakan sebesar 75% pada kuartal keempat tahun 2021 dibandingkan dengan kuartal sebelumnya. Fakta tersebut menunjukkan bahwa ancaman serangan DoS perlu diwaspadai secara khusus dalam desain keamanan

jaringan *mail server* (Cook, 2022). Kejadian serangan DoS terhadap layanan *email* juga pernah tercatat pada tanggal 21 Oktober 2021. Pada saat itu, setidaknya delapan penyedia layanan *email*, yaitu Runbox, Posteo, Fastmail, TheXYZ, Guerilla Mail, Mailfence, Kolab Now, dan RiseUP, menjadi sasaran serangan *Denial of Service* secara bersamaan. Setelah serangan terjadi, penyerang mengirimkan *email* berisi permintaan tebusan sebesar 0,06 BTC (sekitar 4000 dolar AS). Dalam pesan tersebut, pelaku mengancam akan melumpuhkan jaringan layanan *email* secara berkelanjutan apabila permintaan tebusan tidak dipenuhi. Peristiwa ini menunjukkan pentingnya perhatian terhadap ancaman serangan DoS pada layanan *email* serta urgensi implementasi sistem keamanan jaringan yang lebih kuat guna mencegah dampak negatif yang ditimbulkan (Cimpanu, 2021). Serangan DoS bertujuan menghalangi pengguna yang sah untuk mengakses sumber daya jaringan atau sistem dengan cara mengirimkan permintaan palsu secara masif sehingga menghabiskan sumber daya jaringan secara signifikan. Akibatnya, konsumsi energi meningkat, waktu tunda *(delay)* bertambah, kapasitas jaringan *(throughput)* berkurang, serta serangan menjadi sulit dideteksi (Kurniawan & Yazid, 2020). Salah satu jenis serangan DoS adalah SYN *Flood*, yang mengeksploitasi mekanisme *three-way handshake* pada *Transmission Control Protocol* (TCP). Pada serangan ini, penyerang mengirimkan sejumlah besar paket SYN secara terus-menerus tanpa menanggapi paket SYN-ACK dari *server*, sehingga sumber daya *server* terkuras untuk menangani permintaan palsu tersebut, menyebabkan layanan menjadi tidak tersedia (Degirmencioglu et al., 2016). Selain SYN *Flood*, serangan *bruteforce* juga merupakan ancaman nyata, penyerang menggunakan daftar kombinasi *username* dan *password* yang disiapkan untuk mencoba masuk ke sistem secara ilegal (Prasetyo et al., 2020).

Untuk mengatasi ancaman tersebut, penelitian ini bertujuan membangun sebuah sistem keamanan jaringan *mail server* dengan mengimplementasikan konsep *Honeypot*. *Honeypot* adalah sistem keamanan proaktif yang berfungsi untuk menangkap informasi tentang aktivitas mencurigakan seperti waktu serangan, alamat IP penyerang, sistem operasi penyerang, eksploitasi, dan perintah setelah infiltrasi (Fraunholz et al., 2017).

1.2 Perumusan Masalah

Rumusan masalah yang Penulis temukan dari latar belakang di atas antara lain:

1. Bagaimana merancang arsitektur *mail server* berbasis iRedMail pada mesin virtual dengan sistem operasi Debian 12?
2. Bagaimana mengonfigurasi *honeypot* Cowrie sebagai sistem deteksi dini aktivitas mencurigakan seperti SSH *bruteforce*?
3. Sejauh mana efektivitas implementasi *honeypot* dalam mendeteksi dan mencegah serangan terhadap *mail server* utama?

1.3 Batasan Masalah

Agar tidak terlampau luas dalam pembahasan ini, maka diterapkan batasan-batasan masalah sehingga tujuan dari pembuatan dari skripsi dapat tercapai.

Batasan-batasan masalah tersebut adalah sebagai berikut:

a. Pengujian dilakukan menggunakan tiga mesin virtual dengan rincian berikut: 1. Debian 12 sebagai *mail server* (iRedMail) sekaligus *honeypot* Cowrie; 2. Kali Linux 2024.2 sebagai attacker.
b. Pengujian dilakukan dalam jaringan lokal (LAN) dengan subnet 192.168.14.0/24 menggunakan metode NAT *Adapter* dengan IP statis.
c. Jenis serangan yang disimulasikan adalah SSH *bruteforce*.

1.4 Tujuan dan Manfaat

Adapun Tujuan dan Manfaat dari perancangan sistem keamanan jaringan pada *mail server* adalah:

Tujuan

a. Membangun dan mengonfigurasi *mail server* berbasis iRedMail pada mesin virtual Debian 12.
b. Mengimplementasikan *honeypot* Cowrie (SSH) sebagai sistem deteksi dini keamanan *server* dalam menangani serangan SSH *Bruteforce*.
c. Mengukur efektivitas *honeypot* dalam mendeteksi serangan serta menyajikan visualisasi *log* aktivitas mencurigakan.

Manfaat

a. Sebagai panduan praktis bagi institusi dalam membangun *mail server* yang aman.
b. Menghasilkan data intelijen serangan yang berguna untuk analisis forensik serta pengembangan strategi keamanan jaringan berbasis *honeypot* di masa mendatang.
c. Meminimalkan risiko *downtime* dan kebocoran data akibat serangan siber.

1.5 Sistematika Penulisan

Sistematika penulisan skripsi ini disusun secara terstruktur untuk memudahkan pembaca dalam memahami alur penelitian yang dilakukan. Adapun susunan bab dalam penulisan skripsi ini adalah sebagai berikut:

1. BAB I PENDAHULUAN

Bab ini memuat latar belakang penelitian, batasan masalah, rumusan masalah, tujuan penelitian, serta manfaat yang diharapkan dari penyusunan skripsi ini.

2. BAB II TINJAUAN PUSTAKA

Pada bab ini dipaparkan kajian pustaka yang relevan dengan topik penelitian. Penulis mengkaji berbagai sumber seperti jurnal, buku, dan referensi lain yang mendukung pemahaman terhadap permasalahan yang diangkat.

3. BAB III METODE PENELITIAN

Bab ini menjelaskan metodologi penelitian yang digunakan, meliputi teknik pengumpulan data, tahapan studi pustaka, analisis umum, serta deskripsi objek penelitian yang menjadi fokus dalam skripsi ini.

4. BAB IV HASIL DAN PEMBAHASAN

Pada bab ini dipaparkan hasil pengujian dan pembahasan terkait implementasi sistem, mulai dari proses persiapan hingga uji penetrasi terhadap mail server yang telah dilengkapi dengan honeypot sebagai sistem keamanan. Analisis dilakukan untuk mengetahui efektivitas sistem dalam mendeteksi dan mengidentifikasi ancaman, serta sejauh mana tujuan penelitian pada bab I telah tercapai.

5. BAB V PENUTUP

Bab ini berisi kesimpulan dari hasil penelitian dan analisis yang telah dilakukan, serta saran-saran untuk pengembangan sistem sejenis di masa mendatang agar hasil yang diperoleh dapat lebih optimal.

6. DAFTAR PUSTAKA

Berisi daftar sumber referensi yang digunakan selama proses penyusunan skripsi, baik berupa jurnal, buku, maupun tautan situs *web*.

7. DAFTAR RIWAYAT HIDUP

Bagian ini memuat biodata penulis beserta riwayat pendidikan yang telah ditempuh.

8. LAMPIRAN

Bab ini berisi dokumen pendukung yang relevan dengan penelitian, seperti foto, data hasil pengujian, atau berkas lain yang mendukung penulisan skripsi

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan mengenai implementasi *honeypot* pada *mail server* berbasis Linux, dapat diambil beberapa kesimpulan utama sebagai berikut:

1) Implementasi *mail server* berbasis iRedMail pada lingkungan virtual telah berhasil dilakukan dan dapat berjalan sesuai fungsinya sebagai media komunikasi *email* internal.
2) Integrasi *honeypot* Cowrie pada *mail server* efektif dalam mendeteksi dan merekam aktivitas serangan SSH *brute force*. Semua upaya akses ilegal yang diarahkan ke layanan SSH berhasil dialihkan dan didokumentasikan oleh sistem *honeypot* tanpa mengganggu operasional utama *mail server*.
3) Pengujian pada jaringan lokal menunjukkan bahwa sistem yang dibangun mampu memberikan data *log* serangan secara rinci, yang sangat berguna sebagai bahan analisis dan evaluasi keamanan jaringan.

5.2 Saran

Berdasarkan keterbatasan yang ada dalam penelitian ini, terdapat beberapa hal yang perlu dipertimbangkan untuk pengembangan ke depan. Pertama, pengujian sebaiknya dilakukan langsung pada *server* produksi agar hasil yang diperoleh lebih mencerminkan tantangan keamanan di lingkungan nyata. Kedua, penelitian selanjutnya disarankan untuk menguji sistem pada jaringan publik (WAN), sehingga cakupan deteksi dan variasi serangan yang dapat diamati menjadi lebih luas. Ketiga, penggunaan jenis *honeypot* dan simulasi serangan dapat diperluas, tidak hanya terbatas pada Cowrie dan serangan SSH *bruteforce*. Dengan mengadopsi *honeypot* lain serta memvariasikan jenis serangan, hasil penelitian diharapkan mampu memberikan gambaran yang lebih komprehensif mengenai upaya mitigasi ancaman keamanan jaringan.

DAFTAR PUSTAKA

Althobaiti, A. H., AlZain, M. A., Al-Amri, J., Baz, M., & Masud, M. (2019). An Extensive Study of Honeypot Technique. *International Journal of Advanced Trends in Computer Science and Engineering*, *8*(6), 3318–3326. https://doi.org/10.30534/ijatcse/2019/103862019

Anand, S., Mathikshara, P., & Jayavignesh, T. (2019). An Efficient Mask Reduction Strategy to Optimize Storage and Computational Complexity in Routing Table Lookups. *2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, 1–5.

Arief, S. (2019). Analisis Perbandingan Mail Server Open Source: Studi Kasus Zimbra, iRedMail, dan Postfix. *Jurnal Telematika*, *12*(1), 19–27. https://jurnal.stmikroyal.ac.id/index.php/telematika/article/download/54/40

Aulia, D. R., Rahman, W. A., & Zhacque, V. A. (2025). Implementasi dan analisis honeypot berbasis cowrie untuk mendeteksi serangan siber. *Jurnal METHODIKA*, *11*(1), 35–39.

Azginoglu, N. (2020). An open source mail server migration experience: Iredmail. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, *44*(4/W3), 95–98. https://doi.org/10.5194/isprs-archives-XLIV-4-W3-2020-95-2020

Azzahri, M. N., Selian, R. A., Muchallil, S., & Nurdin, Y. (2024). The Application of Cowrie Honeypot to Analyze Attacks on SSH and Telnet Protocols. *2024 IEEE 2nd International Conference on Electrical Engineering, Computer and Information Technology (ICEECIT)*, (tidak tersedia)-(tidak tersedia). https://doi.org/10.1109/ICEECIT63698.2024.10859786

Cimpanu, C. (2021). *DDoS attacks hit multiple email providers*. October 25. https://therecord.media/ddos-attacks-hit-multiple-email-providers/

Cook, S. (2022). *20+ DDoS attack statistics and facts for 2018-2022*. 2022 Updated on November 2nd. https://www.comparitech.com/blog/information-

security/ddos-statistics-facts/

Degirmencioglu, A., Erdogan, H. T., Mizani, M. A., & Yılmaz, O. (2016). *A Classification Approach For Adaptive Mitigation of SYN Flood Attacks*. *AnNet*, 1109–1113.

Franco, J., Aris, A., Canberk, B., & Uluagac, A. S. (2021). A Survey of Honeypots and Honeynets for Internet of Things, Industrial Internet of Things, and Cyber-Physical Systems. *IEEE Communications Surveys and Tutorials*, *23*(4), 2351–2383. https://doi.org/10.1109/COMST.2021.3106669

Fraunholz, D., Zimmermann, M., & Schotten, H. D. (2017). An adaptive honeypot configuration, deployment and maintenance strategy. *International Conference on Advanced Communication Technology, ICACT*, 53–57. https://doi.org/10.23919/ICACT.2017.7890056

Hetzler, C., Chen, Z., & Khan, T. M. (2023). Analysis of SSH Honeypot Effectiveness. In *Lecture Notes in Networks and Systems: Vol. 652 LNNS* (pp. 759–782). https://doi.org/10.1007/978-3-031-28073-3_51

Hills, M., Klint, P., & Vinju, J. (2013). An Empirical Study of PHP Feature Usage. *ISSTA 2013 Proceedings of the 2013 International Symposium on Software Testing and Analysis*, 325–335. http://homepages.cwi.nl/~jurgenv/papers/ISSTA-2013.pdf

iRedMail. (2024a). [SOLVED] iRedMail + MariaDB. In *iRedMail Forum*. https://forum.iredmail.org/topic6436-mariadb.html

iRedMail. (2024b). iRedMail Documentation. In *iRedMail Docs*. https://docs.iredmail.org/

Karamollahi, M., & Williamson, C. (2019). Characterization of IMAPS email traffic. *Proceedings - IEEE Computer Society's Annual International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems, MASCOTS*, *2019-Octob*, 214–220. https://doi.org/10.1109/MASCOTS.2019.00030

Kose, U., Sert, S., & Yildirim, R. (2020). *An Open Source Mail Server Migration*

Experience: iRedMail. Nevsehir University. https://www.researchgate.net/publication/346328567_AN_OPEN_SOURCE_MAIL_SERVER_MIGRATION_EXPERIENCE_IREDMAIL

Küçükkarakurt, F. (2022). *Introduction to Linux Postfix Architecture for Beginners*. May 9, 2022. https://www.makeuseof.com/postfix-architecture-linux/

Kurniawan, M. T., & Yazid, S. (2020). *Mitigation and Detection Strategy of DoS Attack on Wireless Sensor Network Using Blocking Approach and Intrusion Detection System*. *June*, 1–5.

Lammle, T. (2020). *Chapter 9 IP Routing*. *1*, 243–285.

Larasati, N. N., Ikhwan, S., & Wahyudi, E. (2021). Implementasi Mail Server Dengan Roundcube Menggunakan Postfix di CV Surya Mandiri Cilacap. *Dinamika Rekayasa*, *17*(2), 95. https://doi.org/10.20884/1.dr.2021.17.2.364

Large-scale deployment. (2019). In *iRedMail Official Forum*. https://forum.iredmail.org/topic16351-large-scale-deployment.html

Li, X., Karnan, S., & Chishti, J. A. (2017). An empirical study of three PHP frameworks. *2017 4th International Conference on Systems and Informatics, ICSAI 2017*, *2018-Janua*(Icsai), 1636–1640. https://doi.org/10.1109/ICSAI.2017.8248546

Lihet, M. A., & Dadarlat, V. (2015). How to build a honeypot System in the cloud. *2015 14th RoEduNet International Conference - Networking in Education and Research, RoEduNet NER 2015 - Proceedings*, 190–194. https://doi.org/10.1109/RoEduNet.2015.7311992

Loshin, P. (2021). *Definition bash (bourne again shell)*. December. https://www.techtarget.com/searchdatacenter/definition/bash-Bourne-Again-Shell

Maly, I., & Mikovec, Z. (2010). Web Applications Usability Testing with Task Model Skeletons. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *6409 LNCS*, 158–165. https://doi.org/10.1007/978-3-642-16488-0_13

MariaDB. (2024). Storage Engines. In *MariaDB University*. https://uni.mariadb.org/dba/storage-engines/

Monty, M. (2024, October 30). *Celebrating 15 years of MariaDB Server | MariaDB*. https://mariadb.com/resources/blog/celebrating-15-years-of-mariadb-server/

Morić, Z., Dakić, V., & Regvart, D. (2025). Advancing Cybersecurity with Honeypots and Deception Strategies. *Informatics*, *12*(1). https://doi.org/10.3390/informatics12010014

Need advice on iRedMail deployment. (2010). In *iRedMail Official Forum*. https://forum.iredmail.org/topic891-need-advice-on-iredmail-deployment.html

Packer, A., & Dhamim. (2000). *Performance Analysis of IP Routing Lookup Algorithms : Patricia Tree based vs . Hashing based*. *April*.

Patel, K., Patel, R., & Bhatt, C. (2020). A Comparative Analysis of Open Source Mail Servers. *International Journal of Advanced Research in Computer Science*, *11*(5), 17–22. https://www.ijarcs.info/index.php/Ijarcs/article/view/6825/5401

Plesk. (2024). Dovecot. In *Plesk Wiki*. https://www.plesk.com/wiki/dovecot/

Prasetyo, K. A., Idhom, M., Wahanani, H. E., Informatika, P. S., & Komputer, F. I. (2020). *Sistem Pencegahan Serangan Bruteforce Pada Multiple server dengan menggunakan Fail2ban*. *01*(3), 709–715.

Quizlet. (2024). iptables Flash Cards. In *Quizlet*. https://quizlet.com/16951464/iptables-flash-cards/

Raikar, M. M., & Meena, S. M. (2021). SSH brute force attack mitigation in Internet of Things (IoT) network : An edge device security measure. *ICSCCC 2021 - International Conference on Secure Cyber Computing and Communications*, *July*, 72–77. https://doi.org/10.1109/ICSCCC51823.2021.9478131

Roundcube. (2024). Roundcube Webmail. In *Roundcube*. https://roundcube.net/

Saeed, F. (2024, January 17). *Securing MariaDB Server & MariaDB MaxScale Connections (TLS) | MariaDB*. https://mariadb.com/resources/blog/securing-mariadb-server-mariadb-maxscale-connections-tls/

Safargalieva, A., & Vasilomanolakis, E. (2025). Towards bio-inspired cyber-deception: a case study of SSH and Telnet honeypots. *Proceedings of the 4th Workshop on Active Defense and Deception (AD&D), Co-Located with the 10th IEEE European Symposium on Security and Privacy*, (not specified)-(not specified).

Setianto, F., Tsani, E., Sadiq, F., Domalis, G., Tsakalidis, D., & Kostakos, P. (2021). *GPT-2C: A GPT-2 parser for Cowrie honeypot logs* (Issue arXiv:2109.06595). https://arxiv.org/abs/2109.06595

Sharif, A. (2018, June 28). *Improve Performance of Galera Cluster for MySQL or MariaDB | Severalnines*. https://severalnines.com/blog/improve-performance-galera-cluster-mysql-or-mariadb

Sitepu, A. D., Valentino, B., Hatoguan, I. P., & Medan, U. N. (2024). *Iredmail server implementation on ubuntu*. *8*(11).

Swaminathan, D. G. A., & Sowndaryaa, M. M. (2025). TARGETED HONEYPOT DEPLOYMENT FOR ANALYZING AND MITIGATING THREATS. *International Journal for Scientific Research & Development*, *Vol. 8*.

Taran, A., & Silnov, D. S. (2017). Research of attacks on MySQL servers using HoneyPot technology. *Proceedings of the 2017 IEEE Russia Section Young Researchers in Electrical and Electronic Engineering Conference, ElConRus 2017*, 224–226. https://doi.org/10.1109/EIConRus.2017.7910533

Tongkaw, S., & Tongkaw, A. (2017). A comparison of database performance of MariaDB and MySQL with OLTP workload. *ICOS 2016 - 2016 IEEE Conference on Open Systems*, *October 2016*, 117–119. https://doi.org/10.1109/ICOS.2016.7881999

Tran, C. P., & Tran, D. K. (2018). Anomaly Detection in POSTFIX mail log using Principal Component Analysis. *Proceedings of 2018 10th International*

Conference on Knowledge and Systems Engineering, KSE 2018, 107–112. https://doi.org/10.1109/KSE.2018.8573410

Wanjau, S. K., Wambugu, G. M., & Kamau, G. N. (2021). SSH-Brute Force Attack Detection Model based on Deep Learning. *International Journal of Computer Applications Technology and Research*, *10*(01), 42–50. https://doi.org/10.7753/ijcatr1001.1008

Wei, X. (2012). *Based on VMware technology ' s Campus network cloud platform technology research*. 430–433. https://doi.org/10.1109/ICCSEE.2012.23

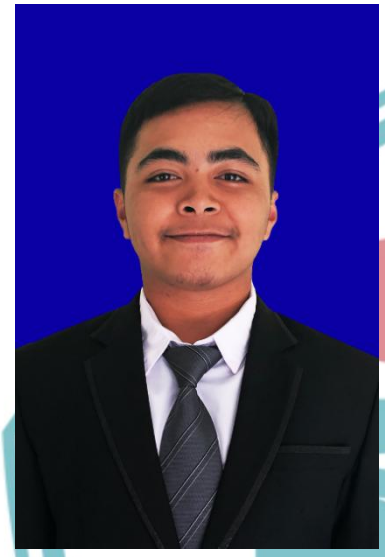
Wu, Q.-X. (2012). The Research and Application of Firewall based on Netfilter. *Physics Procedia*, *25*, 1231–1235. https://doi.org/10.1016/J.PHPRO.2012.03.225

Wu, Y., Cao, P. M., Withers, A., Kalbarczyk, Z. T., & Iyer, R. K. (2021). *Mining Threat Intelligence from Billion-scale SSH Brute-Force Attacks*. 8–10. https://doi.org/10.14722/diss.2020.23007

Yang, X., Yuan, J., Yang, H., Kong, Y., Zhang, H., & Zhao, J. (2023). A Highly Interactive Honeypot-Based Approach to Network Threat Management. *Future Internet*, *15*(4), 1–31. https://doi.org/10.3390/fi15040127

DAFTAR RIWAYAT HIDUP

DAFTAR RIWAYAT HIDUP PENULIS



Aldy Widanto, 1 Mei 1999

Menyelesaikan pendidikan dari SDN Palalangon 02 pada tahun 2011. SMPN 11 Bogor pada tahun 2014, dan SMKN 3 Bogor pada tahun 2017. Mengikuti program *computer education* di *Continuing Education Program–Center for Computing and Information Technology Fakultas Teknik Universitas Indonesia* Lulus tahun 2019.

LAMPIRAN

Lampiran - 1 Isi *file* konfigurasi *cowrie.cfg.dist*

```
# DO NOT EDIT THIS FILE!
# Changes to default files will be lost on update and are difficult to
# manage and support.
#
# Please make any changes to system defaults by overriding them in
# cowrie.cfg
#
# To override a specific setting, copy the name of the stanza and
# setting to the file where you wish to override it.

# ============================================================================
# General Cowrie Options
# ============================================================================
[honeypot]

# Sensor name is used to identify this Cowrie instance. Used by the database
# logging modules such as mysql.
#
# If not specified, the logging modules will instead use the IP address of the
# server as the sensor name.
#
# (default: not specified)
#sensor_name=myhostname

# Hostname for the honeypot. Displayed by the shell prompt of the virtual
# environment
#
# (default: svr04)
hostname = mail-srv


# Directory where to save log files in.
#
# (default: log)
log_path = var/log/cowrie
```

```
# Directory where to save downloaded artifacts in.
#
# (default: downloads)
download_path = ${honeypot:state_path}/downloads


# Directory for static data files
#
# (default: src/cowrie/cowrie)
data_path = src/cowrie/data


# Directory for variable state files
#
# (default: var/lib/cowrie)
state_path = var/lib/cowrie


# Directory for config files
#
# (default: etc)
etc_path = etc


# Directory where virtual file contents are kept in.
#
# This is only used by commands like 'cat' to display the contents of files.
# Adding files here is not enough for them to appear in the honeypot - the
# actual virtual filesystem is kept in filesystem_file (see below)
#
# (default: honeyfs)
contents_path = honeyfs


# Directory for creating simple commands that only output text.
```

Lampiran - 3 Isi *file* konfigurasi *cowrie.cfg.dist* (lanjutan)

```
# Directory for creating simple commands that only output text.
#
# The command must be placed under this directory with the proper path, such
# as:
#   txtcmds/usr/bin/vi
# The contents of the file will be the output of the command when run inside
# the honeypot.
#
# In addition to this, the file must exist in the virtual filesystem
#
# (default: txtcmds)
txtcmds_path = txtcmds


# Maximum file size (in bytes) for downloaded files to be stored in 'download_path'.
# A value of 0 means no limit. If the file size is known to be too big from the start,
# the file will not be stored on disk at all.
#
# (default: 0)
#download_limit_size = 10485760

# TTY logging will log a transcript of the complete terminal interaction in UML
# compatible format.
# (default: true)
ttylog = true

# Default directory for TTY logs.
# (default: ttylog_path = %(state_path)s/tty)
ttylog_path = ${honeypot:state_path}/tty

# Idle timeout determines when logged in sessions are
# terminated for being idle. In seconds.
# (default: 180)
idle_timeout = 180
```

Lampiran - 4 Isi *file* konfigurasi *cowrie.cfg.dist* (lanjutan)

```
# Authentication Timeout
# The server disconnects after this time if the user has not successfully logged in.
# The default is 120 seconds.
authentication_timeout = 120

# EXPERIMENTAL: back-end to user for Cowrie, options: proxy or shell
# (default: shell)
backend = shell

# Logging Type
# Valid options are `rotating` and `plain`.
# Without a config file, the default is `plain`.
#
# Use `rotating` and Cowrie will log to `cowrie.log` and at midnight
# rotate to `cowrie.log-<year>-<month>-<date>.log` and continue writing
# to (a new) `cowrie.log`.
# `plain` will write to `cowrie.log` and will not rotate. Use `plain`
# if you want to use external log rotation solutions like `logrotate`
logtype = rotating

# Timezone Cowrie uses for logging
# This can be any valid timezone for the TZ environment variable
# The special value `system` will let Cowrie use the system time zone
# `system` is not recommended because you will need to deal with daylight
# savings time and other special cases yourself when analysing the logs.
timezone = UTC

# Custom prompt
# By default, Cowrie creates a shell prompt like: root@svr03:~#
# If you want something totally custom, uncomment the option below and set your prompt
# Beware that the path won't be included in your prompt any longer
# prompt = hello>
```

Lampiran - 5 Isi *file* konfigurasi *cowrie.cfg.dist* (lanjutan)

```
auth_class = UserDB

# When AuthRandom is used also set the
#  auth_class_parameters: <min try>, <max try>, <maxcache>
#  for example: 2, 5, 10 = allows access after randint(2,5) attempts
#  and cache 10 combinations.
#
#auth_class = AuthRandom
#auth_class_parameters = 2, 5, 10


[backend_pool]
# ============================================================================
# Backend Pool Configurations
# only used on the cowrie instance that runs the pool
# ============================================================================

# enable this to solely run the pool, regardless of other configurations (disables SSH and Telnet)
pool_only = false

# time between full VM recycling (cleans older VMs and boots newer ones) - involves some downtime betwe>
# -1 to disable in seconds
recycle_period = 1500

# change interface below to allow connections from outside (e.g. remote pool)
listen_endpoints = tcp:6415:interface=127.0.0.1

# guest snapshots
save_snapshots = false
snapshot_path = ${honeypot:state_path}/snapshots

# pool xml configs
config_files_path = ${honeypot:data_path}/pool_configs

network_config = default_network.xml
nw_filter_config = default_filter.xml
```

```
# libvirt URI, common settings are qemu:///system or qemu:///session
libvirt_uri = qemu:///system
# Use this syntax to directly connect to the UNIX socket
# libvirt_uri = qemu+unix:///session?socket=/home/cowrie/.cache/libvirt/libvirt-sock

# ====================================
# Guest details (for a generic x86-64 guest, like Ubuntu)
#
# Used to provide configuration details to save snapshots, identify
# running guests, and provide other details to Cowrie.
#   - SSH and Telnet ports: which ports are listening for these services in the guest OS;
#     if you're not using one of them omit the config or set to 0
#   - Guest private key: used by the pool to control the guest's state via SSH; guest must
#     have the corresponding pubkey in root's authorized_keys (not implemented)
# ====================================
guest_config = default_guest.xml
guest_privkey = ${honeypot:state_path}/ubuntu18.04-guest
guest_tag = ubuntu18.04
guest_ssh_port = 22
guest_telnet_port = 23

# Configs below are used on default XMLs provided.
# If you provide your own XML in guest_config you don't need these configs.
#
# Guest hypervisor can be qemu or kvm, for example. Recent hardware has KVM,
# which is more performant than the qemu software-based emulation. Guest arch
# must match your machine's. If it's older or you're unsure, set it to 'qemu'.
#
# Memory size is in MB.
#
# Advanced: guest_qemu_machine defines which machine Qemu emulates for your VM
# If you get a "unsupported machine type" exception when VMs are loading, change
# it to a compatible machine listed by the command: 'qemu-system-x86_64 -machine help'
guest_image_path = /home/cowrie/cowrie-imgs/ubuntu18.04-minimal.qcow2
guest_hypervisor = kvm
guest_memory = 512
```

Lampiran - 7 Isi *file* konfigurasi *cowrie.cfg.dist* (lanjutan)

```
guest_qemu_machine = pc-q35-bionic

# ====================================
# Guest details (for OpenWRT with ARM architecture)
#
# Used to provide configuration details to save snapshots, identify running guests,
# and provide other details to Cowrie.
# ====================================
#guest_config = wrt_arm_guest.xml
#guest_tag = wrt
#guest_ssh_port = 22
#guest_telnet_port = 23

# Configs below are used on default XMLs provided.
# If you provide your own XML in guest_config you don't need these configs.
#
# Guest hypervisor can be qemu or kvm, for example. Recent hardware has KVM,
# which is more performant than the qemu software-based emulation. Guest arch
# must match your machine's.
#
# Memory size is in MB.
#
# Advanced: guest_qemu_machine defines which machine Qemu emulates for your VM
# If you get a "unsupported machine type" exception when VMs are loading, change
# it to a compatible machine listed by the command: 'qemu-system-arm -machine help'
#guest_image_path = /home/cowrie/cowrie-imgs/root.qcow2
#guest_hypervisor = qemu
#guest_memory = 256
#guest_kernel_image = /home/cowrie/cowrie-imgs/zImage
#guest_qemu_machine = virt-2.9

# ====================================
# Other configs
# ====================================
# Use NAT (for remote pool)
#
```

Lampiran - 8 Isi *file* konfigurasi *cowrie.cfg.dist* (lanjutan)

```
# Guests exist in a local interface created by libvirt; NAT functionality creates a port in the host,
# exposed to a public interface, and forwards TCP data to and from the libvirt private interface.
# Cowrie's proxy receives the public information instead of the local IP of guests.
use_nat = true
nat_public_ip = 192.168.1.40


# ============================================================================
# Proxy Options
# ============================================================================
[proxy]

# type of backend:
#   - simple: backend machine deployed by you (CAREFUL WITH SECURITY ASPECTS!!), specify hosts and ports
#   - pool: cowrie-managed pool of virtual machines, configure below
backend = pool

# ====================================
# Simple Backend Configuration
# ====================================
backend_ssh_host = localhost
backend_ssh_port = 2022

backend_telnet_host = localhost
backend_telnet_port = 2023

# ====================================
# Pool Backend Configuration
# ====================================

# generic pool configurable settings
pool_max_vms = 5
pool_vm_unused_timeout = 600

# allow sharing guests between different attackers if no new VMs are available
```

Lampiran - 9 Isi *file* konfigurasi *cowrie.cfg.dist* (lanjutan)

```
pool_share_guests = true

# Where to deploy the backend pool (only if backend = pool)
#   - "local": same machine as the proxy
#   - "remote": set host and port of the pool below
pool = local

# Remote pool configurations (used with pool=remote)
pool_host = 192.168.1.40
pool_port = 6415

# ====================================
# Proxy Configurations
# ====================================

# real credentials to log into backend
backend_user = root
backend_pass = root

# Telnet prompt detection
#
# To detect authentication prompts (and spoof auth details to the ones the backend accepts) we need to
# login and password prompts, and spoof data to the backend in order to successfully authenticate. If d
# attackers can only use the real user credentials of the backend.
telnet_spoof_authentication = true

# These regex were made using Ubuntu 18.04; you have to adapt these for the prompts
# from your backend. You can enable raw logging above to analyse data passing through
# and identify the format of the prompts you need.
# You should generally include ".*" at the beginning and end of prompts, since Telnet messages can cont
# more data than the prompt.

# For login it is usually <hostname> login:
telnet_username_prompt_regex = (\n|^)ubuntu login: .*

# Password prompt is usually only the word Password
```

Lampiran - 10 Isi *file* konfigurasi *cowrie.cfg.dist* (lanjutan)

```
# Password prompt is usually only the word Password
telnet_password_prompt_regex = .*Password: .*

# This data is sent by clients at the beginning of negotiation (before the password prompt), and contai
# that is trying to log in. We replace that username with the one in "backend_user" to allow the chance
# login after the first password prompt. We are only able to check if credentials are allowed after the
# inserted. If they are, then a correct username was already sent and authentication succeeds; if not,
# password to force authentication to fail.
telnet_username_in_negotiation_regex = (.*\xff\xfa.*USER\x01)(.*?)(\xff.*)

# Other configs #
# log raw TCP packets in SSh and Telnet
log_raw = false


# ============================================================================
# Shell Options
# Options around Cowrie's Shell Emulation
# ============================================================================

[shell]

# File in the Python pickle format containing the virtual filesystem.
#
# This includes the filenames, paths, permissions for the Cowrie filesystem,
# but not the file contents. This is created by the bin/createfs utility from
# a real template linux installation.
#
# (default: fs.pickle)
filesystem = ${honeypot:data_path}/fs.pickle


# File that contains output for the `ps` command.
#
# (default: ${honeypot_data_path}/cmdoutput.json)
processes = ${honeypot:data_path}/cmdoutput.json
```

Lampiran - 11 Isi *file* konfigurasi *cowrie.cfg.dist* (lanjutan)

```
# Fake architectures/OS
# When Cowrie receive a command like /bin/cat XXXX (where XXXX is an executable)
# it replies with the content of a dummy executable (located in data_path/arch)
# compiled for an architecture/OS/endian_mode
# arch can be a comma separated list. When there are multiple elements, a random
# is chosen at login time.
# (default: linux-x64-lsb)

arch = linux-x64-lsb

# Here the list of supported OS-ARCH-ENDIANESS executables
# bsd-aarch64-lsb:          64-bit      LSB     ARM aarch64 version 1 (SYSV)
# bsd-aarch64-msb:          64-bit      MSB     ARM aarch64 version 1 (SYSV)
# bsd-bfin-msb:             32-bit      MSB     Analog Devices Blackfin version 1 (SYSV)
# bsd-mips64-lsb:               64-bit  LSB     MIPS MIPS-III version 1 (SYSV)
# bsd-mips64-msb:               64-bit  MSB     MIPS MIPS-III version 1 (SYSV)
# bsd-mips-lsb:             32-bit      LSB     MIPS MIPS-I version 1 (FreeBSD)
# bsd-mips-msb:             32-bit      MSB     MIPS MIPS-I version 1 (FreeBSD)
# bsd-powepc64-lsb:         64-bit      MSB     64-bit PowerPC or cisco 7500 version 1 (FreeBSD)
# bsd-powepc-msb:               32-bit  MSB     PowerPC or cisco 4500 version 1 (FreeBSD)
# bsd-riscv64-lsb:          64-bit      LSB     UCB RISC-V version 1 (SYSV)
# bsd-sparc64-msb:          64-bit      MSB     SPARC V9 relaxed memory ordering version 1 (FreeBSD)
# bsd-sparc-msb:                32-bit  MSB     SPARC version 1 (SYSV) statically
# bsd-x32-lsb:              32-bit      LSB     Intel 80386 version 1 (FreeBSD)
# bsd-x64-lsb:              64-bit      LSB     x86-64 version 1 (FreeBSD)
# linux-aarch64-lsb:    64-bit  LSB     ARM aarch64 version 1 (SYSV)
# linux-aarch64-msb:    64-bit  MSB     ARM aarch64 version 1 (SYSV)
# linux-alpha-lsb:          64-bit      LSB     Alpha (unofficial) version 1 (SYSV)
# linux-am33-lsb:               32-bit  LSB     Matsushita MN10300 version 1 (SYSV)
# linux-arc-lsb:                32-bit  LSB     ARC Cores Tangent-A5 version 1 (SYSV)
# linux-arc-msb:                32-bit  MSB     ARC Cores Tangent-A5 version 1 (SYSV)
# linux-arm-lsb:                32-bit  LSB     ARM EABI5 version 1 (SYSV)
# linux-arm-msb:                32-bit  MSB     ARM EABI5 version 1 (SYSV)
# linux-avr32-lsb:          32-bit      LSB     Atmel AVR 8-bit version 1 (SYSV)
# linux-bfin-lsb:               32-bit  LSB     Analog Devices Blackfin version 1 (SYSV)
```

Lampiran - 12 Isi *file* konfigurasi *cowrie.cfg.dist* (lanjutan)

```
# Modify the response of '/bin/uname'
# Default (uname -a): Linux <hostname> <kernel_version> <kernel_build_string> <hardware_platform> <oper>
kernel_version = 3.2.0-4-amd64
kernel_build_string = #1 SMP Debian 3.2.68-1+deb7u1
hardware_platform = x86_64
operating_system = GNU/Linux

# SSH Version as printed by "ssh -V" in shell emulation
ssh_version = OpenSSH_7.9p1, OpenSSL 1.1.1a  20 Nov 2018


# ============================================================================
# SSH Specific Options
# ============================================================================
[ssh]

# Enable SSH support
# (default: true)
enabled = true


# Public and private SSH key files. If these don't exist, they are created
# automatically.
rsa_public_key = ${honeypot:state_path}/ssh_host_rsa_key.pub
rsa_private_key = ${honeypot:state_path}/ssh_host_rsa_key
dsa_public_key = ${honeypot:state_path}/ssh_host_dsa_key.pub
dsa_private_key = ${honeypot:state_path}/ssh_host_dsa_key
ecdsa_public_key = ${honeypot:state_path}/ssh_host_ecdsa_key.pub
ecdsa_private_key = ${honeypot:state_path}/ssh_host_ecdsa_key
ed25519_public_key = ${honeypot:state_path}/ssh_host_ed25519_key.pub
ed25519_private_key = ${honeypot:state_path}/ssh_host_ed25519_key

# Public keys supported are: ssh-rsa, ssh-dss, ecdsa-sha2-nistp256, ssh-ed25519
public_key_auth = ssh-rsa,ecdsa-sha2-nistp256,ssh-ed25519

# SSH version string as present to the client.
```

Lampiran - 13 Isi *file* konfigurasi *cowrie.cfg.dist* (lanjutan)

```
# SSH version string as present to the client.
#
# Version string MUST start with SSH-2.0- or SSH-1.99-
#
# Use these to disguise your honeypot from a simple SSH version scan
# Examples:
# SSH-2.0-OpenSSH_5.1p1 Debian-5
# SSH-1.99-OpenSSH_4.3
# SSH-1.99-OpenSSH_4.7
# SSH-1.99-Sun_SSH_1.1
# SSH-2.0-OpenSSH_4.2p1 Debian-7ubuntu3.1
# SSH-2.0-OpenSSH_4.3
# SSH-2.0-OpenSSH_4.6
# SSH-2.0-OpenSSH_5.1p1 Debian-5
# SSH-2.0-OpenSSH_5.1p1 FreeBSD-20080901
# SSH-2.0-OpenSSH_5.3p1 Debian-3ubuntu5
# SSH-2.0-OpenSSH_5.3p1 Debian-3ubuntu6
# SSH-2.0-OpenSSH_5.3p1 Debian-3ubuntu7
# SSH-2.0-OpenSSH_5.5p1 Debian-6
# SSH-2.0-OpenSSH_5.5p1 Debian-6+squeeze1
# SSH-2.0-OpenSSH_5.5p1 Debian-6+squeeze2
# SSH-2.0-OpenSSH_5.8p2_hpn13v11 FreeBSD-20110503
# SSH-2.0-OpenSSH_5.9p1 Debian-5ubuntu1
# SSH-2.0-OpenSSH_6.0p1 Debian-4+deb7u2
# SSH-2.0-OpenSSH_5.9
#
# (default: "SSH-2.0-OpenSSH_6.0p1 Debian-4+deb7u2")
version = SSH-2.0-OpenSSH_6.0p1 Debian-4+deb7u2

# Cipher encryption algorithms to be used.
#
# MUST be supplied as a comma-separated string without
# any spaces or newlines.
#
# Use ciphers to limit to more secure algorithms only
# any spaces.
```

Lampiran - 14 Isi *file* konfigurasi *cowrie.cfg.dist* (lanjutan)

```
# Use ciphers to limit to more secure algorithms only
# any spaces.
# Supported ciphers:
#
# aes128-ctr
# aes192-ctr
# aes256-ctr
# aes256-cbc
# aes192-cbc
# aes128-cbc
# 3des-cbc
# cast128-cbc
ciphers = aes128-ctr,aes192-ctr,aes256-ctr,aes256-cbc,aes192-cbc,aes128-cbc,3des-cbc,cast128-cbc


# MAC Algorithm to be used.
#
# MUST be supplied as a comma-separated string without
# any spaces or newlines.
#
# hmac-sha1 and hmac-md5 are considered insecure now, and
# instead MACs with higher number of bits should be used.
#
# Supported HMACs:
# hmac-sha2-512
# hmac-sha2-384
# hmac-sha2-256
# hmac-sha1
# hmac-md5
macs = hmac-sha2-512,hmac-sha2-384,hmac-sha2-256,hmac-sha1,hmac-md5


# Compression Method to be used.
#
# MUST be supplied as a comma-separated string without
```

Lampiran - 15 Isi *file* konfigurasi *cowrie.cfg.dist* (lanjutan)

```
# MUST be supplied as a comma-separated string without
# any spaces or newlines.
#
# Supported Compression Methods:
# zlib@openssh.com
# zlib
# none
compression = zlib@openssh.com,zlib,none

# Endpoint to listen on for incoming SSH connections.
# See https://twistedmatrix.com/documents/current/core/howto/endpoints.html#servers
# (default: listen_endpoints = tcp:2222:interface=0.0.0.0)
# (use systemd: endpoint for systemd activation)
# listen_endpoints = systemd:domain=INET:index=0
# For both IPv4 and IPv6: listen_endpoints = tcp6:2222:interface=\:\:
# Listening on multiple endpoints is supported with a single space separator
# e.g listen_endpoints = "tcp:2222:interface=0.0.0.0 tcp:1022:interface=0.0.0.0" will result listening
# use authbind for port numbers under 1024

listen_endpoints = tcp:2222:interface=0.0.0.0

# Enable the SFTP subsystem
# (default: true)
sftp_enabled = true


# Enable SSH direct-tcpip forwarding
# (default: true)
forwarding = true


# This enables redirecting forwarding requests to another address
# Useful for forwarding protocols to other honeypots
# (default: false)
forward_redirect = false
```

Lampiran - 16 Isi *file* konfigurasi *cowrie.cfg.dist* (lanjutan)

```
forward_redirect = false


# Configure where to forward the data to.
# forward_redirect_<portnumber> = <redirect ip>:<redirect port>

# Redirect http/https
# forward_redirect_80 = 127.0.0.1:8000
# forward_redirect_443 = 127.0.0.1:8443

# To record SMTP traffic, install an SMTP honeypoint.
# (e.g https://github.com/awhitehatter/mailoney), run
# python mailoney.py -s yahoo.com -t schizo_open_relay -p 12525
# forward_redirect_25 = 127.0.0.1:12525
# forward_redirect_587 = 127.0.0.1:12525


# This enables tunneling forwarding requests to another address
# Useful for forwarding protocols to a proxy like Squid
# (default: false)
forward_tunnel = false


# Configure where to tunnel the data to.
# forward_tunnel_<portnumber> = <tunnel ip>:<tunnel port>

# Tunnel http/https
# forward_tunnel_80 = 127.0.0.1:3128
# forward_tunnel_443 = 127.0.0.1:3128


# No authentication checking at all
# enabling 'auth_none' will enable the ssh2 'auth_none' authentication method
# this allows the requested user in without any verification at all
#
```

Lampiran - 17 Isi *file* konfigurasi *cowrie.cfg*

```
[ssh]
enabled = true
listen_endpoints = tcp:22:interface=0.0.0.0
```

Lampiran - 18 Isi *file* konfigurasi *bin/cowrie*

```
DEFAULT_VIRTUAL_ENV=cowrie-env

first_time_use() {
    echo
    echo "Join the Cowrie community at: https://www.cowrie.org/slack/"
    echo
}

python_version_warning() {
    if python -V 2>&1 | grep -q  '^Python 2.'; then
        echo
        echo "DEPRECATION: Python 2.7 reached the end of its life on January 1st, 2020."
        echo "Cowrie has dropped support for Python 2.7."
        echo
    fi
    if python -V 2>&1 | grep -q  '^Python 3.[5678]'; then
        echo
        echo "DEPRECATION: Python<3.9 is no longer supported by Cowrie."
        echo
    fi
}

find_cowrie_directory() {
    # Determine Cowrie directory
    if [[ "$0" = /* ]]
    then
        COWRIEDIR=$(dirname $0)/..
    else
        COWRIEDIR=$(dirname $PWD/$0)/..
    fi
    COWRIEDIR=$(cd ${COWRIEDIR} && pwd -P 2>/dev/null || pwd)
}

activate_venv() {
    # Activate Python virtual environment
    VENV="$1"
```

Lampiran - 19 Isi *file* konfigurasi *bin/cowrie* (lanjutan)

```
activate_venv() {
    # Activate Python virtual environment
    VENV="$1"
    if [ ! -f "$VENV/bin/activate" ]
    then
        return 1
    fi
    . $VENV/bin/activate
    return 0
}

cowrie_status() {
    # Print status
    PID=$(cat ${PIDFILE} 2>/dev/null || echo "")
    if [ -n "$PID" ]; then
        if ps -e -o pid | grep -e "^\W*$PID$" 2>&1 >/dev/null; then
            echo "cowrie is running (PID: ${PID})."
        else
            echo "cowrie is not running (PID: ${PID})."
            echo "Removing stale PID file ${PIDFILE}"
            rm -f ${PIDFILE}
        fi
    else
        echo "cowrie is not running."
    fi
}

cowrie_start() {
    # Start Cowrie
    COWRIEARGS="$*"
    TWISTEDARGS="${XARGS} --umask=0022 --pidfile=${PIDFILE}"

    # Run foreground or background. Foreground has no file log.
    if [ "$COWRIE_STDOUT" = "yes" ]; then
        TWISTEDARGS="${TWISTEDARGS} -n -l -"
    else
```

Lampiran - 20 Isi *file* konfigurasi *bin/cowrie* (lanjutan)

```
fi

# 1. Check if any virtual environment is active
# 2. Try COWRIE_VIRTUAL_ENV if defined
# 3. Try DEFAULT_VIRTUAL_ENV
# 4. Try ../DEFAULT_VIRTUAL_ENV
# 5. Try without virtual environment

if [ ! -z "$VIRTUAL_ENV" ]; then
    echo 2>&1 "Using activated Python virtual environment \"$VIRTUAL_ENV\""
elif activate_venv "$COWRIE_VIRTUAL_ENV"; then
    echo 2>&1 "Using custom Python virtual environment \"$VIRTUAL_ENV\""
elif activate_venv "$DEFAULT_VIRTUAL_ENV"; then
    echo 2>&1 "Using default Python virtual environment \"$VIRTUAL_ENV\""
# Look one directory higher for the virtual env to not pollute the Cowrie dir
elif activate_venv "../$DEFAULT_VIRTUAL_ENV"; then
    echo 2>&1 "Using default Python virtual environment \"../$VIRTUAL_ENV\""
else
    echo 2>&1 "Not using Python virtual environment"
fi

python_version_warning

# Automatically check if the authbind is enabled or not
authfile="/etc/authbind/byport/22"
if [ -z ${AUTHBIND_ENABLED} ] && [ -x "$authfile" ] && command -v authbind >/dev/null; then
    AUTHBIND_ENABLED=yes
else
    AUTHBIND_ENABLED=no
fi

echo "Starting cowrie: [twistd ${TWISTEDARGS} cowrie ${COWRIEARGS}]..."
if [ "$AUTHBIND_ENABLED" = "no" ]
then
    exec twistd ${TWISTEDARGS} ${COWRIEARGS} cowrie
else
```

Lampiran - 21 Isi *file* konfigurasi *bin/cowrie* (lanjutan)

```
    echo "Starting cowrie: [twistd ${TWISTEDARGS} cowrie ${COWRIEARGS}]..."
    if [ "$AUTHBIND_ENABLED" = "no" ]
    then
        exec twistd ${TWISTEDARGS} ${COWRIEARGS} cowrie
    else
        exec authbind --deep twistd ${TWISTEDARGS} ${COWRIEARGS} cowrie
    fi
}

cowrie_stop () {
    # Stop Cowrie
    PID=$(cat ${PIDFILE} 2>/dev/null || echo "")
    if [ -n "$PID" ]; then
      echo "Stopping cowrie..."
      if kill -TERM $PID; then
          echo -n
      else
          echo "Removing stale PID file ${PIDFILE}"
          rm -f ${PIDFILE}
      fi
    else
        echo "cowrie is not running."
    fi
}

cowrie_force_stop () {
    # Force Stop Cowrie
    PID=$(cat ${PIDFILE} 2>/dev/null || echo -n "")
    if [ -n "$PID" ]; then
        echo -n "Stopping cowrie..."
        if kill -TERM $PID; then
            ((t = 60))
            while ((t > 1)); do
                sleep 1
                echo -n .
                if kill -0 $PID 2>/dev/null; then
```

Lampiran - 22 Isi *file* konfigurasi *bin/cowrie* (lanjutan)

```
                if kill -0 $PID 2>/dev/null; then
                    ((t -= 1))
                else
                    echo "terminated."
                    return
                fi
            done
            kill -KILL $PID
            echo "killed."
        else
            echo "Removing stale PID file ${PIDFILE}"
            rm -f ${PIDFILE}
        fi
    else
        echo "cowrie is not running."
    fi
}

cowrie_usage() {
    echo "usage: $0 <start|stop|force-stop|restart|status|shell>"
}

# Mostly for Docker use, to quickly get a shell in the container
cowrie_shell() {
    $SHELL
}

###############################################################################
## Main script
###############################################################################

if [ "$#" = 0 ]
then
    cowrie_usage
    exit 1
fi
```

Lampiran - 23 Isi *file* konfigurasi *bin/cowrie* (lanjutan)

```
find_cowrie_directory $0
cd ${COWRIEDIR}
export PYTHONPATH=${PYTHONPATH}:${COWRIEDIR}/src

set -e

# Don't store pidfile on Docker persistent volume
if [ "${COWRIE_STDOUT}" = "yes" ]; then
        PIDFILE=""
else
        PIDFILE=var/run/cowrie.pid
fi

if [ ! -f ${COWRIEDIR}/var/log/cowrie/cowrie.log ]
then
    first_time_use
fi

key=$1
shift 1
case $key in
    stop)
        cowrie_stop $*
        ;;
    force-stop)
        cowrie_force_stop $*
        ;;
    start)
        cowrie_start $*
        ;;
    restart)
        cowrie_stop $*
        cowrie_start $*
        ;;
    status)
```

Lampiran - 24 Isi *file* konfigurasi *bin/cowrie* (lanjutan)

```
    ;;
status)
    cowrie_status $*
    ;;
bash)
    cowrie_shell $*
    ;;
sh)
    cowrie_shell $*
    ;;
shell)
    cowrie_shell $*
    ;;
*)
    cowrie_usage
    exit 1
    ;;
esac
```