



ANALISIS KERENTANAN APACHE LOG4J PADA CVE-2021-44228 TERHADAP ANCAMAN REMOTE ACCESS TROJAN DENGAN METODE PENETRATION TESTING EXECUTION STANDARD

LAPORAN SKRIPSI

MUHAMMAD NUR IRSYAD

1807422020

**PROGRAM STUDI TEKNIK MULTIMEDIA DAN JARINGAN
JURUSAN TEKNIK INFORMATIKA DAN KOMPUTER
POLITEKNIK NEGERI JAKARTA
2022**



ANALISIS KERENTANAN APACHE LOG4J PADA CVE-2021-44228 TERHADAP ANCAMAN REMOTE ACCESS TROJAN DENGAN METODE PENETRATION TESTING EXECUTION STANDARD

LAPORAN SKRIPSI

**Dibuat untuk Melengkapi Syarat-Syarat yang Diperlukan
untuk Memperoleh Gelar Sarjana Terapan Politeknik**

**MUHAMMAD NUR IRSYAD
1807422020**

**PROGRAM STUDI TEKNIK MULTIMEDIA DAN JARINGAN
JURUSAN TEKNIK INFORMATIKA DAN KOMPUTER
POLITEKNIK NEGERI JAKARTA
2022**

SURAT PERNYATAAN BEBAS PLAGIARISME

Saya yang bertanda tangan di bawah ini:

Nama : Muhammad Nur Irsyad
NIM : 1807422020
Jurusan : TIK - Teknik Informatika dan Komputer
Program Studi : TMJ - Teknik Multimedia dan Jaringan
Judul Skripsi : Analisis Kerentanan Apache Log4j pada CVE-2021-44228 terhadap Ancaman Remote Access Trojan dengan Metode Penetration Testing Execution Standard

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, __ ____ 2022
Yang membuat pernyataan,

Muhammad Nur Irsyad
NIM. 1807422020

LEMBAR PENGESAHAN

Skripsi diajukan oleh:

Nama : Muhammad Nur Irsyad
NIM : 1807422020
Program Studi : TMJ - Teknik Multimedia dan Jaringan
Judul Skripsi : Analisis Kerentanan Apache Log4j pada CVE-2021-44228 terhadap Ancaman Remote Access Trojan dengan Metode Penetration Testing Execution Standard

Telah diuji oleh tim penguji dalam Sidang Skripsi pada hari __, tanggal __, bulam ____, tahun __, dan dinyatakan **LULUS**.

Disahkan oleh:

Pembimbing I : Ariawan Andi Suhandana, S.Kom., M.T.I. (.....)
Penguji I : Defiana Arnaldy, S.Tp., M.Si. (.....)
Penguji II : Fachroni Arbi Murad, S.Kom., M.Kom. (.....)
Penguji III : Asep Kurniawan, S.Pd., M.Kom. (.....)

Mengetahui:

Jurusan Teknik Informatika dan Komputer
Ketua

Mauldy Laya , S.Kom., M.Kom.
NIP. 197802112009121003

KATA PENGANTAR

aaa

Depok, __ ____ 2022

Muhammad Nur Irsyad

**SURAT PERNYATAAN PERSETUJUAN PUBLIKASI
SKRIPSI UNTUK KEPENTINGAN AKADEMIS**

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

Nama : Muhammad Nur Irsyad
NIM : 1807422020
Jurusan : TIK - Teknik Informatika dan Komputer
Program Studi : TMJ - Teknik Multimedia dan Jaringan

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

Analisis Kerentanan Apache Log4j pada CVE-2021-44228 terhadap Ancaman
Remote Access Trojan dengan Metode Penetration Testing Execution Standard

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 pernyataan ini saya buat dengan sebenarnya.

Depok, __ ____ 2022

Yang menyatakan,

Muhammad Nur Irsyad
NIM. 1807422020

ABSTRAK

aaa

Kata Kunci: aaa

DAFTAR ISI

HALAMAN SAMPUL

HALAMAN JUDUL

SURAT PERNYATAAN BEBAS PLAGIARISME.....3

LEMBAR PENGESAHAN.....4

KATA PENGANTAR.....5

SKRIPSI UNTUK KEPENTINGAN AKADEMIS.....6

ABSTRAK.....7

DAFTAR ISI.....8

PENDAHULUAN.....11

1.1 Latar Belakang.....11

1.2 Rumusan Masalah.....13

1.3 Batasan Masalah.....14

1.4 Tujuan dan Manfaat.....14

1.5 Sistematika Penulisan.....16

TINJAUAN PUSTAKA.....17

2.1 Remote Access Trojan (rangkum).....17

2.1.1 Konektivitas RAT terhadap Firewall dan Sistem (rangkum).....17

2.1.2 Reverse & Bind Shell TCP (rangkum).....18

2.2 Apache Log4j (rangkum).....19

2.2.1 Lightweight Directory Access Protocol (rangkum).....20

2.3 Penetration Testing Execution Standard (rangkum).....22

2.3.1 Stakeholder-Specific Vulnerability Categorization (rangkum).....23

2.3.2 Attack Trees (rangkum).....26

2.4 Unified Modelling Language (padet).....27

2.4.1 Class Diagram.....27

2.4.2 Activity Diagram.....27

2.5 Software Testing.....28

2.5.1 Integration Testing.....28

2.5.2 Alpha Testing.....28

2.6 Statistical Testing.....28

2.6.1 Cochran's Q Test.....28

2.6.1	Chi-Square Table.....	28
2.7	Penelitian Sejenis (update referensi).....	28
PERANCANGAN DAN REALISASI.....		30
3.1	Perancangan Sistem.....	30
3.1.1	Deskripsi Sistem (merging ke 3,1).....	30
3.1.2	Desain Topologi Jaringan.....	30
3.1.3	Desain Skema LDAP.....	32
3.1.4	Spesifikasi Sistem (docker container inspect [name]).....	36
3.2	Realisasi Sistem.....	37
3.2.1	Implementasi Sistem Pengguna.....	37
3.2.1.1	Instalasi dan Konfigurasi OpenLDAP Server.....	37
3.2.1.2	Pengembangan Aplikasi GUI Desktop LDAP Client.....	37
3.2.2	Implementasi Sistem Penyerang.....	37
3.2.2.1	Instalasi dan Konfigurasi OpenLDAP Server.....	37
3.2.2.2	Instalasi dan Konfigurasi Apache HTTP Server.....	38
3.2.2.3	Pengembangan Aplikasi Java HTTP Server.....	38
3.2.2.4	Pengembangan Java Payload.....	38
3.3	Skenario Pengujian.....	38
3.4	Skenario Analisis.....	38
HASIL DAN PEMBAHASAN.....		39
4.1	Pengujian.....	39
4.1.1	Deskripsi Pengujian.....	39
4.1.2	Prosedur Pengujian Aplikasi.....	39
4.1.2.1	Integration Testing.....	39
4.1.2.2	Alpha Testing.....	39
4.1.3	Prosedur Pengujian Kerentanan Sistem.....	39
4.1.3.1	Pre-Engagement.....	39
4.1.3.2	Intelligence Gathering.....	39
4.1.3.3	Threat Modelling.....	40
4.1.3.4	Vulnerability Analysis.....	40
4.1.3.5	Exploitation.....	40
4.1.3.6	Post-Exploitation.....	40
4.1.3.7	Mitigation.....	40

4.1.3.8	Post-Mitigation Exploitation.....	40
4.2	Data Hasil Pengujian.....	40
4.2.1	Hasil Data Pengujian Spesifikasi Aplikasi.....	40
4.2.2	Hasil Data Success Rate Pengujian Kerentanan Sistem.....	40
4.3	Analisis Data.....	40
4.3.1	Analisis Data Pengujian Spesifikasi Aplikasi.....	40
4.3.2	Analisis Data Success Rate Pengujian Kerentanan Sistem.....	40
PENUTUP.....		41
5.1	Kesimpulan.....	41
5.1	Saran.....	41
DAFTAR PUSTAKA.....		42
DAFTAR RIWAYAT HIDUP PENULIS.....		45

BAB I

PENDAHULUAN

1.1 Latar Belakang

Selama dua dekade lebih, kehadiran internet menjadi salah satu faktor penting dalam terintegrasinya komunikasi secara global. Setiap tindakan bisnis, ekonomi, dan sosial dari berbagai macam tingkatan lapisan masyarakat mayoritas sudah dilakukan dalam lingkup siber ini. Dengan adanya faktor akses internet yang mudah, konsep anonimitas, serta banyaknya aktor siber yang semakin mudah terhubung antara satu sama lain, seperti organisasi pemerintahan, aktivis siber, hingga penjahat siber, maka hadirilah juga tantangan di dalamnya yang dikenal sebagai ancaman siber (Li and Liu, 2021). **Potensi** serangan dan ancaman siber ini muncul dikarenakan adanya celah kerentanan terhadap sistem maupun infrastruktur, baik itu dikarenakan oleh kesalahan manusia maupun program dalam logika bisnisnya. Hal tersebut yang kemudian membolehkan sistem untuk diserang melalui berbagai macam perantara yang sesuai dengan bentuk celahnya. Masalah kerentanan ini yang lalu dieksploitasi oleh penyerang dengan landasan untuk manfaat pribadi dan berbagai macam faktor lainnya (Calín *et al.*, 2020).

Pada penerapannya, kerentanan-kerentanan yang telah terekspos tersebut akan diklasifikasi dan dievaluasi oleh para praktisi keamanan sistem. Kegiatan ini dilakukan dalam rangka proses identifikasi kerentanan secara resmi untuk nantinya direkam kedalam National Vulnerability Database (NVD) menggunakan referensi Common Vulnerability Exposure (CVE). Untuk memberikan gambaran mengenai besar pengaruh kerentanan terhadap suatu sistem, CVE memanfaatkan suatu standar yaitu Common Vulnerability Scoring System (CVSS). CVSS umum digunakan dalam mengukur dan menganalisa potensi tingkat kerusakan serta dampak dari kerentanan dan ancaman terhadap aset-aset yang dilindungi (Dobrovoljc, Trček and Likar, 2017; Ruohonen, 2019).

Dalam keamanan siber, bentuk aset yang perlu diamankan dapat berupa komputer atau perangkat keras, informasi internal, jaringan, hingga perangkat lunak dari penyalahgunaan otorisasi dari proses bisnis yang seharusnya. Dikarenakan ancaman

siber menjadi salah satu isu penting yang selalu berkembang pada skala global, maka hal ini mendorong pula untuk berkembangnya pertahanan pada suatu aset serta kebijakan bisnis perusahaan, terutama dalam mencegah kebocoran data internal (Alghamdi, Cordeiro and Barbosa, 2021). **Salah** satu penyebab terjadinya kebocoran data adalah adanya kerentanan sistem terhadap serangan malware yang sudah tertanam ke dalam sistem korban, baik dengan mendownload file malware ataupun menerima phishing email. Hal ini yang kemudian membuat penyerang dapat mengontrol sistem korban secara jarak jauh untuk mengambil aset dan informasi digital secara transparan terhadap supervisi sistem korban (Yin and Khine, 2019).

Salah satu kasus ancaman siber yang muncul pada akhir **November 2021** dengan penyebab yang serupa adalah ancaman Log4Shell, yaitu istilah untuk kerentanan terhadap produk Apache Log4j terhadap serangan remote shell. Hal ini dikonfirmasi oleh Oracle dalam publikasinya pada 10 Desember 2021, yang menjelaskan bahwa kerentanan dengan referensi CVE-2021-44228 tersebut **menyebabkan** penyerang untuk dapat mengontrol sistem korban melalui penyalahgunaan dalam fitur *logging*. Langkah awal ini yang kemudian digunakan untuk mengunduh dan menjalankan *payload malware* yang dirancang dalam bahasa pemrograman Java. **Adanya** eksekusi *arbitrary code* tersebut yang nantinya akan membangun koneksi remote, baik secara *reverse shell* maupun *bind shell*; secara penuh mengontrol sistem korban tanpa ada autentikasi diantaranya (Khan and Neha, 2016; Apache, 2021; CVE, 2021; Oracle, 2021). **Berdasarkan** dampaknya, kerentanan tersebut memberikan resiko yang tinggi terhadap perusahaan secara global, seperti Adobe Creative Cloud, Cisco, IBM, hingga Amazon Web Service (AWS). Hal ini dikarenakan teknologi *open-source* tersebut digunakan dalam berbagai *platform*, seperti cloud service, web service dan software development, sehingga ada potensi besar pada pengembangan kerentanan tersebut untuk memaksimalkan serangannya (Nath, 2022).

Selain itu, kerentanan CVE-2021-44228 juga merupakan satu-satunya kerentanan Log4j yang memiliki nilai CVSS tertinggi, yaitu 10.0 dengan status severity critical. Hal yang membuatnya berbeda dari kerentanan Log4j lainnya adalah kerentanan tersebut menjadikannya sebagai pelopor untuk 3 kerentanan Log4j yang baru, yang mana hasil pengembangannya hanya berdurasi kurang dari tiga minggu (26/11/2021

– 11/12/2021). Hal ini berbeda dengan 2 kasus kerentanan sebelum adanya Log4Shell ini, yang mana memiliki periode interval hingga 4 tahun lebih (29/01/2017 – 26/11/2021). Maka dari itu, pengembangan kerentanan pada CVE-2021-44228 tergolong cepat dikarenakan fleksibilitas dan media serangannya yang luas (Apache, 2021).

Dalam rangka mendalami kerentanan tersebut, peneliti berupaya untuk menganalisa ancaman Apache Log4j pada referensi CVE-2021-44228 terhadap pengembangan eksploitasinya. Hal ini dilakukan dengan adanya pengujian pada post exploitation menggunakan ancaman *remote access trojan* dalam bentuk *persistence backdoor*. Keseluruhan tahapan pengujian akan berbasiskan pada model *Penetration Testing Execution Standard* (PTES) untuk mendapatkan hasil yang akurat pada suatu lingkup panduan pengujian (Dalalana and Zorzo, 2017). Implementasi pengujian dilakukan dalam serangan *remote code execution* (RCE) dengan memanfaatkan URL *entry manipulation* secara *hands-on-keyboard* yang didasari pada miskonfigurasi aplikasi serta lemahnya validasi request input pengguna (Biswas *et al.*, 2018). Pengujian kemudian dikembangkan dengan menyisipkan *backdoor* kedalam sistem target untuk dapat mempertahankan akses yang sudah didapatkan. Adapun mitigasi yang diadaptasikan merujuk kepada pendekatan *static analysis* untuk mendeteksi source code dan algoritma program yang menjadi sumber celah kerentanan, serta pemanfaatan *system utility* layaknya program dan port monitoring tools, dalam meminimalisir ancaman tersebut. Hal tersebut kemudian akan dijadikan sebagai tolak ukur untuk dianalisis mengenai seberapa luas dan besarnya tingkat keberhasilan mitigasi tersebut (CEH, 2013; Muñoz and Mirosh, 2016; Kaushik *et al.*, 2021).

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah yang telah dipaparkan di atas, maka rumusan masalah dalam penelitian dapat dijabarkan sebagai berikut:

1. Bagaimana perancangan infrastruktur instrumen dengan integrasi Log4j dalam lingkup white box testing?
2. Bagaimana pendekatan mitigasi terhadap tingkat keberhasilan pengujian ancaman *remote access trojan* pada kerentanan Log4j?

1.3 Batasan Masalah

Adanya pembatasan suatu masalah digunakan untuk menghindari potensi pelebaran pokok masalah dari lingkup yang seharusnya, sehingga dapat membuat penelitian lebih terarah untuk tercapainya tujuan dari penelitian ini. Beberapa batasan masalah dalam penelitian ini kemudian dijabarkan sebagai berikut:

1. Batasan dalam perancangan infrastruktur instrumen
 - a. Demonstrasi dilakukan dalam lingkup virtual dengan rancangan arsitektur client-server secara lokal (*ldap-server*, *GUI desktop ldap-client*, *attacker machine*) pada platform Linux
 - b. Pengujian kerentanan dilakukan dengan pendekatan white box testing, dan hanya ditujukan untuk *library* Log4j dalam versi 2.0-beta9 hingga 2.14.1 (Java 8) yang sesuai dengan referensi CVE-2021-44228, sehingga tidak mencakup *security patch* terakhir, yaitu versi 2.17.1 per tanggal 28/12/2021
2. Batasan dalam implementasi mitigasi dan pengujiannya
 - a. Bentuk mitigasi mencakupi pendekatan *static analysis* serta pemanfaatan *system utility*, tidak mencakup bentuk pengaplikasian *system update* ataupun *security patch*
 - b. Proses pengujian dilakukan dalam 2 tahap, yaitu pra dan pasca adanya mitigasi, sehingga tergambarinya pencapaian yang dapat dianalisa besar tingkat keberhasilannya

1.4 Tujuan dan Manfaat

Berdasarkan rumusan masalah dalam sub-bab sebelumnya, maka adapun tujuan serta manfaat yang ingin dicapai dalam pembentukan penelitian ini. Tujuan penelitian dijabarkan sebagai berikut:

1. Membangun instrumen penelitian, yang mencakup keseluruhan infrastruktur baik untuk sisi penyerang dan sisi korban
2. Mampu melakukan keseluruhan proses pengujian dalam white box testing yang spesifik terhadap kerentanan pada versi Log4j pada batasan masalah, sehingga lingkup objek penelitian tidak melebar dari yang ditunjukkan

3. Mampu **mengimplementasikan** pendekatan mitigasi yang sesuai dengan teknologi dan platform infrastruktur yang diujikan
4. Menyajikan data hasil pengujian yang dapat dianalisis terkait dengan **tingkat keberhasilan** mitigasi dan dampak pengembangan ancamannya

Berdasarkan tujuan penelitian yang hendak dicapai, diharapkan pula penelitian ini mempunyai manfaat dalam sisi pendidikan, teknologi, serta keamanan dalam general, baik secara langsung maupun tidak langsung. Adapun manfaat penelitian yang dijabarkan sebagai berikut:

1. Manfaat teoretis

- a. Memberikan sumbangan pemikiran pada analisis keamanan siber, baik dari sisi pendekatan serangan serta mitigasinya terhadap kerentanan Log4j dan ancaman pengembangan *remote access trojan*
- b. Sebagai pijakan referensi untuk penelitian sejenis terkait pada kerentanan Log4j kedepannya, serta sebagai kajian lebih lanjut dalam analisis ancaman *remote access trojan* terhadap teknologi Log4j dan tingkat keefektifan terhadap mitigasinya

2. Manfaat praktis

- a. Bagi pembaca dan masyarakat
Penelitian ini dapat dijadikan sebagai salah satu penggambaran terhadap besarnya dampak kerusakan terhadap kerentanan Log4j yang diharapkan dapat memberikan kewaspadaan pada pemakaian aplikasi yang kiranya memiliki kerentanan serupa
- b. Bagi pengembang aplikasi
Penelitian ini dapat membantu memaparkan seberapa jauh eksploitasi kerentanan Log4j dan seberapa efektif pendekatan mitigasinya yang sifatnya temporer dan lebih preventif tersebut
- c. Bagi perusahaan
Penelitian ini dapat dijadikan sebagai contoh kasus penjabaran dari dampak kerentanan Log4j terhadap sisi pengguna produk perusahaan yang rentan, sehingga diharapkan bisa memberikan suatu

pertimbangan dan kesadaran mengenai pentingnya keamanan dalam sisi infrastruktur

1.5 Sistematika Penulisan

Sistematika penulisan berikut dibentuk untuk mempermudah dalam penyusunan proposal penelitian ini dengan penulisan yang baik. Sistematika penulisan yang digunakan dijabarkan sebagai berikut:

BAB I PENDAHULUAN

Bab pendahuluan mendeskripsikan mengenai latar belakang serta bagaimana urgensi masalah, perumusan masalah, menentukan batasan-batasan masalah, mendefinisikan tujuan dan manfaat penelitian, serta sistematika struktur penulisan dalam merancang laporan penelitian ini

BAB II TINJAUAN PUSTAKA

Bab tinjauan pustaka berisikan seluruh teori-teori landasan yang digunakan dalam inti pembahasan pada rancangan penelitian dari berbagai sumber yang kredibel. Adapun penjabaran terkait penelitian sejenis yang digunakan sebagai penunjang dan pengembangan dari penelitian yang sebelumnya dalam kurun waktu 10 tahun terakhir

BAB III PERENCANAAN DAN REALISASI

Bab perencanaan dan realisasi menjelaskan tahapan-tahapan yang dilakukan dalam membangun instrumen penelitian dan bagaimana proses pengujian kerentanan dapat berlangsung terhadap objek penelitian menggunakan metodologi ataupun framework yang sudah ditentukan

BAB IV PEMBAHASAN

Bab pembahasan memaparkan bagaimana data yang didapatkan dari hasil pengujian untuk dianalisa menggunakan pendekatan statistika. Metrik penilaian bersifat kualitatif dalam bentuk katagorikal, yaitu dalam bentuk pencapaian ataupun milestone terhadap setiap parameter-parameter dalam pengujian

BAB V PENUTUP

Bab penutup menjelaskan mengenai pembuktian terhadap tujuan yang ingin dicapai dalam penelitian dan bagaimana hasil penelitiannya. Adapun saran yang diberikan terkait dengan hasil pengujian yang sifatnya konstruktif

BAB II

TINJAUAN PUSTAKA

2.1 Remote Access Trojan (rangkum)

Konsep dari istilah trojan adalah bagaimana suatu ancaman atau serangan malware dapat dikemas sedemikian rupa sehingga adanya kesan transparansi saat serangan dijalankan kedalam sistem korban. Hal ini ditunjukkan agar keseluruhan serangan tetap bersifat false negative terhadap sistem keamanan korban. *Payload trojan* dapat dikirim menggunakan berbagai pendekatan, seperti *phishing*, *adware*, ataupun dengan pendekatan social engineering. Pendekatan ini dimanfaatkan oleh penyerang untuk masuk ke dalam aset, data, serta *resource* korban untuk dikelola sepenuhnya secara tak terbatas. Pada Remote Access Trojan (RAT), trojan dispesifikasikan untuk mengontrol sistem korban sepenuhnya secara *remote*, yang mana memanfaatkan koneksi *berarsitektur client-server* di antara keduanya (CEH, 2013; Hama Saeed, 2020).

2.1.1 Konektivitas RAT terhadap Firewall dan Sistem (rangkum)

Terkait dengan transmisi data antar sistem korban dengan penyerang, salah satu faktor yang dapat menghambat komunikasi RAT adalah terdapatnya penggunaan *firewall*. Hal ini disebabkan karena adanya potensi *inbound rules* terhadap port tertentu yang diizinkan untuk berkomunikasi dari pihak luar masuk kedalam sistem. Apabila sistem korban serta penyerang terpisahkan oleh *firewall*, maka probabilitas akan lebih kecil untuk serangan RAT dapat berhasil. Maka dari itu, apabila penyerang dan korban berada dalam satu jaringan dibawah *firewall*, proses penyerangan akan dapat dilakukan dengan mudah, terlepas dari system firewall yang dimiliki bawaan pada sistem korban. Hal ini yang kemudian membuat *firewall* harus dapat dikendalikan ataupun dimodifikasi *rules* nya terlebih dahulu, agar pemasangan koneksi *backdoor* dapat menjadi lebih stabil (CEH, 2013).

Walaupun begitu, setiap koneksi yang terbuka maupun yang berhasil tersambung ke dalam sistem korban akan tetap dapat terlihat dalam network utilities. Kumpulan kategori perangkat lunak tersebut umum digunakan dalam mengaudit koneksi aktif yang berada dalam sistem. Hal ini yang kemudian mendorong serangan untuk dapat

berjalan satu langkah lebih transparan lagi, dengan salah satu pendekatannya adalah adanya penggunaan rootkit. Rootkit merupakan perangkat lunak yang ditujukan untuk menyembunyikan suatu kegiatan dalam sistem, sehingga dapat meningkatkan kontinuitas berjalannya suatu program dari faktor luar. Salah satu implementasinya yaitu dengan menyembunyikan konektivitas pada spesifik port tertentu lewat memanipulasi konten buffer dalam file system `/proc/net/tcp`. Pendekatan ini dapat menjadi esensial karena secara umum, file system tersebut digunakan secara langsung oleh network utilities dalam membaca koneksi protokol TCP yang sedang aktif di dalam sistem, yang kemudian diambil datanya sebagai bahan audit dalam program. Dengan begitu, port yang ditujukan dalam berlangsungnya serangan RAT diharapkan tetap dapat berjalan dibawah kontrol dan supervisi sistem dan firewall dalam pendekatan yang sederhananya (Junnila, 2020; Phillip, 2020).

2.1.2 Reverse & Bind Shell TCP (rangkum)

Penggunaan koneksi remote access dengan protokol TCP menjadi pilihan yang lebih dapat diandalkan dibandingkan protokol UDP karena sifatnya yang connection-oriented. Dalam membangun koneksi tersebut, adapun skenario yang sudah dianalisis terlebih dahulu karena keberhasilan serta stabilitas akan bergantung terhadap pendekatan dari pengiriman payload, serta topologi infrastruktur jaringannya (Maraj, Rogova and Jakupi, 2020). Secara umum, terdapat 2 bentuk payload yang dapat digunakan, yaitu secara reverse dan bind, dengan tujuan yang sama yaitu untuk mengontrol sistem korban melalui akses shell yang didapatkannya.

Dalam bind shell, inisiasi pembukaan koneksi *remote* dilakukan dalam sistem korban pada spesifik port tertentu. Hal ini dilanjutkan dengan sistem penyerang menyambung kedalam koneksi tersebut agar mendapatkan shell dari sistem korban. Berdasarkan perspektif penyerang, pendekatan ini diistilahkan sebagai kegiatan *bind* atau menyambungkan. Apabila koneksi *port* dalam sistem korban masih berstatus *listening*, maka esensinya seluruh sistem lain di jaringan juga memiliki kesempatan untuk mengontrol sistem tersebut dalam satu waktu. Dikarenakan koneksi dibukakan dari sistem korban, maka *port* yang digunakan haruslah sesuai dengan *inbound rules* dalam *firewall*, sehingga dibutuhkan pendekatan lebih untuk membuatnya lebih efisien, terlebih dalam jaringan yang tersegmentasi secara kompleks.

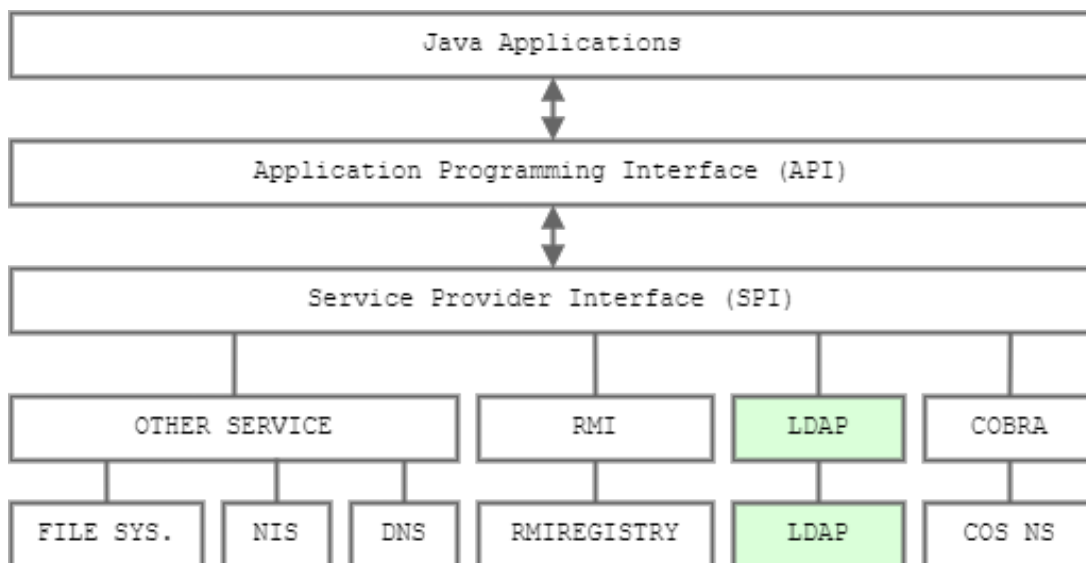
Berbeda dengan payload *bind shell*, penggunaan *reverse shell* membutuhkan sistem penyerang untuk menginisiasi koneksi port terlebih dahulu, yang kemudian membutuhkan sistem korban untuk melakukan *binding* terhadap koneksinya. Dikarenakan hal tersebut, pendekatan ini disebut juga sebagai *reverse* yang berarti disambungkan. Mekanisme ini juga memiliki kelebihan dibandingkan dengan penggunaan *bind shell*, yang salah satunya adalah tidak ada kekhawatiran terhadap ketersediaan port dalam sistem korban. Hal ini disebabkan karena pemilihan inisiasi port dilakukan oleh sistem penyerang, sehingga variabel tersebut menjadi sesuatu yang dapat dikontrol secara penuh. Selain itu, penggunaan *reverse shell* juga dapat meminimalisir dari keterbatasannya dalam menghadapi *firewall*, karena secara umum *outbound rules* tidak lebih restriktif dibandingkan dengan *inbound rules* pada konsep *bind shell* (Miller, 2019).

2.2 Apache Log4j (rangkum)

Apache Log4j, merupakan salah satu *framework logging* yang umum digunakan oleh berbagai aplikasi dan layanan secara global, seperti Minecraft Gaming Platform, Apple iCloud, serta Amazon Web Services (AWS) (Stanger, 2022). Adapun pengembangan signifikan yang dikeluarkan dalam versi 2 ini untuk mengganti rilis perdananya yang berakhir pada Agustus 2015 lalu. Berikut merupakan beberapa fitur pengembangan yang dihadirkan dalam Log4j 2:

1. Log4j 2.x didesain untuk tidak menghilangkan *event logback* saat dilakukan rekonfigurasi dalam waktu yang bersamaan dengan memanfaatkan exception terhadap Appender nya
2. Log4j 2.x merupakan aplikasi *stand-alone* yang bersifat *garbage free*, sehingga meringankan peran *garbage collector* dan memberikan performa yang baik dalam *response time*-nya
3. Log4j 2.x mendukung untuk pemakaian *custom plugins*, sehingga mudah untuk mengembangkan fungsionalitas *framework* tersebut, seperti menambahkan fitur *Layouts*, *Filter*, *Pattern Converters*, hingga *Lookups* yang dapat diatur dalam file konfigurasinya

Dalam melakukan fungsi logging, Log4j juga dapat terintegrasi dengan berbagai macam layanan naming dan directory service untuk mengambil informasi data maupun objek di dalamnya. Hal ini dapat diraih melalui penggunaan Java Naming and Directory Interface (JNDI). JNDI inilah yang kemudian dapat membantu Log4j untuk melakukan pencarian, yang disebut juga sebagai fungsi lookup, terhadap layanan tersebut, baik itu tersedia dalam lingkup lokal maupun remote (Apache, 2022). Berikut pada gambar 2.1 merupakan arsitektur dari penggunaan JNDI terhadap aplikasi serta layanan yang didukungnya:



Gambar 2.1 Arsitektur JNDI

Sumber: Roy, 2015

Merujuk pada gambar 2.1, konsep arsitektur JNDI menggunakan 3 komponen utama, yaitu aplikasi Java, JNDI API dan JNDI Service Provider Interface (SPI). JNDI SPI digunakan untuk membuat tersedianya konektivitas layanan naming dan directory service terhadap aplikasi secara transparan dan efisien. Ketersediaan tersebut yang dimanfaatkan oleh aplikasi dalam mengakses informasi dan objek melalui fungsi-fungsi yang dimiliki JNDI API. Beberapa layanan yang didukung diantaranya adalah Common Object Service (COS) Name Service, Lightweight Directory Access Protocol (LDAP), serta Domain Name Service (DNS) (Roy, 2015).

2.2.1 Lightweight Directory Access Protocol (rangkum)

LDAP merupakan salah satu teknologi yang berperan penting pada infrastruktur komputasi dalam skala industri. Adapun fungsi utama layanan LDAP adalah untuk

menyimpan skema informasi pengguna baik untuk kebutuhan autentikasi seperti Single Sign-On (SSO), distribusi public key dalam kriptografi asimetrik, serta email routing. Hal ini berkaitan karena LDAP tidak mengedepankan kecepatan dalam menulis atau mengalter data yang umumnya bersifat konstan, namun membutuhkan performa tinggi dalam pembacaanya, sehingga disebut juga “write once, read many times”.

Layanan LDAP berarsitekturkan client-server serta berbasiskan struktural direktori dalam menyimpan informasi di dalamnya. Pada mekanisme penyimpanan datanya, LDAP menggunakan suatu object class yang merupakan suatu template dari data yang akan di-entri terhadap atribut yang dimilikinya, beberapa object class seperti Groups, Organizational Person, Device, serta Country. Object class juga dapat digunakan untuk pewarisan atau inheritance, sehingga object class dibawahnya memiliki atribut dari object class parent nya. Berikut pada tabel 2.1 merupakan contoh inheritance terhadap object class inetOrgperson (structural class) dari top (abstract class) dan beberapa atribut yang dimilikinya:

top (user) → person → inetUser → organizationalPerson → inetOrgPerson

Tabel 2.1 Keterangan atribut LDAP dalam object class inetOrgPerson

No.	Atribut	Deskripsi	Parent Class
1	uid	ID unik pengguna	top (user)
2	description	Deskripsi / informasi	person
3	telephoneNumber	Nomor telepon utama	person
4	inetUserStatus	Status keaktifan akun	inetUser
5	ou	Nama unit organisasi	organizationalPerson
6	title	Jabatan / posisi	organizationalPerson
7	postalCode	Kode pos pengiriman barang	organizationalPerson
8	mail	Alamat email pengguna	-
9	employeeNumber	Nomor / ID karyawan	-
10	labeledURI	Link asosiasi akun personal	-

Sumber: Oracle, 2010

Agar aplikasi dapat mengakses informasi dalam layanan LDAP, maka adapun konfigurasi yang disimpan dalam bentuk properti yang mana JNDI butuhkan, yaitu `javax.naming.Context.INITIAL_CONTEXT_FACTORY` dan `javax.nam`

`ing.Context.PROVIDER_URL`. Pada umumnya, kedua key tersebut akan dibaca dari file `.properties` dan disimpan ke dalam struktur data hash table untuk nantinya digunakan sebagai inisialisasi context, yang umum disebut juga sebagai environment variable (Roy, 2015; Helmke, Hudson and Hudson, 2019).

2.3 Penetration Testing Execution Standard (rangkum)

Penetration Testing Execution Standard (PTES) merupakan salah satu framework dalam menjalankan evaluasi keamanan yang dibangun pada 2009. Berbeda dari beberapa framework pengujian lainnya, seperti National Institute of Standards and Technology (NIST) Guideline dan Open-Source Security Testing Methodology Manual (OSSTMM), landasan awal PTES sudah ditunjukkan sebagai panduan pengujian kerentanan, sehingga lingkup bahasan sudah terfokus dan berkembang dalam konteks yang sama. Dari 6 metrik yang digunakan sebagai bahan perbandingan (coverage, flexibility, modelling, adaptation, planning & documentation), PTES unggul terhadap segi planning yang mana membutuhkan keseluruhan elemen untuk direncanakan secara baik sebelum dijalankannya pengujian. **Walaupun** terdapat penjelasan lengkap terkait keseluruhan panduan, PTES tidak memberikan suatu bentuk konkrit terhadap seperti apa contoh dokumentasi laporan akhir, sehingga di satu sisi pengguna dapat mengembangkan bagaimana penyusunan laporan yang sesuai berdasarkan struktural dari panduan tersebut (Dalalana and Zorzo, 2017).

Adapun model ini ditunjukkan untuk lingkup bisnis dan penyedia layanan keamanan sebagai panduan yang sesuai dengan standar bisnis dan industri secara komprehensif. PTES terdiri dari 7 tahapan dasar, yang mencakup mulai dari latar belakang inisiasi dan persiapan instrumen dalam melakukan pentest, hingga ke tahap akhir yaitu pelaporan, yang mana dijabarkan sebagai berikut:

1. Pre-Engagement: mendefinisikan lingkup serta instrumen pengujian. Fase ini mencakup penentuan estimasi pengerjaan, timeline, objek yang akan diuji, biaya yang dikeluarkan, hingga adanya kesepakatan atau terms of agreement terhadap kedua pihak apabila dibutuhkan
2. Intelligence Gathering: mengumpulkan kelengkapan informasi yang berkaitan dengan karakteristik objek yang akan diuji baik secara aktif

maupun pasif. Pendekatan pun disesuaikan dengan besarnya lingkup objek tersebut, baik itu sistem organisasi, sistem individu, atau dalam lingkup teknologi saja

3. Threat Modelling: menggambarkan bagaimana pengujian akan dilakukan terkait dengan perancangan bentuk dan jenis serangan. Fase ini yang kemudian disesuaikan dengan informasi dan bentuk arsitektur dari yang didapatkan dalam tahap sebelumnya
4. Vulnerability Analysis: menganalisis potensi celah kerentanan yang terdapat dalam lingkup objek pengujian. Adapun analisa yang dapat dilakukan salah satunya dengan melakukan riset terhadap kerentanan CVE pada spesifik teknologi dari vendor tertentu
5. Exploitation: melakukan eksploitasi terhadap objek target yang difokuskan terhadap goals yang telah dibuat sebelumnya. Besar keberhasilan dalam tahap ini dipengaruhi dari bagaimana analisa kerentanan dan pemodelan kerentanan dilakukan dengan tepat
6. Post-Exploitation: mengembangkan ataupun meningkatkan hasil eksploitasi baik itu dengan meningkatkan hak akses penyerang, menjaga stabilitas akses remote, serta melakukan manipulasi terhadap pertahanan sistem seperti firewall. Hal ini dilakukan dalam rangka untuk mengukur seberapa jauh dampak kerentanan terhadap sistem
7. Reporting: mendokumentasi seluruh tahapan dan hasil kegiatan secara struktural dan informatif. Adapun informasi seperti kontak yang bertanggung jawab serta aset pengujian yang digunakan untuk dapat dimasukkan sebagai pengantar laporan (PTES, 2021)

Berdasarkan keleluasan informasi yang tersedia, PTES dapat diimplementasikan dalam 3 bentuk strategi, yang salah satunya adalah white box testing. Dalam whitebox testing, penguji memiliki seluruh informasi yang dibutuhkan mengenai objek target sebelum pengujian berlangsung. Pendekatan ini ditujukan sebagai dasar untuk meningkatkan introspeksi, ketelitian, serta stabilitas. Dikarenakan hal tersebut, pendekatan white box testing umum dilakukan oleh pihak internal organisasi maupun individu yang merupakan pengembang aplikasi tersebut (Madhavi, 2016).

2.3.1 Stakeholder-Specific Vulnerability Categorization (rangkum)

Dalam memprioritaskan tindakan remediasi yang harus dilakukan terlebih dahulu, praktisi keamanan sistem umum menggunakan CVSS sebagai parameter prioritas kuantitatifnya. Adapun salah satu framework yang serupa sebagai pengembangan CVSS yaitu Stakeholder-Specific Vulnerability Categorization (SSVC). Salah satu bentuk pembaharuan yang diberikan SSVC adalah faktor prioritas dilakukan dengan pendekatan decision-tree, bukan technical-severity sebagai prinsip dasarnya. Pendekatan ini memberikan sifat adaptasi dan fleksibilitas yang kuat dibandingkan penilaian CVSS, yang mana bersifat statik dan tidak temporer terhadap konteks manajemen dan analisa dampak kerentanan. Hal ini mencangkup terhadap aktor dan lingkup yang berbeda, baik itu dilakukan oleh manajemen internal maupun eksternal.

Berbeda dengan CVSS yang menggunakan penilaian numerik, SSVC menggunakan analisis cost-benefit terhadap bagaimana tingkat urgensi pada remediasinya. Hal ini menitik-beratkan pada konteks kondisi dan situasi lingkungan terhadap dampak kerentanan dalam implementasi nyatanya, yang mana berperan penting dalam pengambilan keputusan secara utuh dan relevan (Burgee, 2021; Liska, 2021; Spring *et al.*, 2021). **Adapun** informasi terkait komponen SSVC yang dipaparkan, yaitu pada tabel 2.2 merupakan tingkat keputusan yang digunakan dalam skala prioritasnya serta pada tabel 2.3 merupakan metrik yang digunakan dalam pemilihan jalur pada decision-tree. Deskripsinya sebagai berikut:

Tabel 2.2 Skala prioritas keputusan SSVC

Skala		Deskripsi
	Defer	Kerentanan tidak membutuhkan perhatian eksternal diluar dari pihak manajemen kerentanan (MK). Prosedur tetap mengikuti perkembangan informasi dan melakukan analisis prioritas kembali apabila dibutuhkan
	Scheduled	Kerentanan dengan karakteristik tertentu yang membutuhkan pendekatan lebih untuk meninjau perkembangannya. Dapat dilakukannya maintenance secara reguler

	Out-of-Cycle	Kerentanan membutuhkan perhatian eksternal diluar pihak MK. Adanya potensi kerusakan mengharuskan keseluruhan pihak untuk mendapatkan publikasi segera, selagi memantau perkembangan dan bukti serangan terhadap kerentanan tersebut, atau proof-of-concept (PoC). Adapun pengembangan mitigasi yang mulai dilakukan dan mengaplikasikan security patch apabila tersedia
	Immediate	Kerentanan telah memiliki PoC yang cukup dan valid serta tersebarnya publikasi resmi terhadap kerentanan tersebut dengan tingkat potensi serangan yang besar yang nyata. Keseluruhan pihak harus menjalankan remediasi dan menunda kegiatan operasional apabila dibutuhkan

Sumber: Burgee, 2021; Spring et al., 2021

Tabel 2.3 Definisi metrik decision-tree SSVC

Metrik		Deskripsi
Exploitation	None	Tidak ada atau minimnya bukti terhadap status aktif eksploitasi, serta ketersediaan PoC mengenai dokumentasi tersebut
	PoC	Bukti PoC masih bersifat internal, adanya diskusi terhadap potensi eksploitasi, dan kerentanan mulai terekam ke dalam lingkup framework seperti ExploitDB dan Metasploit
	Active	Kerentanan telah terekspos secara publik dengan sumber yang kredibel terhadap bagaimana situasi dan implementasi yang nyatanya
Automatable	No	Kerentanan tidak mendukung eksploitasi yang bersifat kontinuitas, yang dapat dikarenakan celah tidak selalu terdeteksi atau terenumerasi setiap saat
	Yes	Kerentanan mendukung adanya eksploitasi secara kontinuitas, yang mana umum terjadi terhadap eksploitasi RCE ataupun command injection
Value Density	Diffuse	Sistem yang rentan memiliki resource yang sedikit dan tidak berperan vital, salah satunya adalah mesin komputasi pengguna

	Concentrated	Sistem yang rentan memiliki resource yang luas dengan peran yang vital, seperti server
Utility	Laborious	Tidak dapat diautomasi dan bernilai diffuse
	Efficient	Dapat diautomasi dan bernilai diffuse, atau tidak dapat diautomasi dan bernilai concentrated
	S. Effective	Dapat diautomasi dan bernilai concentrated
Technical Impact	Partial	Eksplorasi memberikan akses kontrol yang terbatas atau memiliki potensi yang rendah untuk mendapatkan kompromisasi secara penuh
	Total	Eksplorasi memberikan suatu akses kontrol secara penuh terhadap resource target
Public-Safety Impact	Minimal	Dampak exploit bersifat minor terhadap client atau lingkup yang luas
	Significant	Dampak exploit yang bersifat masif dan bersifat bencana untuk berjalannya bisnis

Sumber: Spring et al., 2021

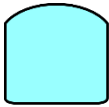



2.3.2 Attack Trees (rangkum)

Dalam tahap threat modelling, kini sudah tersedia berbagai macam metodologi yang dapat digunakan dalam menganalisa potensi serangan yang dapat terjadi terhadap suatu kerentanan. Dari beberapa bentuk metodologi yang ada, salah satu bentuk yang menitikberatkan terhadap perspektif penyerang (attacker-centric) adalah attack trees. Attack trees merupakan metodologi yang dibangun berdasarkan analisa serangan yang kemudian dilakukan estimasi terhadap bagaimana besar dampak risiko tersebut. Pada konsep dasarnya, terdapat 3 kondisi yang harus terpenuhi terlebih dahulu agar serangan dapat dilakukan dengan baik dan sukses terhadap keamanan sistem target, yaitu sebagai berikut:

1. Sistem target harus memiliki kerentanan untuk dapat dieksploitasi
2. Penyerang memiliki resource yang cukup dalam melaksanakan eksploitasi terhadap sistem target, yang istilahkan sebagai kapabilitas
3. Adanya dasar motivasi untuk mendapatkan keuntungan pada perspektif penyerang dalam melakukan eksploitasi yang berhasil

Seperti namanya, attack trees menggambarkan potensi vektor serangan dalam bentuk diagram pohon, yang mana dikerjakan dari bawah hingga mencapai ke puncak atas. Puncak attack trees diakari dengan satu tujuan utama atau root tree, yang dapat berisikan berbagai macam tujuan menengah atau subtree di dalamnya. Baik subtree maupun root tree dapat memiliki leaf node, yang merupakan vektor-vektor serangan untuk merealisasikan suatu tujuan tersebut. Adapun relasi antar subtree yaitu AND (dependen) dan OR (independen), yang menggambarkan apakah keberhasilan suatu subtree memerlukan seluruh tujuan dalam subtree dibawahnya untuk tercapai atau tidak (Shevchenko *et al.*, 2018; Ingoldsby, 2021). Untuk memahami komponen attack trees, berikut merupakan keterangan simbol-simbol yang digunakan dalam menggambarkan keseluruhan strukturnya:

Tabel 2.4 Deskripsi simbol attack trees

Simbol	Deskripsi
	Simbol node AND; dibutuhkan dua atau lebih subtree yang sukses untuk dapat mencapai atau melanjutkan tree yang ada di atasnya
	Simbol node OR; hanya membutuhkan salah satu subtree yang sukses untuk mencapai atau melanjutkan tree yang ada di atasnya
	Simbol leaf node; menggambarkan suatu vektor serangan yang bersifat standalone dan tidak dapat dijadikan sebagai subtree atau tree di atasnya
	Simbol line; menggambarkan relasi setiap komponen yang tersambung diantaranya

Sumber: Ingoldsby, 2021

2.4 Unified Modelling Language (padet)

2.4.1 Class Diagram

[jelasin logo / simbol class diagram]

[utk program gui client + java http server]

2.4.2 Activity Diagram

[jelasin logo / simbol activity diagram]

[utk program gui client + java http server]

2.5 Software Testing

2.5.1 Integration Testing

2.5.2 Alpha Testing

2.6 Statistical Testing

2.6.1 Cochran's Q Test

2.6.1 Chi-Square Table

2.7 Penelitian Sejenis (update referensi)

Penyusunan laporan ini mengambil dari beberapa referensi penelitian sebelumnya termasuk jurnal-jurnal yang berhubungan dengan penelitian ini untuk dikembangkan ataupun memberikan variabel baru, pemaparannya sebagai berikut:

Tabel 2.5 Perbandingan penelitian sejenis

Jurnal	Perbandingan Tinjauan Penelitian
<p>A novel approach to generate a reverse shell: Exploitation and Prevention (2021)</p> <p>—</p> <p>Kaushik et al.</p>	<p>Hasil penelitian menunjukkan pembuatan serta pencegahan <i>reverse shell</i> dalam <i>code injection</i>, <i>file upload</i>, serta <i>phishing</i> dalam beberapa bahasa pemrograman terhadap <i>php web-server</i></p> <p>—</p> <p>Peneliti memanfaatkan dan mengadaptasi seksi <i>code injection</i> tersebut untuk menjalankan <i>remote code execution</i> dalam <i>payload</i> Java terhadap <i>ldap-server</i> melalui <i>ldap-client</i> secara white box</p>
<p>Malware in Computer Systems: Problems and Solutions (2020)</p> <p>—</p> <p>Saaed, M</p>	<p>Hasil penelitian menunjukkan pencegahan <i>malware</i> dalam OS Windows menggunakan <i>Task Manager</i> terhadap <i>hidden / malicious process</i></p> <p>—</p> <p>Peneliti mempelajari dan menerapkan konsep mitigasi <i>malware</i> dalam OS Linux dengan beberapa <i>system utility</i> (<i>netstat</i>, <i>unhide</i>, <i>dll</i>) terhadap <i>hidden process</i>, khususnya pada trojan dan rootkit</p>

<p>Testing of network security systems through DoS, SQL injection, reverse TCP and social engineering attacks (2019)</p> <p>—</p> <p>Maraj, Rogova & Jakupi</p>	<p>Hasil penelitian menunjukkan proses implementasi <i>reverse TCP</i> dalam <i>virtualization</i> dengan instrumen pengujian berupa <i>web application</i> dan <i>external firewall</i>. <i>Payload</i> disebarkan melalui pendekatan <i>phishing email</i>, dengan <i>firewall</i> yang ter-<i>embedded</i> oleh <i>malware</i></p> <p>—</p> <p>Peneliti mengadaptasi <i>reverse TCP</i> dengan <i>environment testing</i> berupa <i>GUI-desktop application</i> dan <i>system firewall server</i>. <i>Payload</i> dan serangan disebarkan melalui pendekatan <i>hands-on-keyboard attack</i> dan <i>Bad USB</i></p>
---	--

Sumber: Hama Saeed, 2020; Maraj, Rogova and Jakupi, 2020; Kaushik et al., 2021

BAB III

PERANCANGAN DAN REALISASI

3.1 Perancangan Sistem

Pada perancangan sistem, akan dijelaskan mengenai bagaimana alur pembangunan keseluruhan sistem yang sesuai dengan desain topologi jaringan. Adapun desain skema terhadap penyimpanan data pada LDAP server yang sifatnya konseptual, sehingga dapat membantu proses konfigurasi server serta mempopulasikannya. Keseluruhan spesifikasi, baik untuk teknologi maupun hardware, juga terdokumentasikan.

3.1.1 Deskripsi Sistem (merging ke 3,1)

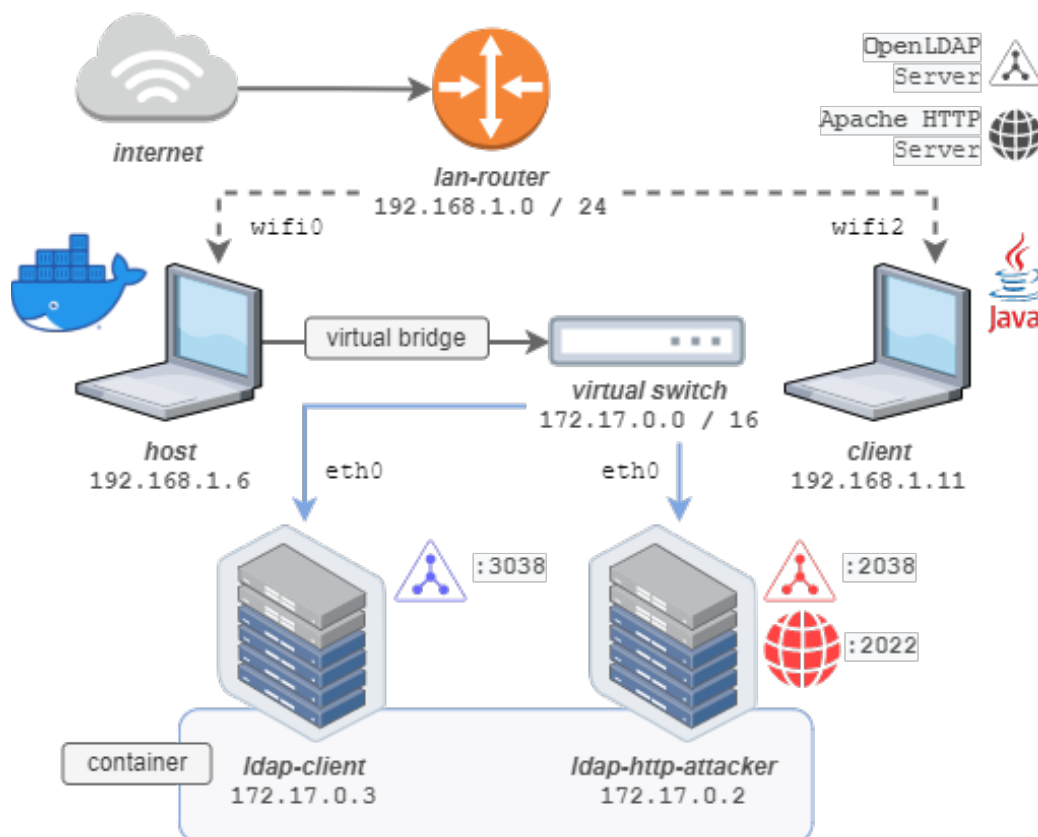
Keseluruhan sistem didasarkan pada pendekatan white box testing, dengan tujuan untuk membangun instrumen yang nantinya digunakan sebagai lingkup pengujian kerentanan Log4j pada CVE-2021-44228. Perancangan sistem terbagi menjadi dua komponen utama, yaitu pada sisi penyerang serta sisi penngguna. Perancangan pada sisi pengguna ditunjukkan untuk menjalankan fungsi kerentanan log4j serta integrasinya pada LDAP server untuk fungsi autentikasi aplikasi. Sedangkan payload malware disimpan dalam server di sisi penyerang untuk melakukan serangan remote code execution. Hal ini juga melingkupi proses pengembangan payload tersebut dan tools pengujian yang akan digunakan. Pada tahap awal, seluruh aspek sistem pengguna akan diekspos dengan konfigurasi sebagaimana adanya terhadap kerentanan Log4j, sehingga dapat dilihat tingkat keberhasilan pengujian pada pra dan pasca mitigasinya.

3.1.2 Desain Topologi Jaringan

Dalam membangun keseluruhan sistem, adapun topologi jaringan yang dirancang untuk menggambarkan keseluruhan arsitektur instrumen pengujian. Keterangan terhadap topologi jaringan dipaparkan pada gambar 3.1 sebagai berikut:

- a. Jaringan sistem menggunakan dua buah mesin laptop, baik sebagai host untuk menjalankan fitur virtualisasi docker, serta sebagai media untuk sisi pengguna (client) dalam menjalankan aplikasinya secara langsung

- b. Pada mesin host, terdapat dua buah instance container yang digunakan yaitu untuk membangun OpenLDAP server pada sisi pengguna yang sifatnya standalone (ldap-client), serta untuk mesin penyerang yang juga membangun OpenLDAP server dan Apache HTTP server sebagai salah satu bentuk instrumen pengujiannya (ldap-http-attacker)
- c. Adapun keterangan penggunaan port number pada gambar 3.1. Pada ldap-client, OpenLDAP server menggunakan port :3038. Sedangkan pada ldap-http-attacker, OpenLDAP server menggunakan port :2038 serta port :2022 untuk virtual host pada Apache HTTP server. Hal ini disesuaikan dengan penggunaan port binding pada mesin host, yang juga disebutkan pada tabel 3.4 terkait spesifikasi container
- d. Lingkup topologi jaringan adalah skala Local Area Network (LAN) dengan memanfaatkan LAN-router yang tersambung pada internet



Gambar 3.1 Desain topologi jaringan

Tabel 3.1 Konfigurasi IP jaringan

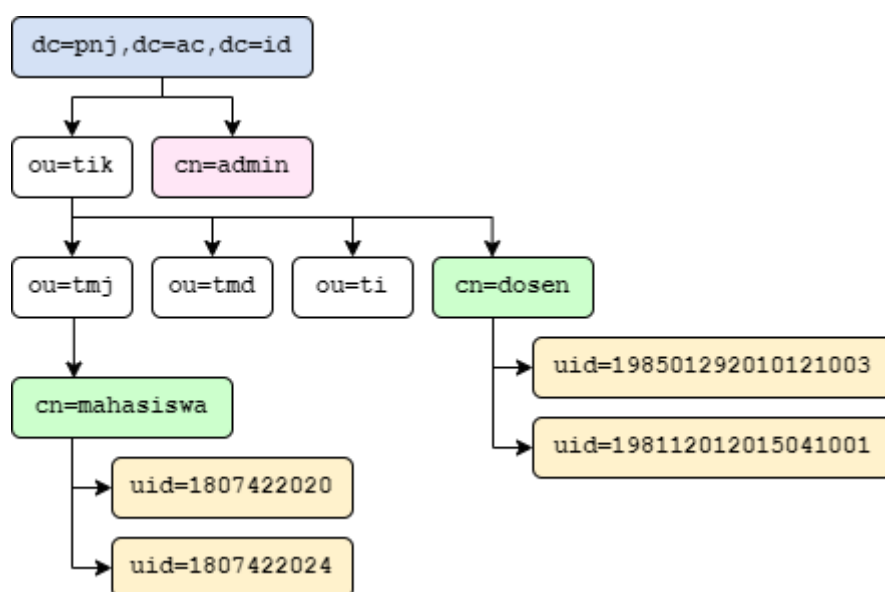
Mesin	NIC	IPv4	Gateway	CIDR
-------	-----	------	---------	------

host	wifi0	192.168.1.6	192.168.1.1	/24
client	wifi2	192.168.1.11	192.168.1.1	/24
ldap-client	eth0	172.17.0.3	172.17.0.1	/16
ldap-http-attacker	eth0	172.17.0.2	172.17.0.1	/16

3.1.3 Desain Skema LDAP

Perancangan desain skema LDAP dibutuhkan untuk menentukan bagaimana struktural penyimpanan data dapat dibangun. Perancangan meliputi penggambaran struktur penyimpanan data beserta penentuan atribut untuk setiap entri datanya. Hal ini diimplementasikan baik untuk sisi pengguna sebagai salah satu integrasi terhadap fitur autentikasi, serta untuk sisi penyerang sebagai referensi penyimpanan class payload yang digunakan saat pengujian. Pemodelan desain skema LDAP digambarkan menggunakan model Directory Information Tree (DIT), yang mana diawali oleh suatu root Relative Distinguished Name (RDN), yang umum disebut juga sebagai base DN. Base DN ini yang nantinya dapat memiliki sub entri di dalamnya yang terbuat dari satu atau lebih object class (ZyTrax, 2022). Dalam implementasinya, mayoritas operasi yang dilakukan pada skema, seperti penambahan, penghapusan dan perubahan entri, akan dilakukan oleh CN admin dengan role LDAP administrator. Konsep ini menambahkan aspek keamanan dalam bagaimana suatu skema LDAP dapat dimanajemen dengan role yang terotorisasi.

Bagian berikut merupakan pemodelan DIT serta komponen-komponen atribut LDAP di dalamnya yang digunakan untuk sisi pengguna, pada gambar 3.2 dan tabel 3.1:



Gambar 3.2 Skema DIT LDAP pada sisi pengguna

Pada gambar 3.2 diatas, base DN yang digunakan pada DIT di sisi pengguna adalah `dc=pnj,dc=ac,dc=id`, yang merupakan beberapa Domain Component (DC) dari DNS yang dipilih, yaitu `pnj.ac.id`. Dalam model tersebut, terdapat beberapa entri Organizational Unit (OU) yang menggambarkan struktur jurusan dengan program studinya secara sederhana. Dalam OU `tik`, dibuatkan pula suatu grup dengan Common Name (CN) `dosen`, yang sama halnya seperti CN `mahasiswa` dalam OU `tmj`. Implementasi ini ditunjukkan sehingga setiap entri akun dalam `dosen` dan `mahasiswa` dapat dilakukan mapping melalui Unique Identifier (UID) nya. Salah satu contoh bentuk DN dalam entri CN `mahasiswa` yaitu `uid=1807422020,cn=mahasiswa,ou=tmj,ou=tik,dc=pnj,dc=ac,dc=id`.

Pada tabel 3.1 di bawah merupakan atribut – atribut yang dimiliki setiap entri dalam skema DIT LDAP pada gambar 3.2 di atas. Adapun penggunaan beberapa atribut umum seperti mail dan initials untuk ditampilkan dalam sisi profil akun pada aplikasi GUI client nsnti. Dalam menggunakan suatu atribut, setiap object class dapat memiliki satu atau lebih atribut yang sifatnya mandatory dan opsional. Berikut merupakan beberapa entri terhadap object class serta atribut yang harus dipenuhi:

- a. tik : organizationalUnit (ou)
- b. tmj : organizationalUnit (ou)
- c. mahasiswa : posixGroup (gidNumber)
- d. 1807422020 : posixAccount (cn, uid, uidNumber, gidNumber, homeDirectory)
: inetOrgPerson
: shadowAccount (uid)

Pada object class `posixGroup` dan `posixAccount`, keduanya membutuhkan `gidNumber` untuk mengidentifikasi suatu grup dalam skema, yang sama halnya dengan `uid` dan `uidNumber` terhadap akun dalam grup tersebut. Pada entri yang berasosiasi dengan `posixAccount`, atribut `General Electric Comprehensive Operating Supervisor (GECOS)` digunakan untuk memberikan keterangan tambahan terhadap

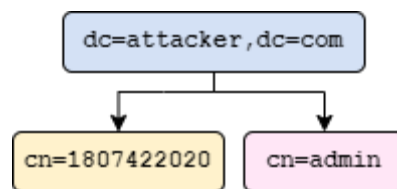
lingkup akun dalam skema serta homeDirectory digunakan sebagai full path kepada direktori akun tersebut pada suatu sistem.

Tabel 3.2 Atribut skema LDAP pada sisi pengguna

RDN	Atribut	
cn=admin	cn	admin
	description	LDAP administrator
ou=tik	ou	tik
ou=tmj	ou	tmj
ou=tmd	ou	tmd
ou=ti	ou	ti
cn=dosen	cn	dosen
	gidNumber	1000
uid=19850129 2010121003	uid	198501292010121003
	sn	Suhandana
	cn	Ariawan Andi Suhandana
	displayName	Ariawan Andi Suhandana
	initials	Ariawan
	mail	ariawan.andisuhandana@tik.pnj.ac.id
	uidNumber	1000198501292010121003
	gidNumber	1000
	gecos	TIK
	homeDirectory	/home/ldap/TIK/198501292010121003
uid=19811201 2015041001	uid	198112012015041001
	sn	Arnaldy
	cn	Defiana Arnaldy
	displayName	Defiana Arnaldy
	initials	Defian
	mail	defiana.arnaldy@tik.pnj.ac.id
	uidNumber	1000198112012015041001
	gidNumber	1000
	gecos	TIK
	homeDirectory	/home/ldap/TIK/198112012015041001
cn=mahasiswa	cn	Mahasiswa
	gidNumber	2000
uid=18074220 20	uid	1807422020
	sn	Irsyad
	cn	Muhammad Nur Irsyad
	displayName	Muhammad Nur Irsyad
	initials	Irsyad
	mail	muhammad.nurirsyad.tik18@mhs.pnj.ac.id
	uidNumber	20001807422020
	gidNumber	2000
	gecos	TMJ_CCIT_SEC_8
	homeDirectory	/home/ldap/TIK/TMJ/1807422024
uid=18074220	uid	1807422024

24	sn	Nugroho
	cn	Muhammad Fauzan Nugroho
	displayName	Muhammad Fauzan Nugroho
	initials	Fauzan
	mail	fauzan.nugroho.tik18@mhsw.pnj.ac.id
	uidNumber	20001807422024
	gidNumber	2000
	gecos	TMJ_CCIT_SEC_8
	homeDirectory	/home/ldap/TIK/TMJ/1807422024

Pada bagian berikut merupakan pemodelan DIT serta komponen-komponen atribut LDAP di dalamnya yang digunakan untuk sisi penyerang, di gambar 3.3 dan tabel 3.2:



Gambar 3.3 DIT skema LDAP pada sisi penyerang

Pada gambar 3.3 diatas, base DN yang digunakan pada DIT di sisi penyerang adalah `dc=attacker,dc=com` sebagai pembeda dari DNS yang sudah digunakan sebelumnya. Dalam model tersebut, terdapat satu entri CN untuk mengalokasikan referensi class payload yang tersimpan pada remote server, dengan value CN yang dimiripkan oleh salah satu UID pada CN mahasiswa di gambar 3.2 sebelumnya. Contoh bentuk DN yang dapat terbuat dari CN ini yaitu `cn=1807422020,dc=attacker,dc=com`.

Pada skema LDAP di sisi penyerang, entri tidak menggunakan UID sebagai fungsi mapping nya, dikarenakan entri tidak dibentuk dari object class `posixAccount` dan tidak ada asosisasi dengan `posixGroup`. Dalam rangka untuk menyimpan referensi remote class payload, adapun object class dalam skema yang dapat digunakan yaitu `javaNamingReference`. `javaNamingReference` merupakan suatu object class untuk melakukan referensi JNDI yang bertipe `auxiliary`, yang berarti hanya bersifat sebagai karakteristik tambahan terhadap entri. Dikarenakan tipe `auxiliary` tidak dapat berdiri sendiri, maka entri kemudian harus ditambahkan minimal satu object class yang sifatnya `structural` sebagai dasar utamanya. Dalam implementasinya, digunakan satu

object class yang sifatnya structural namun tidak memiliki atribut yang sifatnya mengikat, salah satunya adalah object class device. Sehingga, object class yang digunakan serta atribut yang harus dipenuhi pada entri tersebut adalah object class device (cn) dan javaNamingReference (javaClassName).

Pada tabel 3.2 di bawah merupakan atribut – atribut yang dimiliki setiap entri dalam skema DIT LDAP pada gambar 3.3 di atas. Terdapat 3 atribut utama dalam entri yang digunakan untuk dapat mereferensikan remote class payload pada suatu server, yaitu javaCodebase, javaClassName, dan javaFactory. Atribut javaClassName berisikan keseluruhan URI yang mencakup URL beserta nama file dari Java class tersebut. Adapun dua atribut lainnya merupakan komponen dari javaClassName itu sendiri, yaitu javaCodebase untuk menyimpan URL dimana lokasi file class tersebut disimpan, dalam kasus ini adalah pada remote server, serta javaFactory menyimpan nama Java class yang sifatnya case-sensitive.

Tabel 3.3 Atribut skema LDAP pada sisi penyerang

RDN	Atribut	
cn=admin	cn	admin
	description	LDAP administrator
cn=1807422020	cn	1807422020
	javaClassName	http://192.168.1.6:2022/Payload.class
	javaCodebase	http://192.168.1.6:2022/
	javaFactory	Payload

3.1.4 Spesifikasi Sistem (docker container inspect [name])

Perancangan sistem pada sisi pengguna maupun penyerang meliputi penggunaan beragam macam software pendukung yang dijalankan dalam instance container docker. Berikut merupakan keterangan spesifikasi sistem yang digunakan selama penelitian dan pengujian berlangsung, yang mana juga direferensikan dari topologi jaringan pada gambar 3.1 dan pada batasan masalah:

Tabel 3.4 Spesifikasi sistem

A	Container		
No	Nama	Spesifikasi	
1	ldap-client	OS Image	Ubuntu Server 20.04
		Shell	/bin/bash

		Port Bindings	3000 - 3100 / tcp
		Physical Size	42.9 MB
		Virtual Size	469 MB
		Bridge Network	172.17.0.1 / 16
		IP Address	172.17.0.3
2	ldap-http-attacker	OS Image	Ubuntu Server 20.04
		Shell	/bin/bash
		Port Bindings	2000 - 2100 / tcp
		Physical Size	33.4 MB
		Virtual Size	1.18 GB
		Bridge Network	172.17.0.1 / 16
		IP Address	172.17.0.2
B	Software		
No	Nama		Qty
1	Docker Desktop Server (4.8.2)		1
2	Apache HTTP Server (2.4.41)		1
3	OpenLDAP Server (2.4.49)		2
4	JXplorer (3.3.1.2)		1
5	Windows Terminal (1.12)		1
6	Sublime Text Editor (4126)		1
7	Apache NetBeans IDE (12.3)		2
8	Oracle Java SDK (1.8.0_321 & 1.8.0_181)		2
9	Apache Maven (3.6.3)		2

3.2 Realisasi Sistem

3.2.1 Implementasi Sistem Pengguna

3.2.1.1 Instalasi dan Konfigurasi OpenLDAP Server

[mulai dari docker pull → docker exec]

[install openldap]

[ldif + reference gambar di 3.2]

[test ldapsearch client]

3.2.1.2 Pengembangan Aplikasi GUI Desktop LDAP Client

[class diagram]

[activiy diagram → client (user // gui // ldap)]

[structure tree]

[minimum viable product]

3.2.2 Implementasi Sistem Penyerang

3.2.2.1 Instalasi dan Konfigurasi OpenLDAP Server

[mulai dari docker pull → docker exec]

[install openldap]

[ldif + reference gambar di 3.3]

[test ldapsearch attacker]

3.2.2.2 Instalasi dan Konfigurasi Apache HTTP Server

[pakai container yg sudah ada]

[install apache2]

[proses buat virtual host + touch file payload]

[test curl + lynx]

3.2.2.3 Pengembangan Aplikasi Java HTTP Server

[class diagram]

[activiy diagram → attacker (user // http)]

[structure tree]

[test curl + lynx]

[minimum viable product]

3.2.2.4 Pengembangan Java Payload

[class diagram]

[activiy diagram : pada 3.3 untuk attended & unattended pentest]

[minimum viable product]

3.3 Skenario Pengujian

[uji 1 : integration testing + alpha testing]

[uji 2 : PTES]

[attended : act. diag → client (user // gui) & attacker (ldap // http // system)]

[unattended : act. diag → client (system) & attacker (java // ldap // http // system)]

3.4 Skenario Analisis

[uji 1 : integration testing + alpha testing]

[uji 2 : non parametrik (cochran) + parameter dari attack tree]

BAB IV

HASIL DAN PEMBAHASAN

4.1 Pengujian

4.1.1 Deskripsi Pengujian

4.1.2 Prosedur Pengujian Aplikasi

4.1.2.1 Integration Testing

[client]

- LDAP Context
- Log4j Rolling Files
- Config Properties

[attacker java http]

- Remote Config Properties
- Log4j Rolling Files

[attacker java payload]

- Remote Config Properties

4.1.2.2 Alpha Testing

[client]

- LDAP Authentication Entries
- Log4j Message Lookup Substitution
- Remote JNDI Lookup Context

[attacker java http]

- Custom HTTP Header Request
- Log4j Message Lookup Substitution

[attacker java payload]

- Local Encrypted Reverse Shell

4.1.3 Prosedur Pengujian Kerentanan Sistem

[jelasin step secara luas + aksi exploit → mitigasi → post exploit]

4.1.3.1 Pre-Engagement

[dokumentasi whitebox]

4.1.3.2 Intelligence Gathering

[osint whitebox]

4.1.3.3 Threat Modelling

[attack trees]

4.1.3.4 Vulnerability Analysis

[research CVE-2021-44228 + SSVC]

4.1.3.5 Exploitation

[gui → log4shell attended hands-on-keyboard]

4.1.3.6 Post-Exploitation

[cronjob → log4shell unattended daemon RAT]

4.1.3.7 Mitigation

[mitigasi untuk 4.1.3.5 dan 4.1.3.6]

4.1.3.8 Post-Mitigation Exploitation

[ulang eksploitasi untuk 4.1.3.5 dan 4.1.3.6]

4.2 Data Hasil Pengujian

[data tingkat keberhasilan dari attack tree utk java 1.81 dan 3.2.1]

4.2.1 Hasil Data Pengujian Spesifikasi Aplikasi

[evaluation matrix integration & alpha testing pakai x atau $\sqrt{}$]

4.2.2 Hasil Data Success Rate Pengujian Kerentanan Sistem

[evaluation matrix statistik cochrane]

4.3 Analisis Data

4.3.1 Analisis Data Pengujian Spesifikasi Aplikasi

[analisis uji alpha dan integration testing (success rate)]

4.3.2 Analisis Data Success Rate Pengujian Kerentanan Sistem

[uji statistik cochrane dengan table r / chi squared]

BAB V

PENUTUP

5.1 Kesimpulan

5.1 Saran

DAFTAR PUSTAKA

Alghamdi, M.I., Cordeiro, C. and Barbosa, H. (2021) *Effects of Knowledge of Cyber Security on Prevention of Attacks*, *Journal of Basic and Applied Science*. Available at: <http://sj.bu.edu.sa> (Accessed: March 14, 2022).

Apache (2021) *Apache Log4j Security Vulnerabilities*, *Apache Software Foundation*. Available at: <https://logging.apache.org/log4j/2.x/security.html> (Accessed: March 17, 2022).

Apache (2022) *Apache Log4j 2 v. 2.17.2 User's Guide*, *Apache Software Foundation*. Available at: <https://logging.apache.org/log4j/2.x/log4j-users-guide.pdf> (Accessed: March 31, 2022).

Biswas, S. *et al.* (2018) *A Study on Remote Code Execution Vulnerability in Web Applications*, *International Conference on Cyber Security and Computer Science*. Available at: <https://www.researchgate.net/publication/328956499>.

Burgee, K. (2021) "Stakeholder Specific Vulnerability Categorization: CVSS Alternative," *BSidesNoVA* [Preprint]. Available at: <https://www.youtube.com/watch?v=x8jVxxrxyck> (Accessed: April 5, 2022).

Calin, M. *et al.* (2020) *Software Vulnerabilities Overview: A Descriptive Study*, *Tsinghua Science and Technology*. doi:10.26599/TST.2019.9010003.

CEH (2013) *Trojans and Backdoors - Module 06*, *EC-Council*. Available at: <http://securitvwatch.pcmag.com>.

CVE (2021) *CVE-2021-44228*, *CVE Mitre Org*. Available at: <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-44228> (Accessed: May 4, 2022).

Dalalana, D.B. and Zorzo, A.F. (2017) "Overview and Open Issues on Penetration Test," *Journal of the Brazilian Computer Society*, 23(1). doi:10.1186/s13173-017-0051-1.

Dobrovoljc, A., Trček, D. and Likar, B. (2017) "Predicting exploitations of information systems vulnerabilities through attackers' characteristics," *IEEE Access*, 5, pp. 26063–26075. doi:10.1109/ACCESS.2017.2769063.

Hama Saeed, M.A. (2020) "Malware in Computer Systems: Problems and Solutions," *IJID (International Journal on Informatics for Development)*, 9(1), p. 1. doi:10.14421/ijid.2020.09101.

Helmke, M., Hudson, A. and Hudson, P. (2019) *Ubuntu Unleashed: 2019 Edition*, *Pearson Education, Inc.*

- Ingoldsby, T.R. (2021) *Attack Tree-based Threat Risk Analysis*, Amenaza Technologies Limited. Available at: www.amenaza.com.
- Junnila, J. (2020) *Effectiveness of Linux Rootkit Detection Tools*, University of Oulu - Master's Thesis. Available at: <http://jultika.oulu.fi/files/nbnfioulu-202004201485.pdf> (Accessed: March 28, 2022).
- Kaushik, K. et al. (2021) "A Novel Approach to Generate a Reverse Shell: Exploitation and Prevention," *International Journal of Intelligent Communication, Computing, and Networks*, 2(2). doi:10.51735/ijiccn/001/33.
- Khan, A. and Neha, R.P. (2016) "Analysis of Penetration Testing and Vulnerability in Computer Networks," *GRD Journals-Global Research and Development Journal for Engineering* |, 1(6). Available at: www.eeye.com.
- Li, Y. and Liu, Q. (2021) "A Comprehensive Review Study of Cyber-attacks and Cyber Security; Emerging Trends and Recent Developments," *Energy Reports*, 7, pp. 8176–8186. doi:10.1016/j.egyr.2021.08.126.
- Liska, A. (2021) "CVSS Scores Are Dead. Let's Explore 4 Alternatives," *RSA Conference* [Preprint]. Available at: <https://www.youtube.com/watch?v=iw67KgI0Osw&t=860s> (Accessed: April 5, 2022).
- Madhavi, D. (2016) "A White Box Testing Technique in Software Testing: Basis Path Testing," *Journal for Research*, 2(4), pp. 12–17. Available at: www.journalforresearch.org.
- Maraj, A., Rogova, E. and Jakupi, G. (2020) *Testing of Network Security Systems through DoS, SQL Injection, Reverse TCP and Social Engineering Attacks*, *Int. J. Grid and Utility Computing*. doi:10.1504/IJGUC.2020.103976.
- Miller, M. (2019) *Command And Control: Bind Vs Reverse Payloads*, Triaxiom Security. Available at: <https://www.triaxiomsecurity.com/command-and-control-bind-vs-reverse-payloads/> (Accessed: March 30, 2022).
- Muñoz, A. and Mirosh, O. (2016) *A JOURNEY FROM JNDI/LDAP MANIPULATION TO REMOTE CODE EXECUTION DREAM LAND*. Available at: <https://www.blackhat.com/> (Accessed: March 14, 2022).
- Nath, O. (2022) *Log4j Flaw: Top 10 Affected Vendors and Best Solutions to Mitigate Exploitations*, Ziff Davis LLC. Available at: <https://www.toolbox.com/it-security/vulnerability-management/articles/> (Accessed: March 17, 2022).
- Oracle (2010) *inetOrgPerson Object Class*, Oracle Corporation. Available at: <https://docs.oracle.com/cd/E19225-01/820-6551/bzbpb/index.html> (Accessed: May 5, 2022).

Oracle (2021) *Oracle Security Alert Advisory - CVE-2021-44228*, Oracle Corporation. Available at: <https://www.oracle.com/security-alerts/alert-cve-2021-44228.html> (Accessed: March 17, 2022).

Phillip, H. (2020) *Linux Rootkits Part 8: Hiding Open Ports*, TheXcellerator. Available at: https://xcellerator.github.io/posts/linux_rootkits_08/ (Accessed: March 29, 2022).

PTES (2021) *The Penetration Testing Execution Standard Documentation - Release 1.1*, The PTES Team. Available at: <https://pentest-standard.readthedocs.io/en/latest/tree.html> (Accessed: April 3, 2022).

Roy, U.K. (2015) *Advanced Java programming*, Oxford University Press. Available at: <https://india.oup.com/product/advanced-java-programming-9780199455508> (Accessed: March 31, 2022).

Ruohonen, J. (2019) “A Look at the Time Delays in CVSS Vulnerability Scoring,” *Applied Computing and Informatics*, 15(2), pp. 129–135. doi:10.1016/j.aci.2017.12.002.

Shevchenko, N. et al. (2018) *Threat Modeling: A Summary Of Available Methods*, Carnegie Mellon University: Software Engineering.

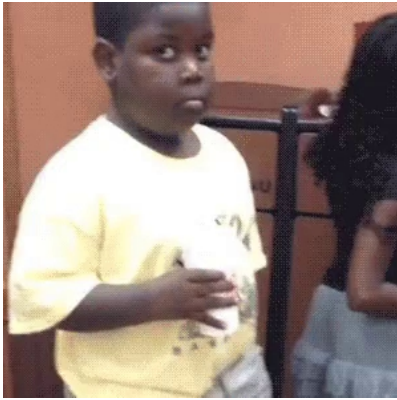
Spring, J.M. et al. (2021) *Prioritizing Vulnerability Response: A Stakeholder-Specific Vulnerability Categorization (Version 2.0)*, Carnegie Mellon University: Software Engineering Institute. Available at: https://resources.sei.cmu.edu/asset_files/WhitePaper/2021_019_001_653461.pdf (Accessed: April 5, 2022).

Stanger, J. (2022) *Beyond The Security Alert Dance: Learn Some Useful Steps*, CompTIA Inc. Available at: <https://www.comptia.org/blog/log4j-vulnerability> (Accessed: March 30, 2022).

Yin, K.S. and Khine, M.A. (2019) “Optimal Remote Access Trojans Detection Based on Network Behavior,” *International Journal of Electrical and Computer Engineering*, 9(3), pp. 2177–2184. doi:10.11591/ijece.v9i3.pp2177-2184.

ZyTrax (2022) *LDAP for Rocket Scientists*, ZyTrax Inc. Available at: <https://www.zytrax.com/books/ldap/> (Accessed: May 16, 2022).

DAFTAR RIWAYAT HIDUP PENULIS



Muhammad Nur Irsyad lahir di Jakarta, pada 19 November 1999. Penulis saat ini berdomisili di Pasar Minggu, Jakarta Selatan. Pendidikan yang telah ditempuh penulis yaitu SDI An-Nizomiyah (2006 - 2011) dan SD Vidatra YPVDP Bontang (2011 - 2012), lalu melanjutkan SMP Vidatra YPVDP Bontang (2012 - 2015), kemudian melanjutkan SMAI Al-Azhar 2 Pejaten (2015 - 2018) dalam jurusan IPA. Pada masa kuliah, penulis menempuh D II di CCIT-FTUI program Network Administrator Professional (2018 - 2020) sembari menempuh D IV di Politeknik Negeri Jakarta program studi Teknik Multimedia dan Jaringan dengan sub konsentrasi Keamanan Sistem Informasi hingga saat ini