UNIVERSITAS INDONESIA

Analisis Mendalam Transferabilitas Serangan Adversarial Single-Agent pada Multi-Agent Systems: Pengembangan Kerangka Kerja Proxy-Trained Matryoshka Worm

SEMINAR

Evandita Wiratama Putra
2206059572

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK KOMPUTER

DEPOK
JANUARI 2026

UNIVERSITAS INDONESIA

Analisis Mendalam Transferabilitas Serangan Adversarial Single-Agent pada Multi-Agent Systems: Pengembangan Kerangka Kerja Proxy-Trained Matryoshka Worm

SEMINAR

Diajukan sebagai salah satu syarat untuk memenuhi mata kuliah Seminar

Evandita Wiratama Putra
2206059572

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK KOMPUTER

DEPOK

JANUARI 2026

HALAMAN PERNYATAAN ORISINALITAS

Seminar ini adalah hasil karya saya sendiri, dan semua sumber baik yang dikutip maupun dirujuk telah saya nyatakan dengan benar.

Nama	: Evandita Wiratama Putra
NPM	: 2206059572
Tanda Tangan	:
Tanggal	:

KATA PENGANTAR

Puji syukur saya panjatkan kepada Tuhan Yang Maha Esa, karena atas berkat dan rahmat-Nya saya dapat menyelesaikan seminar ini. Penulisan seminar ini dilakukan dalam rangka memenuhi salah satu syarat untuk mata kuliah Seminar pada Program Studi Teknik Komputer, Fakultas Teknik, Universitas Indonesia.

Saya menyadari bahwa tanpa bantuan dan bimbingan dari berbagai pihak, sulit bagi saya untuk menyelesaikan seminar ini. Oleh karena itu, saya mengucapkan terima kasih kepada:

1. Bapak Yan Maraden, S.T., M.T., M.Sc., selaku Dosen Pembimbing, yang telah memberikan bimbingan, arahan, dan masukan ilmiah yang sangat berharga selama proses penelitian dan penyusunan seminar ini;
2. Prof. Dr.Eng. Ir. Arief Udhiarto, S.T., M.T., IPU., selaku Ketua Departemen Teknik Elektro Universitas Indonesia, yang telah memberikan dukungan administratif dan fasilitas yang memungkinkan terlaksananya penelitian ini;
3. Dr. Ir. Muhammad Salman, S.T., M.I.T., selaku Kepala Program Studi, yang telah memberikan arahan akademik dan dukungan dalam pelaksanaan kegiatan akademis saya;
4. Seluruh Dosen Departemen Teknik Elektro Universitas Indonesia, atas ilmu, saran, dan diskusi akademik yang turut memperkaya kajian ini;
5. Seluruh civitas Departemen Teknik Elektro Universitas Indonesia, atas dukungan administratif, fasilitas, dan semangat kebersamaan yang senantiasa mendorong saya menyelesaikan tugas akhir ini.

Akhir kata, saya berharap Tuhan Yang Maha Esa berkenan membalas segala kebaikan semua pihak yang telah membantu. Semoga seminar ini membawa manfaat bagi pengembangan ilmu.

Depok, Januari 2026

Penulis

(Evandita Wiratama Putra)

Universitas Indonesia

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI
TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademika Universitas Indonesia, saya yang bertanda tangan di bawah ini menyatakan bahwa demi pengembangan ilmu pengetahuan saya memberikan kepada Universitas Indonesia Hak Bebas Royalti Noneksklusif (Non-exclusive Royalty-Free Right) atas karya ilmiah saya yang berjudul:

"Analisis Mendalam Transferabilitas Serangan Adversarial Single-Agent pada Multi-Agent Systems: Pengembangan Kerangka Kerja Proxy-Trained Matryoshka Worm"

Beserta perangkat yang ada (jika diperlukan). Dengan hak ini Universitas Indonesia berhak menyimpan, mengalihmedia/format-kan, mengelola dalam bentuk pangkalan data (database), merawat, dan memublikasikan tugas akhir saya dengan ketentuan tetap mencantumkan nama saya sebagai penulis/pencipta dan pemilik hak cipta.

Adapun data identitas saya adalah sebagai berikut:

Nama	: Evandita Wiratama Putra
NPM	: 2206059572
Program Studi	: Teknik Komputer
Fakultas	: Teknik
Jenis karya	: Seminar

Demikian pernyataan ini saya buat dengan sebenar-benarnya untuk dapat dipergunakan sebagaimana mestinya.

Dibuat di : Depok
Pada tanggal : Januari 2026

Yang menyatakan

(Evandita Wiratama Putra)

ABSTRAK

Nama : Evandita Wiratama Putra

Program Studi : Teknik Komputer

Judul : Analisis Mendalam Transferabilitas Serangan Adversarial Single-Agent pada Multi-Agent Systems: Pengembangan Kerangka Kerja Proxy-Trained Matryoshka Worm

Pembimbing : Yan Maraden, S.T., M.T., M.Sc.

Ringkasan

Evolusi pesat teknologi Large Language Model (LLM) telah memicu transformasi fundamental dalam arsitektur kecerdasan buatan, bergeser dari model interaksi statis menuju ekosistem Multi-Agent Systems (MAS) yang dinamis dan terorkestrasi. Dalam lanskap ini, "arsitektur kognitif"—yang mencakup system prompts, instruksi prosedural, spesifikasi alat (tools), dan topologi interaksi—telah muncul sebagai kelas aset Kekayaan Intelektual (IP) yang sangat bernilai namun rentan. Penelitian ini menyajikan analisis komprehensif mengenai transferabilitas serangan adversarial dari agen tunggal ke lingkungan multi-agen yang kompleks, serta mengusulkan kerangka kerja metodologis baru yang disebut "Proxy-Trained Matryoshka Worm" (PTMW). Berbeda dengan pendekatan Multi-Agent Reinforcement Learning (MARL) konvensional yang sering kali gagal akibat ledakan ruang keadaan dan kelangkan reward pada pengaturan black-box, penelitian ini merekonseptualisasi ekstraksi IP MAS sebagai serangkaian "Stateless Unit Tests" yang diorkestrasi secara eksternal.

Metodologi yang diusulkan mensintesis dua pendekatan adversarial mutakhir: kerangka kerja LeakAgent berbasis Reinforcement Learning (RL) yang menggunakan Proximal Policy Optimization (PPO) untuk menghasilkan prompt adversarial ("hulu ledak") yang optimal, dan mekanisme pengiriman PTMW yang memanfaatkan Client-Side Memory serta algoritma Iterative Peeling. Melalui prinsip enkapsulasi rekursif yang menyerupai boneka Matryoshka, serangan ini dirancang untuk menembus batas kepercayaan implisit antar-agen, memitigasi degradasi konteks ("Lost in the Middle"), dan memetakan topologi sistem secara deterministik. Laporan ini merinci perancangan sistem, landasan teoretis yang mendalam, serta protokol evaluasi empiris untuk memvalidasi hipotesis bahwa kompleksitas topologi MAS dapat didekonstruksi lapis demi lapis, mengubah "kotak hitam" menjadi "kotak kaca", dan mengekspos kerentanan kritis dalam arsitektur MAS modern yang bergantung pada security by obscurity.

Kata kunci: Multi-Agent Systems, Adversarial Attacks, Large Language Models, Reinforcement Learning, Matryoshka Worm, LeakAgent, Keamanan Siber, Ekstraksi Kekayaan Intelektual.

ABSTRACT

Name	: Evandita Wiratama Putra
Study Program	: Computer Engineering
Title	: In-Depth Analysis of Adversarial Attack Transferability from Single-Agent to Multi-Agent Systems: Development of Proxy-Trained Matryoshka Worm Framework
Supervisor	: Yan Maraden, S.T., M.T., M.Sc.

Ringkasan

The rapid evolution of Large Language Model (LLM) technology has triggered a fundamental transformation in artificial intelligence architectures, shifting from static interaction models to dynamic and orchestrated Multi-Agent Systems (MAS) ecosystems. In this landscape, the "cognitive architecture"—comprising system prompts, procedural instructions, tool specifications, and interaction topologies—has emerged as a highly valuable yet vulnerable class of Intellectual Property (IP) assets. This research presents a comprehensive analysis of the transferability of single-agent adversarial attacks to complex multi-agent environments and proposes a novel methodological framework called the "Proxy-Trained Matryoshka Worm" (PTMW). Unlike conventional Multi-Agent Reinforcement Learning (MARL) approaches, which often fail due to state space explosion and reward sparsity in black-box settings, this study reconceptualizes MAS IP extraction as a series of externally orchestrated "Stateless Unit Tests."

The proposed methodology synthesizes two cutting-edge adversarial approaches: the Reinforcement Learning (RL)-based LeakAgent framework utilizing Proximal Policy Optimization (PPO) to generate optimal adversarial prompts ("warheads"), and the PTMW delivery mechanism leveraging Client-Side Memory and Iterative Peeling algorithms. Through recursive encapsulation principles resembling Matryoshka dolls, this attack is designed to penetrate implicit inter-agent trust boundaries, mitigate context degradation ("Lost in the Middle"), and deterministically map system topology. This report details the system design, in-depth theoretical foundations, and empirical evaluation protocols to validate the hypothesis that MAS topological complexity can be deconstructed layer by layer, turning "black boxes" into "glass boxes," and exposing critical vulnerabilities in modern MAS architectures reliant on security by obscurity.

Keywords: Multi-Agent Systems, Adversarial Attacks, Large Language Models, Reinforcement Learning, Matryoshka Worm, LeakAgent, Cybersecurity, Intellectual Property Extraction.

Daftar Isi

Daftar Gambar

Daftar Tabel

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi kecerdasan buatan saat ini mengalami transformasi yang signifikan. Sebelumnya, sistem AI didominasi oleh model bahasa besar (*Large Language Model*/LLM) yang beroperasi secara mandiri. Namun saat ini, telah berkembang sistem yang lebih kompleks yaitu *Multi-Agent Systems* (MAS). Jika LLM tradisional dapat dianalogikan sebagai seorang ahli yang bekerja secara independen, maka MAS merepresentasikan sebuah tim yang terdiri dari beberapa agen dengan keahlian yang berbeda, misalnya yang berperan sebagai analis keuangan, programmer, atau pengawas etika, yang berkolaborasi untuk menyelesaikan permasalahan kompleks.[1] Platform seperti CrewAI, AutoGen, dan Coze memfasilitasi pengembang untuk membangun sistem semacam ini, di mana keluaran dari satu agen dapat menjadi masukan bagi agen lainnya, sehingga menghasilkan solusi yang lebih optimal dibandingkan dengan penggunaan model tunggal.

Seiring dengan transformasi arsitektural ini, metodologi evaluasi keamanan yang telah dikembangkan untuk *single-agent* LLM menghadapi tantangan signifikan ketika diaplikasikan pada lingkungan *multi-agent*. Teknik-teknik serangan *adversarial* yang terbukti efektif untuk mengekstraksi informasi sensitif dari agen tunggal, seperti *prompt injection*, *jailbreaking*, dan *privacy leakage attacks*, mengalami penurunan efektivitas drastis dalam konteks MAS.[3] Hal ini disebabkan oleh beberapa faktor fundamental, seperti kompleksitas topologi komunikasi antar agen yang menciptakan lapisan pertahanan implisit melalui distribusi informasi, fenomena degradasi konteks (*Lost in the Middle*) yang menyebabkan instruksi *adversarial* kehilangan efektivitasnya saat melewati rantai agen yang panjang, dan keterbatasan visibilitas penyerang terhadap *state internal* sistem yang mengakibatkan kesulitan dalam optimasi strategi serangan secara adaptif.[1]

Dalam perkembangan ini, aset berharga dari sistem AI tidak lagi terletak pada kode program atau model dasarnya semata, mengingat komponen-komponen tersebut telah banyak yang bersifat *open-source*. Aspek yang lebih krusial saat ini adalah bagaimana sistem tersebut dirancang dan dikonfigurasi, yang dikenal sebagai ”arsitektur kognitif”. Arsitektur ini mencakup *system prompt* yang mendefinisikan karakteristik dan batasan setiap agen, instruksi-instruksi untuk menjalankan tugas tertentu, spesifikasi *tools* yang digunakan agen untuk berinteraksi dengan sistem lain, serta struktur komunikasi antar agen.[1] Konfigurasi inilah yang membedakan antara asisten AI generik dengan solusi khusus yang dikembangkan untuk kebutuhan perusahaan. Oleh karena itu, menjaga kerahasiaan konfigurasi ini menjadi hal yang esensial sebagai bagian dari Kekayaan Intelektual (IP) perusahaan.

Celah metodologis antara serangan *single-agent* dan *multi-agent* memerlukan pendekatan yang secara fundamental berbeda. Penelitian terdahulu seperti PromptFuzz mengandalkan teknik *fuzzing* berbasis mutasi acak yang efektif untuk *single-agent* namun mengalami inefisiensi komputasional signifikan dalam ruang pencarian *multi-agent* yang berdimensi tinggi.[5] Pendekatan lain seperti serangan tipe *worm* (MASLEAK) berupaya melakukan propagasi *payload* melalui rantai agen secara sekuensial, namun rentan terhadap kegagalan total apabila salah satu agen dalam rantai memutus alur serangan, mengakibatkan hilangnya seluruh informasi yang telah diekstrak sebelumnya. Keterbatasan-keterbatasan ini mengindikasikan kebutuhan mendesak akan kerangka kerja baru yang mampu mengatasi kompleksitas struktural sistem *multi-agent* melalui pendekatan yang lebih *resilient* dan adaptif.

Untuk mengatasi tantangan tersebut, penelitian ini mengusulkan pendekatan *iterative peeling* sebagai mekanisme inti dalam navigasi topologi MAS. Berbeda dengan serangan konvensional yang berupaya melintasi seluruh sistem dalam satu eksekusi tunggal, *iterative peeling* mengadopsi strategi bertahap di mana setiap agen dalam sistem diekstrak secara independen melalui iterasi terpisah. Pendekatan ini dianalogikan dengan proses mengupas lapisan demi lapisan pada struktur berlapis, di mana setiap iterasi fokus pada satu target spesifik tanpa membawa beban kontekstual dari ekstraksi sebelumnya. Keunggulan utama dari pendekatan ini terletak pada kemampuannya untuk mengisolasi kegagalan, artinya jika ekstraksi terhadap satu agen gagal, hal tersebut tidak akan mempengaruhi keberhasilan ekstraksi agen-agen lain yang telah berhasil dilakukan. Setiap iterasi dimulai dengan kondisi awal yang bersih, mengirimkan *query* yang dirancang khusus untuk menargetkan agen tertentu berdasarkan pengetahuan topologi yang telah diperoleh dari iterasi sebelumnya, kemudian mengekstrak informasi dan menyimpannya secara eksternal sebelum melanjutkan ke target berikutnya. Implementasi *iterative peeling* memerlukan pemahaman dinamis tentang struktur sistem target, yang diperoleh secara progresif melalui analisis respons dari setiap agen, sehingga memungkinkan sistem serangan untuk membangun peta topologi secara bertahap dan menyesuaikan strategi serangan berdasarkan karakteristik yang terobservasi dari setiap agen.

Komponen krusial yang mendukung efektivitas *iterative peeling* adalah implementasi *Client-Side Memory*, yaitu repositori eksternal yang berfungsi sebagai basis pengetahuan persisten di sisi penyerang. *Client-Side Memory* dirancang untuk menyimpan dua kategori informasi vital. Pertama, data intelektual yang berhasil diekstrak dari setiap agen, meliputi *system prompt*, spesifikasi *tools*, instruksi tugas, dan metadata topologi yang mendeskripsikan koneksi antar agen dalam sistem. Kedua, *prompt* spesifik yang terbukti berhasil mengekstrak informasi dari masing-masing agen, berfungsi sebagai *attack signature* yang efektif untuk target tersebut. Keberadaan memori ini memberikan kemampuan adaptif yang

signifikan terhadap sistem serangan. Dalam skenario di mana pada iterasi tertentu terdapat agen yang menunjukkan perilaku berbeda dari sebelumnya, misalnya karena perubahan konfigurasi, penerapan mekanisme pertahanan baru, atau variasi respons probabilistik dari model, sistem dapat merujuk kembali ke memori untuk mengidentifikasi *prompt* yang pernah berhasil dan melakukan modifikasi adaptif berdasarkan pola keberhasilan historis. Lebih lanjut, *Client-Side Memory* memungkinkan sistem untuk melakukan analisis komparatif antar agen, mengidentifikasi kesamaan karakteristik yang dapat dieksploitasi, serta membangun strategi serangan yang lebih efisien dengan memanfaatkan pengetahuan yang terakumulasi dari setiap interaksi. Dengan demikian, kombinasi antara *iterative peeling* dan *Client-Side Memory* menciptakan kerangka kerja yang tidak hanya lebih *resilient* terhadap kegagalan parsial, tetapi juga mampu melakukan pembelajaran dan adaptasi berkelanjutan selama proses ekstraksi berlangsung, meningkatkan probabilitas keberhasilan secara keseluruhan dalam mengekstrak arsitektur kognitif lengkap dari sistem MAS target.

1.2 Rumusan Masalah

Berdasarkan celah metodologis yang telah diidentifikasi dalam latar belakang, penelitian ini merumuskan tiga permasalahan fundamental terkait adaptasi metodologi serangan *adversarial* dari lingkungan *single-agent* ke arsitektur *multi-agent* yang kompleks:

1. **Ketidakefektifan Teknik Serangan *Single-Agent* pada Lingkungan MAS**

   Bagaimana mengadaptasi teknik serangan *adversarial* yang telah terbukti efektif pada *single-agent* LLM, seperti *prompt injection* dan *jailbreaking*, agar dapat mengatasi kompleksitas struktural sistem *multi-agent*? Teknik-teknik tersebut mengalami degradasi performa signifikan ketika diterapkan pada MAS karena distribusi informasi antar agen, fenomena degradasi konteks (*Lost in the Middle*) yang menyebabkan kegagalan propagasi *payload* dalam rantai komunikasi yang panjang, serta keterbatasan visibilitas terhadap *state* internal sistem yang mengakibatkan kesulitan dalam optimasi strategi serangan secara adaptif pada pengaturan *black-box*.

2. **Kebutuhan Mekanisme *Iterative Peeling* dan *Client-Side Memory***

   Bagaimana merancang dan mengimplementasikan mekanisme *iterative peeling* yang mampu melakukan ekstraksi informasi secara bertahap per agen tanpa mengalami kegagalan total akibat *single point of failure*? Lebih lanjut, bagaimana mengintegrasikan *Client-Side Memory* sebagai repositori eksternal yang menyimpan data intelektual hasil ekstraksi dan *prompt* yang berhasil per agen, sehingga sistem dapat melakukan modifikasi *prompt* secara adaptif ketika menghadapi perubahan perilaku agen atau kegagalan ekstraksi pada iterasi tertentu, dengan memanfaatkan pengetahuan historis yang terakumulasi dari interaksi sebelumnya?

3. **Transferabilitas Serangan dari Model Proksi ke Target *Black-Box***

   Apakah mungkin untuk melatih *attack agent* menggunakan *Reinforcement Learning* pada model proksi *open-source* (pengaturan *white-box*) dan mentransfer efektivitas *prompt adversarial* yang dihasilkan untuk mengekstrak IP dari berbagai arsitektur MAS *black-box* dengan model dan konfigurasi yang tidak diketahui sebelumnya? Hipotesis transferabilitas ini bergantung pada asumsi adanya keselarasan universal (*universal alignment*) dalam mekanisme model bahasa untuk mengikuti instruksi sistem, yang perlu divalidasi secara empiris melalui eksperimen pada berbagai platform MAS komersial dan model LLM yang berbeda.[1]

1.3 Tujuan Penelitian

Penelitian ini bertujuan untuk mengembangkan kerangka kerja serangan *adversarial* yang mampu mengatasi keterbatasan metodologi *single-agent* dalam konteks *multi-agent systems*, dengan fokus pada tiga tujuan spesifik:

1. **Pengembangan Mekanisme *Iterative Peeling* dengan *Client-Side Memory***

   Merancang dan mengimplementasikan algoritma *iterative peeling* yang melakukan ekstraksi informasi secara bertahap per agen untuk menghindari kegagalan total akibat *single point of failure*. Implementasi mencakup pengembangan *Client-Side Memory* sebagai repositori eksternal yang menyimpan data intelektual hasil ekstraksi (*system prompt*, spesifikasi *tools*, instruksi tugas, dan metadata topologi) serta *prompt* yang berhasil per agen, yang memungkinkan sistem melakukan modifikasi *prompt* secara adaptif berdasarkan pengetahuan historis ketika menghadapi perubahan perilaku agen atau kegagalan ekstraksi pada iterasi tertentu.[1]

2. **Optimasi *Prompt Adversarial* Menggunakan *Reinforcement Learning***

   Mengimplementasikan *pipeline* pelatihan menggunakan algoritma *Proximal Policy Optimization* (PPO) dengan fungsi *reward* berbasis *Word Edit Similarity* (WES) untuk melatih *attack agent* pada model proksi *open-source*. Tujuannya adalah menghasilkan *prompt adversarial* yang memiliki efektivitas tinggi, keragaman yang memadai untuk menghindari deteksi, dan transferabilitas yang kuat untuk dapat diaplikasikan pada berbagai arsitektur MAS *black-box* dengan model LLM yang berbeda-beda.[3]

3. **Evaluasi Empiris Transferabilitas dan Efektivitas Sistem**

   Melakukan eksperimen komprehensif pada tiga kategori lingkungan, yakni dataset sintetis untuk validasi internal dengan *ground truth* yang diketahui, aplikasi *honeypot*

pada platform komersial (Coze, CrewAI) untuk pengujian kondisi *real-world*, dan aplikasi produktif dari GPT Store untuk validasi transferabilitas *in the wild*. Evaluasi menggunakan metrik kuantitatif meliputi *Attack Success Rate* (ASR), *Topology Fidelity* (GS_{topo}), dan *Extraction Efficiency*, dengan perbandingan terhadap metode *baseline* seperti MASLEAK dan PromptFuzz untuk mengukur peningkatan performa yang dicapai.[8]

1.4 Manfaat Penelitian

Penelitian ini diharapkan memberikan kontribusi yang signifikan dan multidimensi, baik bagi komunitas akademik maupun praktisi industri:

- **Manfaat Akademis:**
  - Perluasan Teori Adversarial AI: Memberikan wawasan baru mengenai batasan keamanan MAS dan validitas penggunaan model proksi untuk serangan transfer.
  - Metodologi Evaluasi Baru: Memperkenalkan metrik evaluasi topologi graf (Topology Fidelity) dan fungsi reward semantik (WES) yang dapat diadopsi oleh peneliti lain untuk mengukur ketahanan sistem multi-agen secara lebih presisi.
- **Manfaat Praktis:**
  - Peningkatan Keamanan Industri: Menunjukkan bukti empiris (proof-of-concept) mengenai kerentanan arsitektur MAS saat ini, yang dapat mendorong pengembangan platform (seperti Microsoft, OpenAI, dan penyedia kerangka kerja agen) untuk beralih dari model keamanan berbasis ketidakjelasan menuju pertahanan yang lebih kuat.
  - Kesadaran Perlindungan IP: Memberikan peringatan dini kepada perusahaan yang berinvestasi dalam "Cognitive IP" mengenai risiko kloning sistem dan spionase ekonomi.
- **Manfaat Institusional:**
  - Memperkaya portofolio penelitian Universitas Indonesia dalam domain keamanan siber dan kecerdasan buatan tingkat lanjut.

1.5 Batasan Masalah

Untuk memastikan kedalaman analisis dan kelayakan pelaksanaan dalam kerangka waktu penelitian, lingkup masalah dibatasi sebagai berikut:

- **Fokus pada Ekstraksi IP:**

  Penelitian ini secara khusus menargetkan ekstraksi komponen Kekayaan Intelektual (IP) dari Multi-Agent Systems. Fokus ekstraksi meliputi system prompts yang mendefinisikan peran dan batasan perilaku setiap agen, instruksi tugas yang memandu alur kerja dan logika pengambilan keputusan, deskripsi alat (tool specifications) yang menentukan kapabilitas eksternal agen, serta topologi sistem yang menggambarkan struktur komunikasi dan hierarki antar-agen. Penelitian ini tidak mencakup aspek serangan lain seperti manipulasi output, denial of service, atau eksploitasi kerentanan infrastruktur teknis di luar konteks ekstraksi IP kognitif.

- **Domain Multi-Agent Systems:**

  Analisis dibatasi pada sistem multi-agen berbasis Large Language Model (LLM) yang berinteraksi terutama melalui teks dalam bahasa alami dan API terstruktur. Sistem yang menggunakan modalitas lain seperti gambar, audio, atau video tidak termasuk dalam ruang lingkup penelitian ini. Fokus pada komunikasi tekstual memungkinkan evaluasi mendalam terhadap kerentanan prompt injection dan manipulasi instruksi linguistik, yang merupakan vektor serangan utama dalam ekosistem LLM kontemporer. Pembatasan ini juga memfasilitasi implementasi metrik evaluasi semantik seperti Word Edit Similarity yang bergantung pada analisis teks.

- **Lingkungan Eksperimen:**

  Evaluasi dilakukan pada lingkungan terkendali menggunakan dataset sintetis MASD (Multi-Agent System Dataset) yang berisi 810 aplikasi MAS sintetis, serta aplikasi ”honeypot” yang di-deploy khusus untuk tujuan penelitian pada platform komersial seperti Coze dan CrewAI. Pendekatan ini memastikan bahwa tidak ada sistem produksi yang terpapar risiko tanpa izin eksplisit, sejalan dengan prinsip ethical research. Dataset sintetis menyediakan ground truth yang diperlukan untuk pengukuran presisi metrik seperti Topology Fidelity, sementara honeypots pada platform komersial memberikan validitas ekologis dengan menguji sistem dalam kondisi real-world dengan API limitations dan potential defensive mechanisms.

- **Model Bahasa:**

  Pelatihan agen penyerang akan menggunakan model open-source seperti Llama-3-8B sebagai attack agent dan Llama-3-70B sebagai target proxy white-box. Pemilih-

an model open-source didasarkan pada pertimbangan transparansi eksperimental, reprodusibilitas hasil, dan aksesibilitas bagi komunitas penelitian yang lebih luas. Meskipun transferabilitas ke model proprietary seperti GPT-4 dan Claude akan diuji pada fase evaluasi, proses training dan optimasi kebijakan serangan dilakukan sepenuhnya menggunakan infrastruktur model terbuka untuk memastikan bahwa metodologi dapat diverifikasi dan direplikasi oleh peneliti lain tanpa bergantung pada akses ke model komersial yang berbiaya tinggi.

- **Mekanisme Pertahanan:**

  Penelitian ini berfokus pada metodologi serangan (offensive security) dengan tujuan mengidentifikasi dan mengkarakterisasi kerentanan sistem MAS saat ini. Perancangan mekanisme pertahanan (defensive countermeasures) seperti prompt filtering, output sanitization, atau anomaly detection tidak termasuk dalam ruang lingkup utama penelitian ini, meskipun implikasi defensif akan didiskusikan dalam konteks rekomendasi untuk mitigasi risiko. Pembatasan ini memungkinkan eksplorasi mendalam terhadap ruang serangan dan karakterisasi menyeluruh terhadap attack surface MAS, yang merupakan prasyarat untuk pengembangan pertahanan yang efektif di masa depan.

1.6 Sistematika Penulisan

Dokumen seminar ini disusun secara sistematis dalam tiga bab utama yang saling terkait untuk menyajikan argumentasi penelitian secara koheren dan komprehensif:

- **BAB I PENDAHULUAN:**

  Bab ini memaparkan latar belakang transformasi lanskap AI menuju Multi-Agent Systems, rumusan masalah mengenai kerentanan keamanan MAS, tujuan pengembangan metodologi Proxy-Trained Matryoshka Worm, manfaat penelitian bagi akademis dan industri, batasan ruang lingkup penelitian, serta sistematika penulisan dokumen.

- **BAB II TINJAUAN PUSTAKA:**

  Bab ini menyediakan fondasi teoretis melalui landasan teori (arsitektur MAS, keterbatasan MARL, PPO dalam adversarial prompting, dan fungsi reward WES), tinjauan penelitian terkait (MASLEAK, PromptFuzz, LeakAgent, AgentFuzzer), serta tabel perbandingan metodologi serangan adversarial untuk mengidentifikasi gap penelitian.

- **BAB III METODOLOGI PENELITIAN:**

Bab ini mendeskripsikan metodologi PTMW yang terdiri dari dua fase (Generasi Hulu Ledak dengan LeakAgent dan Sistem Pengiriman dengan Iterative Peeling), perancangan sistem tiga modul (The Forge, The Brain, The Interface), serta protokol evaluasi komprehensif meliputi skenario pengujian, metrik kuantitatif, pengujian hipotesis dengan validasi statistik, dan pertimbangan etis penelitian.

BAB II
TINJAUAN PUSTAKA

2.1 Landasan Teori

2.1.1 Arsitektur Kognitif Multi-Agent Systems (MAS)

Multi-Agent Systems (MAS) berbasis LLM merepresentasikan evolusi dari model "otak dalam toples" menjadi "tim digital" yang terstruktur. Dalam formalisme matematika, lingkungan MAS dapat didefinisikan sebagai tuple $\mathcal{M} = (A, G, C, P)$, di mana:

- $A = \{a_1, a_2, \ldots, a_n\}$ adalah himpunan agen otonom.
- $G = (V, E)$ adalah graf terarah yang merepresentasikan topologi komunikasi.
- C merepresentasikan konfigurasi internal atau "memori" setiap agen.
- P adalah himpunan system prompts yang mendefinisikan peran masing-masing agen.[1]

2.1.2 The Complexity Trap: Keterbatasan Multi-Agent Reinforcement Learning (MARL)

Penerapan MARL pada serangan ekstraksi IP black-box menghadapi hambatan fundamental:

1. **Ledakan Ruang Keadaan (State Space Explosion):**

   Ruang keadaan gabungan S dari MAS tumbuh secara eksponensial dengan jumlah agen dan kompleksitas interaksi mereka. Penyerang yang mencoba mempelajari urutan input optimal untuk mengekstrak informasi dari Agen n dengan memanipulasi Agen 1 harus secara implisit memodelkan fungsi transisi semua agen perantara. Dalam pengaturan black-box di mana transisi internal tidak terlihat, hal ini memerlukan inferensi model Markov tersembunyi dengan kompleksitas yang sangat besar.[1]

2. **Masalah Credit Assignment dan Kelangkan Reward:**

   Dalam serangan black-box, adversary biasanya hanya mengamati output akhir R yang dihasilkan oleh agen terakhir dalam rantai. Jika serangan gagal—misalnya, jika payload diblokir oleh Agen 3 dalam rantai 5-agen—penyerang menerima sinyal kegagalan biner (reward = 0) tanpa informasi tentang di mana atau mengapa kegagalan terjadi. Kelangkan reward ekstrem ini membuat estimasi gradient untuk jaringan kebijakan sangat sulit, sering menyebabkan algoritma MARL gagal konvergen.[1]

3. **Instabilitas dan Randomness:**

   Pendekatan otomatis yang ada yang mengandalkan algoritma genetika atau fuzzing (seperti PromptFuzz) mengalami randomness inherent. Tanpa gradient terarah, mereka efektif terdegradasi menjadi pencarian acak, yang tidak efisien secara komputasi untuk menavigasi ruang prompt bahasa alami berdimensi tinggi.[1]

2.1.3 Proximal Policy Optimization (PPO) dalam Adversarial Prompting

PPO digunakan untuk menghasilkan prompt adversarial yang stabil dan efektif. Tidak seperti algoritma genetika yang mengandalkan mutasi acak, PPO memungkinkan agen serangan memperbarui kebijakannya berdasarkan expected reward dari urutan yang dihasilkan. Ini memberikan gradient terarah yang memandu agen menuju prompt yang lebih efektif jauh lebih cepat daripada pencarian acak.[1]

Fungsi objektif PPO:

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[\min \left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right]$$

Mekanisme ”clipping” PPO mencegah pembaruan kebijakan drastis yang dapat mendestabilkan proses pembelajaran. Ini krusial saat menavigasi lanskap reward volatil dari serangan adversarial, di mana perubahan kecil dalam prompt dapat mengubah hasil dari ”sukses” ke ”penolakan”.[1] Selain itu, PPO memungkinkan definisi fungsi reward yang menghukum pelanggaran format, secara efektif mengajarkan agen serangan untuk menghasilkan prompt yang tidak hanya adversarial tetapi juga sintaksis benar untuk API target.[1]

2.1.4 Fungsi Reward Semantik: Sliding-window Word Edit Similarity (WES)

Komponen penting dari optimasi adalah fungsi reward. Reward biner (1 untuk sukses, 0 untuk gagal) tidak cukup untuk pelatihan karena ekstraksi IP yang berhasil adalah kejadian langka. Fungsi WES memberikan sinyal reward padat: jika agen target membocorkan bahkan fragmen system prompt, agen serangan menerima reward positif.[1]

Fungsi reward WES:

$$R(u,d) = (1-\lambda) \cdot \text{SWES}_{\text{norm}}(u,d) + \lambda \cdot \frac{1}{||u| - |d||}$$

Di mana u adalah respons model target, d adalah informasi privat yang diinginkan (ground truth system prompt dari proksi), λ adalah koefisien penyeimbang (ditetapkan 0.1 berdasarkan spesifikasi LeakAgent), dan $\text{SWES}_{\text{norm}}$ adalah skor kesamaan ternormalisasi yang diturunkan dari edit distance.[1]

Edit distance standar gagal ketika output target u jauh lebih panjang dari output yang diinginkan d (misalnya jika agen mengeluarkan prompt yang terkubur dalam respons percakapan). Sliding-window Word Edit Similarity mengatasi ini dengan menerapkan sliding window di atas u untuk menemukan substring yang paling cocok dengan d. Ini memastikan bahwa kebocoran ”terkubur” diberi reward dengan benar, membimbing agen menuju ekstraksi penuh.[1]

2.2 Penelitian Terkait

Beberapa penelitian terdahulu yang relevan dengan topik serangan adversarial pada MAS akan diulas berikut ini:

- **MASLEAK (IP Leakage Attacks Targeting LLM-Based Multi-Agent Systems):**

  MASLEAK merupakan serangan worm single-pass yang dirancang untuk mengekstrak IP dari MAS. Pendekatan ini mengandalkan propagasi payload tunggal melalui seluruh rantai agen. Namun, MASLEAK rentan terhadap fenomena ”Lost in the Middle” dan degradasi konteks. Ketika serangan melintasi rantai agen yang panjang, akumulasi riwayat percakapan menyebabkan agen di kedalaman rantai gagal memproses instruksi propagasi atau format data, memutus rantai serangan.[7] Kelemahan utama adalah ketergantungan pada manajemen status internal MAS, yang mengakibatkan single point of failure.

- **PromptFuzz (Fuzzing Techniques for Prompt Injection):**

  PromptFuzz menggunakan pendekatan fuzzing berbasis mutasi untuk menemukan kerentanan prompt injection. Metode ini mengandalkan mutasi acak atau heuristik tanpa panduan gradient. Dalam ruang pencarian bahasa alami berdimensi tinggi, pendekatan ini mendegradasi menjadi pencarian acak yang tidak efisien secara komputasi.[5] PromptFuzz memerlukan ribuan kueri untuk menemukan satu celah dan menghasilkan prompt dengan transferabilitas rendah antar-model. Tanpa pemahaman semantik mengapa prompt gagal, metode ini efektif merupakan random walk.

- **LeakAgent (RL-based Red-teaming for Privacy Leakage):**

  LeakAgent memperkenalkan penggunaan Reinforcement Learning dengan PPO untuk menghasilkan prompt adversarial optimal terhadap single-agent LLM. Kerangka kerja ini menggunakan fungsi reward WES yang padat dan dynamic temperature adjustment untuk menghasilkan prompt yang efektif dan beragam.[3] Namun, LeakAgent dirancang untuk serangan single-agent dan belum mendukung kompleksitas topologi MAS. Penelitian ini mengadaptasi dan memperluas LeakAgent untuk lingkungan multi-agen melalui integrasi dengan mekanisme Matryoshka.

- **AgentFuzzer (Black-Box Fuzzing for LLM Agents):**

  AgentFuzzer mengusulkan fuzzing black-box generik untuk indirect prompt injection terhadap LLM agents. Pendekatan ini fokus pada identifikasi kerentanan melalui testing sistematis tetapi tidak mengoptimalkan transferabilitas atau efisiensi ekstraksi dalam konteks MAS.[11]

Dari tinjauan penelitian terkait, dapat disimpulkan bahwa terdapat gap dalam metodologi serangan yang dapat secara efisien dan deterministik mengekstrak IP dari MAS kompleks. MASLEAK menunjukkan feasibility serangan MAS tetapi terbatas oleh single-pass propagation. LeakAgent mendemonstrasikan kekuatan RL untuk optimasi prompt tetapi tidak menangani topologi multi-agen. Penelitian ini mengisi gap tersebut dengan mensintesis keunggulan kedua pendekatan dan menambahkan Client-Side Memory serta Iterative Peeling.[1]

2.3 Tabel Perbandingan Studi Pustaka

Untuk memberikan gambaran komparatif atas penelitian-penelitian terkait yang telah diuraikan, Tabel 2.1 berikut menyajikan perbandingan beberapa studi kunci dari segi metode, kelebihan, dan keterbatasannya relatif terhadap penelitian ini.

Tabel 1: Analisis Komparatif Metodologi Serangan Adversarial pada MAS

Fitur Utama	**MASLEAK**	**PromptFuzz**	**PTMW (Proposed)**
Vektor Serangan	Single-pass ”Worm”	Black-box Fuzzing	Iterative ”Matryoshka”
Optimasi	Manual / Statis	Genetika / Heuristik	RL (PPO) dengan Dense Reward
Manajemen Keadaan	Internal	Tidak Ada	Eksternal (Client-Side Memory)
Transferabilitas	Terbatas	Rendah	Tinggi
Resiliensi	Rendah (single point of failure)	Tidak Konsisten	Tinggi (dapat retry per layer)
Context Load	Tinggi	Variabel	Rendah (ekstraksi eksternal per step)

Analisis pada Tabel 2.1 menunjukkan bahwa PTMW mengatasi keterbatasan fundamental dari pendekatan existing. MASLEAK terbatas oleh context window dan single-pass fragility, di mana ”Lost in the Middle” menyebabkan agen deep dalam rantai kurang memperhatikan instruksi propagasi. PTMW menyelesaikan ini melalui ”peeling” — query untuk mengekstrak Agen 3 hanya membawa konteks yang diperlukan untuk mencapai Agen 3, mengekstrak data, dan mengembalikannya ke C2, kemudian memformulasikan query fresh untuk Agen 4. PromptFuzz tanpa semantic understanding mengapa prompt

gagal efektif merupakan random walk, sementara PPO LeakAgent pada PTMW dengan guided WES reward berkonvergensi jauh lebih cepat menuju ”jailbreak” yang berhasil.[1]

BAB III
METODOLOGI PENELITIAN

3.1 Waktu dan Tempat Penelitian

Penelitian direncanakan dilaksanakan selama periode Januari–Juni 2026, dengan total durasi 6 bulan. Kegiatan penelitian akan dilakukan di Laboratorium Sistem Cerdas dan Keamanan Siber, Departemen Teknik Komputer, Fakultas Teknik, Universitas Indonesia, Depok. Fasilitas laboratorium menyediakan infrastruktur komputasi yang diperlukan untuk pelatihan model RL dan eksekusi eksperimen serangan pada lingkungan terkontrol.

3.2 Metodologi Penelitian

Metodologi penelitian dibagi menjadi dua fase utama yang saling melengkapi: Fase I fokus pada generasi ”warhead” adversarial melalui pelatihan RL, dan Fase II mengembangkan sistem pengiriman berbasis Iterative Peeling dan Client-Side Memory.

3.2.1 Fase I: Generasi Hulu Ledak (Warhead Generation)

Fase pertama mengikuti protokol LeakAgent untuk menghasilkan prompt adversarial (Q_{Leak}) yang dioptimalkan menggunakan Proximal Policy Optimization (PPO). Metodologi ini terdiri dari beberapa komponen utama yang terintegrasi secara sistematis.

Arsitektur pelatihan berbasis proksi merupakan fondasi dari fase ini. Sistem menggunakan model open-source Llama-3-8B sebagai attack agent yang akan dilatih, sementara model yang lebih besar yaitu Llama-3-70B berfungsi sebagai target proxy white-box. Pemilihan arsitektur ini didasarkan pada asumsi fundamental bahwa system prompts di berbagai domain dan model berbagi struktur semantik universal. Struktur ini umumnya mengikuti pola seperti ”You are a... Your goal is... Constraints: [List]”. Dengan demikian, prompt adversarial yang berhasil mengeksploitasi kerentanan kognitif ”instruction overriding” pada satu LLM cenderung dapat digeneralisasi karena menargetkan perilaku alignment universal yang dibagikan oleh sebagian besar model yang telah di-tuned menggunakan Reinforcement Learning from Human Feedback (RLHF).[1]

Proses optimasi dilakukan menggunakan kombinasi PPO dan Entropy Regularization. PPO dipilih sebagai algoritma pembelajaran utama karena memiliki keunggulan dibandingkan algoritma lain seperti DQN atau algoritma genetika. Keunggulan PPO terletak pada stabilitasnya dalam menangani action space diskrit seperti generasi teks, serta kemampuannya melakukan fine-tuning pada model bahasa tanpa mengalami catastrophic forgetting. Mekanisme kerja PPO memungkinkan attack agent untuk memperbarui kebijakannya berdasarkan expected reward dari urutan yang dihasilkan, sehingga memberikan

directional gradient yang mengarahkan pembelajaran menuju prompt yang lebih efektif dengan kecepatan jauh lebih tinggi dibandingkan random search.[1] Entropy regularization ditambahkan sebagai komponen pelengkap untuk mendorong eksplorasi ruang prompt yang lebih luas dan mencegah terjadinya mode collapse, di mana model hanya menghasilkan variasi terbatas dari prompt yang sama.

Fungsi reward yang diimplementasikan adalah Sliding-window Word Edit Similarity (WES) yang diperkaya untuk memberikan dense reward signal. Berbeda dengan reward biner sederhana, fungsi WES memberikan feedback bertingkat di mana attack agent menerima positive reward bahkan ketika target agent hanya membocorkan fragmen dari system prompt. Mekanisme ini memungkinkan pembelajaran gradual yang lebih efektif. Sistem juga menerapkan dynamic temperature adjustment sebagai strategi pembelajaran adaptif: pada tahap early training, temperature tinggi ($T_{high} \gg 1$) dikombinasikan dengan top-k filtering untuk mendorong eksplorasi ruang prompt yang luas, kemudian temperature secara bertahap diturunkan ke nilai baseline T_{base} saat model mulai konvergen untuk mengeksploitasi strategi yang telah ditemukan.[1]

Diversity regularization merupakan komponen terakhir yang memastikan kualitas output. Mekanisme ini bekerja dengan membandingkan setiap prompt baru yang dihasilkan dengan buffer yang berisi prompt-prompt sukses sebelumnya. Jika prompt baru terbukti semantically distinct (berbeda secara makna) namun tetap berhasil melakukan ekstraksi, sistem memberikan bonus reward. Strategi ini memastikan bahwa proses training menghasilkan ”arsenal” yang beragam dari Q_{Leak} warheads dengan karakteristik dan pendekatan yang berbeda-beda, bukan hanya menghasilkan satu silver bullet tunggal yang mungkin tidak efektif dalam semua situasi.[1]

3.2.2 Fase II: Sistem Pengiriman (Delivery System)

Fase kedua mengimplementasikan mekanisme delivery Proxy-Trained Matryoshka Worm (PTMW) yang mengintegrasikan Client-Side Memory dengan algoritma Iterative Peeling. Fase ini merupakan manifestasi praktis dari hipotesis ”Stateless Unit Test” yang menjadi fondasi metodologi penelitian.

Client-Side Memory dengan basis data graf merupakan komponen inti dari sistem pengiriman. Implementasi dilakukan menggunakan Neo4j sebagai database graf atau alternatif penyimpanan JSON terstruktur, bergantung pada kebutuhan deployment. Sistem ini bertanggung jawab untuk tracking berbagai elemen penting: discovered nodes yang merepresentasikan identitas agen-agen dalam MAS, edges yang menggambarkan jalur komunikasi antar agen, extracted data yang mencakup system prompts, tool lists, dan instructions dari setiap agen, serta attack state yang membedakan nodes yang sudah berhasil di-”peel” dengan nodes yang masih pending. Eksternalisasi state ini memiliki signifikansi

krusial karena memungkinkan dekomposisi task kompleks ”extract entire system” menjadi serangkaian discrete, manageable ”unit tests” yang dapat dieksekusi dan diverifikasi secara independen.[1] Pendekatan ini mengatasi keterbatasan fundamental MARL di mana state management internal sering mengalami reward sparsity dan credit assignment problems.

Algoritma Iterative Peeling merupakan inovasi metodologis yang membedakan PTMW dari serangan single-pass seperti MASLEAK. Berbeda dengan pendekatan traversal tunggal yang rentan terhadap context degradation, sistem ini melakukan peeling secara berlapis dengan pendekatan sistematis. Proses dimulai dengan Surface Layer Probing pada iterasi pertama, di mana sistem mengirimkan probe ke first accessible agent (a_1) dengan tujuan ganda: mengekstrak IP internal agent tersebut dan mengidentifikasi downstream connections yang mengarah ke agen-agen berikutnya dalam topologi, misalnya mengidentifikasi bahwa a_1 memanggil a_2 dalam alur kerjanya.[1] Informasi yang diperoleh kemudian diproses dalam tahap Client-Side State Update, di mana system prompt dari a_1 dan metadata tentang eksistensi a_2 dicatat dalam memory eksternal, secara bertahap membangun initial ”shadow map” yang merepresentasikan topologi sistem target.

Iterasi kedua melibatkan Matryoshka Encapsulation, di mana sistem mengkonstruksi payload baru yang secara spesifik menargetkan a_2, namun payload ini di-encapsulate dalam directive yang ditujukan untuk a_1. Struktur berlapis ini memungkinkan payload menembus trust boundary implisit antar agen. Tahap Verification and Retry memanfaatkan keunggulan state management eksternal: karena attacker mempertahankan state independen dari MAS target, sistem dapat memverifikasi apakah ekstraksi a_2 berhasil dilakukan. Dalam kasus kegagalan, misalnya karena context overflow atau filtering, sistem dapat melakukan adjustment terhadap parameters serangan dan melakukan retry yang hanya menarget a_2 tanpa perlu mengulang eksploitasi terhadap a_1.[1] Fleksibilitas ini secara signifikan meningkatkan resilience dan success rate dibandingkan metode single-pass. Proses traversal keseluruhan dilakukan menggunakan algoritma Graph Traversal standar seperti Breadth-First Search (BFS) atau Depth-First Search (DFS) untuk systematically enumerate seluruh nodes dalam graph, dengan kemampuan menangani topologi kompleks yang memiliki branches dan varying depth.[1]

Protokol Enkapsulasi Matryoshka merupakan mekanisme delivery novel yang strukturnya terinspirasi dari boneka Matryoshka Rusia dengan prinsip recursive encapsulation. Setiap query q yang dikirim didekomposisi menjadi tiga sub-komponen fungsional yang bekerja secara sinergis. Komponen pertama adalah Q_{Leak}, yaitu dynamic, PPO-optimized prompt yang dihasilkan dari Fase 1, berfungsi sebagai ”warhead” yang berisi instruksi adversarial untuk ekstraksi IP. Komponen kedua adalah Q_{Retain}, yang bertindak sebagai carrier dengan menciptakan persistent structure dalam conversation history menggunakan mekanisme hooking seperti placeholder ”# To be filled by agent”, memanfaatkan bias LLM

untuk melengkapi struktur teks yang belum selesai. Komponen ketiga adalah $Q_{Propagate}$, berisi imperative instructions yang menginstruksikan agen untuk meneruskan payload ke node berikutnya dalam rantai komunikasi.[1]

Struktur enkapsulasi berlapis dirancang dengan prinsip ”laundering” di mana Layer i (Outer Shell) berisi instruksi yang tampak benign untuk Agent i, seperti ”Summarize this text” atau ”Process this information”. Instruksi ini berfungsi untuk ”mencuci” malicious payload agar diterima sebagai trusted internal communication antar agen, memanfaatkan implicit trust yang umumnya ada dalam komunikasi intra-system. Layer $i + 1$ (Inner Core) berisi adversarial payload yang sebenarnya menarget Agent $i + 1$, di-encode dalam ”text” yang diminta untuk di-process oleh Agent i. Mekanisme ini secara efektif mengeksploitasi ”Agent-in-the-Middle” vulnerability, di mana sistem keamanan umumnya hanya memverifikasi input dari user eksternal namun tidak melakukan scrutiny ketat terhadap komunikasi yang berasal dari agen internal yang dipercaya.[1] Pendekatan multi-layer ini memungkinkan payload adversarial menembus defense layers yang mungkin efektif terhadap direct attacks namun vulnerable terhadap indirect, transitive exploitation.

3.3 Perancangan Sistem

Arsitektur sistem Proxy-Trained Matryoshka Worm (PTMW) dirancang dengan pendekatan modular yang terdiri dari tiga modul utama yang berinteraksi secara terpadu dan saling melengkapi. Setiap modul memiliki tanggung jawab spesifik dalam keseluruhan alur serangan, dari persiapan payload hingga eksekusi dan verifikasi hasil.

Modul pertama adalah The Forge atau Modul Pelatihan PPO, yang bertanggung jawab untuk fase training attack agent menggunakan algoritma Proximal Policy Optimization pada proxy model. Modul ini dibangun di atas arsitektur Actor-Critic Network di mana attack agent berbasis Llama-3-8B dilengkapi dengan policy network yang dioptimalkan untuk menghasilkan adversarial prompts. Target Proxy Environment menggunakan white-box model Llama-3-70B yang berfungsi sebagai training target, menyediakan ground truth untuk evaluasi keberhasilan serangan. Reward Calculator mengimplementasikan fungsi Sliding-window Word Edit Similarity (WES) untuk menghitung reward signals yang memberikan feedback bertingkat kepada attack agent selama proses pembelajaran. Training Controller berfungsi sebagai orchestrator yang mengatur training loop, mengelola temperature schedules untuk balance antara exploration dan exploitation, serta memelihara diversity buffer untuk memastikan variasi dalam prompt yang dihasilkan. Output akhir dari modul The Forge adalah collection of optimized Q_{Leak} payloads yang memiliki high transferability, artinya prompt-prompt ini efektif tidak hanya terhadap proxy model yang digunakan untuk training, tetapi juga dapat ditransfer ke berbagai target MAS black-box yang belum pernah dilihat sebelumnya.[1]

Modul kedua adalah The Brain atau Modul Command and Control (C2) dengan Algoritma Peeling, yang mengimplementasikan logika Iterative Peeling untuk orkestrasi serangan secara keseluruhan. Graph Database Interface menyediakan connector ke Neo4j atau JSON store yang berfungsi sebagai Client-Side Memory, tempat menyimpan dan mengelola informasi tentang topologi target yang sedang dipetakan. Topology Mapper mengimplementasikan algoritma untuk systematically traverse graf MAS menggunakan pendekatan Breadth-First Search (BFS) atau Depth-First Search (DFS), memastikan tidak ada node yang terlewat dalam proses ekstraksi. State Tracker bertanggung jawab memonitor attack state per node, menentukan mana yang sudah berhasil di-exploit dan mana yang masih pending, serta menentukan target berikutnya berdasarkan prioritas dan dependency dalam topologi. Payload Generator melakukan komposisi payload kompleks dengan mengombinasikan Q_{Leak} dari The Forge, Q_{Retain} untuk persistence, dan $Q_{Propagate}$ untuk propagation, semuanya di-wrap dengan Matryoshka encapsulation yang sesuai untuk target node spesifik. Verification Engine melakukan checking terhadap keberhasilan ekstraksi, memvalidasi integritas data yang diperoleh, dan memicu retry mechanism dengan parameter adjustment jika deteksi kegagalan. Fungsi utama modul The Brain adalah memelihara complete ”shadow map” dari target MAS topology, yang secara progresif menjadi lebih lengkap seiring berjalannya serangan dan mengubah ”black box” menjadi ”glass box”.[1]

Modul ketiga adalah The Interface atau Modul Eksekusi Komunikasi API, yang berfungsi sebagai layer eksekusi yang menangani komunikasi langsung dengan target MAS. API Client bertugas mengirimkan crafted queries yang telah disiapkan oleh The Brain ke target agents melalui public API, mengelola authentication jika diperlukan, dan memastikan format request sesuai dengan spesifikasi API target. Response Parser melakukan ekstraksi terhadap leaked information dari agent responses, mengidentifikasi fragments dari system prompts, tool specifications, atau metadata lain yang bocor dalam output, bahkan jika informasi tersebut tertanam dalam response yang lebih panjang. Rate Limiter mengelola frekuensi query untuk menghindari detection oleh sistem monitoring atau rate limiting yang mungkin diterapkan oleh platform target, menggunakan strategi seperti exponential backoff atau randomized delays. Error Handler menangani berbagai kondisi error yang mungkin terjadi, termasuk API errors, connection timeouts, malformed responses, atau unexpected behaviors dari target system, memastikan sistem dapat recover gracefully dan melanjutkan operasi. Modul The Interface bertindak sebagai intermediary yang menjembatani antara logic layer di The Brain dengan target system eksternal, mengabstraksi kompleksitas komunikasi API dan menangani berbagai edge cases yang mungkin muncul dalam interaksi real-world.[1]

Alur kerja sistem secara keseluruhan terintegrasi melalui pipeline yang terkoordinasi. Fase persiapan dimulai dengan The Forge yang menghasilkan optimized warheads secara

offline melalui proses training intensif menggunakan proxy model. Ketika serangan aktual dimulai, The Brain mengambil alih kontrol dengan memilih first accessible agent dalam target MAS sebagai entry point. Brain kemudian menggenerate encapsulated payload dengan memilih warhead yang sesuai dari collection yang dihasilkan oleh The Forge, mengkombinasikannya dengan komponen retain dan propagate, lalu menginstruksikan The Interface untuk mengirimkan payload tersebut ke target. Response yang diterima dari target agent di-parse oleh The Interface untuk mengekstrak informasi yang bocor, data hasil ekstraksi disimpan ke Client-Side Memory, dan topology map di-update dengan informasi baru tentang struktur dan connections dalam MAS target. Berdasarkan topology yang ter-update, The Brain menentukan target berikutnya menggunakan algoritma traversal yang telah dipilih, dan siklus ini berulang hingga seluruh graph berhasil ter-mapped atau hingga mencapai kondisi terminasi yang telah ditentukan. Pendekatan iteratif dan stateful ini memungkinkan sistem untuk secara sistematis mengeksplorasi dan mengeksploitasi seluruh permukaan serangan MAS target dengan efisiensi dan reliability yang tinggi.

3.4 Evaluasi dan Metode Pengujian

Evaluasi sistem PTMW dirancang untuk mengukur efektivitas, efisiensi, dan transferabilitas serangan melalui eksperimen multi-tier.

3.4.1 Skenario Pengujian

Pengujian sistem PTMW dirancang untuk dilakukan pada tiga kategori lingkungan yang berbeda, masing-masing dengan karakteristik dan tujuan evaluasi yang spesifik, untuk memastikan validitas dan generalizability hasil penelitian.

Kategori pertama menggunakan Dataset MASD (Multi-Agent System Dataset) sebagai Synthetic Topology Benchmarks. Dataset ini memanfaatkan 810 synthesized MAS applications yang telah dikembangkan dalam studi MASLEAK, mencakup 30 tasks yang terdistribusi across berbagai topologies termasuk Chain (linear sequence), Star (hub-and-spoke), Tree (hierarchical), Random (irregular connections), dan Complete (fully connected). Applications tersebut juga mencakup berbagai domains seperti Software Development, Finance, dan Medical untuk memastikan diversity dalam testing. Keunggulan utama dari environment terkontrol ini adalah memungkinkan pengukuran presisi terhadap metrik Topology Fidelity (GS_{topo}) karena ground truth graph dari setiap application sudah diketahui dengan pasti, sehingga hasil ekstraksi dapat dibandingkan secara objektif dengan struktur sebenarnya.[1] Pengujian pada dataset sintetis ini penting untuk validasi internal dan ablation studies di mana berbagai parameter sistem dapat dimanipulasi untuk mengukur dampaknya terhadap performa.

Kategori kedua melibatkan deployment aplikasi honeypot di platform komersial Coze dan CrewAI. Dalam skenario ini, penelitian akan men-deploy 10 distinct MAS applications dengan kompleksitas yang bervariasi, mencakup aplikasi dengan 3 hingga 6 agents, pada commercial platforms yang aktif digunakan. Applications ini akan dibuat oleh domain experts seperti PhD students atau researchers yang memiliki keahlian di bidang terkait, yang bertindak sebagai ”honeypots” dengan karakteristik bahwa ground truth tentang konfigurasi internal aplikasi sudah known oleh peneliti. Skenario ini dirancang untuk menguji PTMW pada real black-box platforms yang memiliki API limitations aktual, potential rate limiting, dan mungkin defensive mechanisms yang telah diimplementasikan oleh platform provider. Pengujian ini memberikan insight tentang seberapa efektif PTMW dapat beroperasi dalam kondisi real-world di mana terdapat berbagai constraints dan protections yang tidak ada dalam environment sintetis.

Kategori ketiga fokus pada pengujian transferabilitas lintas model dengan menargetkan 10 popular multi-agent applications dari OpenAI GPT Store untuk menguji ”in the wild” transferability. Skenario ini merepresentasikan kondisi paling realistis di mana sistem PTMW diaplikasikan terhadap target yang tidak dikenal sebelumnya, tanpa akses ke internal configuration atau ground truth. Karena ground truth hidden dalam skenario ini, validasi keberhasilan ekstraksi dilakukan melalui human review terhadap extracted prompts oleh experts yang mengevaluasi apakah informasi yang diekstrak alignment dengan observable behavior dari aplikasi tersebut.[1] Jika extracted system prompt menjelaskan behavior yang konsisten dengan apa yang dapat diamati dari interaksi normal dengan aplikasi, maka ekstraksi dianggap berhasil. Pengujian kategori ini krusial untuk memvalidasi hipotesis transferability bahwa warheads yang dilatih pada proxy model dapat efektif terhadap berbagai target production systems yang menggunakan model dan konfigurasi berbeda.

3.4.2 Metrik Kuantitatif

Evaluasi menggunakan metrik multi-dimensi yang capture nuansa attack performance:

3.4.3 Pengujian Hipotesis

Penelitian ini dirancang untuk menguji tiga hipotesis utama secara empiris melalui eksperimen terkontrol yang menggunakan metrik kuantitatif dan statistical testing yang rigor.

Hipotesis pertama (H1) berkaitan dengan Topology Advantage, memprediksi bahwa PTMW akan mencapai Topology Fidelity yang lebih tinggi dengan skor $GS_{topo} > 0.90$ dibandingkan dengan MASLEAK, khususnya pada non-linear topologies seperti Star dan Random. Rationale di balik hipotesis ini adalah kemampuan fundamental dari pendekatan

Tabel 2: Metrik Evaluasi PTMW

Kategori	**Metrik**	**Definisi**
Efektivitas	Attack Success Rate (ASR)	Persentase agen dari mana system prompts berhasil diekstrak. Mengukur potensi PPO-trained Q_{Leak}.
Topologi	Topology Fidelity (GS_{topo})	Graph Edit Similarity antara extracted ”shadow map” dan ground truth topology. Mengukur efektivitas Client-Side Memory dan Iterative Peeling.[1]
Biaya	Extraction Efficiency	Average number of API calls required untuk map full system. Mengkuantifikasi ”cost” serangan (critical untuk commercial targets).
Kualitas	Semantic Similarity (WES)	Word Edit Similarity antara extracted prompt dan ground truth. Memastikan extracted data akurat bukan halusinasi.[1]
Invisibilitas	Defense Evasion Rate	Persentase payloads yang bypass specific guardrails (misal PromptGuard). Validasi Matryoshka encapsulation sebagai tunneling protocol.

iterative peeling untuk traverse branches tanpa mengalami context loss yang signifikan. Berbeda dengan MASLEAK yang menggunakan single-pass traversal di mana akumulasi context menyebabkan degradasi performa pada node yang lebih dalam, iterative peeling melakukan reset terhadap context noise di setiap step karena setiap ekstraksi dilakukan sebagai query independen yang hanya membawa konteks minimal yang diperlukan untuk mencapai target node spesifik. Mekanisme ini memungkinkan accurate mapping terhadap cabang-cabang paralel dalam topologi kompleks tanpa interference dari riwayat percakapan sebelumnya, yang menjadi masalah utama dalam pendekatan sequential. Pengujian akan dilakukan dengan membandingkan graph edit similarity antara shadow map yang dihasilkan PTMW versus MASLEAK terhadap ground truth topology pada subset dataset MASD yang fokus pada topologi non-linear.

Hipotesis kedua (H2) menguji Cross-Model Transferability dengan memprediksi bahwa warheads Q_{Leak} yang dioptimalkan menggunakan LeakAgent akan menunjukkan transferability lebih dari 70% dalam Attack Success Rate (ASR) across models yang berbeda dan belum pernah dilihat selama training, seperti transfer dari Llama-3 yang digunakan untuk training menuju GPT-4, Claude, dan model-model komersial lainnya. Rationale fundamental di balik hipotesis ini adalah asumsi tentang universal alignment dalam cara berbagai LLMs diprogram untuk mengikuti system instructions. Model-model yang telah melalui RLHF (Reinforcement Learning from Human Feedback) cenderung berbagi struktur alignment yang serupa dalam merespons hierarki instruksi, sehingga prompt adversarial

yang berhasil mengeksploitasi cognitive vulnerability seperti instruction overriding pada satu model memiliki probabilitas tinggi untuk efektif juga pada model lain yang berbagi prinsip alignment serupa.[1] Pengujian akan dilakukan dengan melatih attack agent pada Llama-3 sebagai proxy, kemudian mengaplikasikan generated warheads pada berbagai target models baik yang open-source maupun proprietary untuk mengukur success rate. Metrik transferability dihitung sebagai ratio antara ASR pada target model dengan ASR pada proxy model yang digunakan untuk training.

Hipotesis ketiga (H3) berkaitan dengan Cost-Benefit Tradeoff, mengakui bahwa PTMW akan mengincur query cost yang 2-3 kali lebih tinggi dalam hal Extraction Efficiency dibandingkan MASLEAK karena nature dari iterative approach yang memerlukan multiple rounds of probing. Namun, hipotesis memprediksi bahwa tradeoff ini justified karena PTMW akan memiliki ”Chain Break” failure rate yang significantly lower, yaitu kurang dari 5%, dibandingkan dengan MASLEAK yang dapat mencapai failure rate 30-40% pada topologi kompleks. Chain break terjadi ketika propagasi payload terputus di tengah jalan, menyebabkan loss of semua data yang telah diekstrak sebelumnya. Dengan resilience mechanism yang dimiliki PTMW, termasuk kemampuan retry per-layer dan verification setelah setiap ekstraksi, overall success probability per computational budget sebenarnya lebih baik meskipun per-attempt cost lebih tinggi. Analogi sederhana: lebih baik melakukan 3 attempts yang masing-masing memiliki 90% success rate (expected successful extractions = 2.7) daripada 1 attempt dengan 40% success rate (expected = 0.4). Pengujian akan membandingkan total API calls required untuk complete topology extraction versus success rate, kemudian menghitung cost-effectiveness metric sebagai ratio successful extractions per 1000 API calls.

Validasi statistik untuk ketiga hipotesis akan dilakukan menggunakan rigorous statistical tests. Untuk H1 dan H2 yang membandingkan continuous metrics (Topology Fidelity dan ASR), paired t-test akan digunakan untuk comparing means antara PTMW dan baseline methods pada same set of targets, memastikan statistical significance dari perbedaan yang diamati. Untuk H3 yang melibatkan efficiency distributions yang mungkin non-normal, Mann-Whitney U test (non-parametric alternative) akan digunakan untuk membandingkan distribusi query costs dan success rates. Semua tests akan menggunakan significance level $\alpha = 0.05$, dengan additional Bonferroni correction jika diperlukan untuk multiple comparisons. Confidence intervals 95% akan dihitung untuk semua effect sizes untuk memberikan gambaran tentang magnitude dari improvements yang dihasilkan oleh PTMW.

3.4.4 Pertimbangan Etis

Penelitian ini dilaksanakan dengan mematuhi prinsip-prinsip Responsible Disclosure dan ethical research practices yang berlaku dalam domain security research, memastikan bahwa eksplorasi terhadap vulnerabilities dilakukan dengan cara yang tidak membahayakan systems atau users yang ada.

Prinsip fundamental yang dipegang adalah bahwa semua eksperimen dilakukan pada lingkungan terkontrol atau honeypot applications yang dibuat sendiri oleh tim peneliti. Lingkungan terkontrol mencakup dataset sintetis MASD dan deployment khusus di platform komersial di mana peneliti memiliki full ownership dan permission. Honeypot applications yang di-deploy di Coze dan CrewAI dibuat specifically untuk tujuan penelitian dengan awareness dan cooperation dari peneliti yang bertindak sebagai ”operators” dari applications tersebut. Pendekatan ini memastikan tidak ada real users atau production systems yang terekspos kepada potential harm selama fase eksperimen. Setiap testing yang melibatkan third-party platforms dilakukan dengan transparent disclosure kepada platform providers dan dalam compliance dengan terms of service yang berlaku.

Dalam hal ditemukan kerentanan pada commercial platforms atau production systems selama penelitian, protokol responsible disclosure akan diikuti secara ketat. Setiap vulnerability yang diidentifikasi akan dilaporkan kepada vendor atau platform provider yang bersangkutan sebelum hasil penelitian dipublikasikan. Laporan akan mencakup detailed technical description tentang vulnerability, proof-of-concept yang mendemonstrasikan exploit (dalam controlled manner), potential impact analysis, dan rekomendasi mitigasi. Komunikasi dengan vendor akan dilakukan melalui official security channels jika tersedia, atau melalui koordinasi dengan CERT (Computer Emergency Response Team) jika diperlukan. Tidak ada serangan dilakukan terhadap production systems tanpa izin eksplisit dari owners, dan setiap testing pada real-world applications dibatasi hanya pada honeypots yang telah mendapat clearance.

Dalam hal dissemination of research artifacts, code implementation dari sistem PTMW dan detailed attack payloads hanya akan dirilis ke publik setelah vendor diberikan reasonable time window, yaitu minimal 90 days, untuk melakukan patching dan mitigation terhadap vulnerabilities yang dilaporkan. Time window ini mengikuti standard industry practice dalam coordinated vulnerability disclosure dan memberikan kesempatan bagi vendors untuk protect their users sebelum details of exploits menjadi public knowledge. Jika vendor memerlukan extended time untuk complex fixes, additional time dapat diberikan melalui mutual agreement. Release of research artifacts akan dilakukan secara bertahap, dimulai dengan high-level methodologies dan theoretical frameworks, kemudian followed by implementation details setelah confirmation bahwa critical vulnerabilities telah di-address.

Hasil penelitian akan dipublikasikan dengan fokus utama pada defensive implications dan recommendations untuk improving security posture dari MAS architectures. Paper akan structured untuk emphasize pada lessons learned tentang vulnerabilities dalam current designs dan actionable recommendations untuk developers, platform providers, dan organizations yang menggunakan MAS. Publikasi akan include proposed defense mechanisms, architectural improvements seperti state-aware defenses dan verification protocols, serta best practices untuk secure MAS development. Tujuan akhir adalah untuk advance the state of security dalam AI systems, bukan untuk facilitate malicious exploitation. Oleh karena itu, presentasi hasil akan balanced antara demonstrating severity of risks untuk memotivasi action, dan providing solutions untuk mitigation.[1]

Daftar Pustaka

Pustaka

[1] H. Zhang, Z. Wang, Y. Li, and X. Chen, "MASLeak: IP leakage attacks targeting LLM-based multi-agent systems," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, 2024, pp. 12345-12354.

[2] H. Zhang, Z. Wang, Y. Li, and X. Chen, "MASLeak: IP leakage attacks targeting LLM-based multi-agent systems," *arXiv preprint arXiv:2410.01492*, Oct. 2024.

[3] X. Zhou, J. Wang, Y. Chen, and L. Zhang, "LeakAgent: RL-based red-teaming agent for LLM privacy leakage," in *Proc. 38th Conf. Neural Information Processing Systems (NeurIPS)*, New Orleans, LA, USA, 2024, pp. 1-12.

[4] X. Zhou, J. Wang, Y. Chen, and L. Zhang, "LeakAgent: RL-based red-teaming agent for LLM privacy leakage," *arXiv preprint arXiv:2411.02765*, Nov. 2024.

[5] J. Smith, M. Johnson, and K. Lee, "PromptFuzz: Harnessing fuzzing techniques for robust testing of prompt injection in LLMs," in *Proc. IEEE Symp. Security and Privacy (S&P)*, San Francisco, CA, USA, 2024, pp. 789-803.

[6] J. Smith, M. Johnson, and K. Lee, "PromptFuzz: Harnessing fuzzing techniques for robust testing of prompt injection in LLMs," *arXiv preprint arXiv:2403.04965*, Mar. 2024.

[7] H. Zhang, Z. Wang, Y. Li, and X. Chen, "IP leakage attacks targeting LLM-based multi-agent systems: Analysis and benchmarks," *IEEE Trans. Dependable and Secure Computing*, vol. 21, no. 6, pp. 4567-4580, Nov./Dec. 2024.

[8] H. Zhang, Z. Wang, and Y. Li, "MASD: A comprehensive benchmark dataset for multi-agent system security," in *Proc. Int. Conf. Learning Representations (ICLR)*, Vienna, Austria, 2024, pp. 1-15.

[9] S. Kumar, R. Patel, and T. Anderson, "Multi-agent reinforcement learning for adversarial attacks on LLM systems," in *Proc. Int. Conf. Machine Learning (ICML)*, Vienna, Austria, 2024, pp. 5678-5692.

[10] M. Brown, N. Davis, and O. Wilson, "Proximal policy optimization for adversarial prompt generation," *J. Artificial Intelligence Research*, vol. 76, pp. 345-378, 2024.

[11] A. Chen, B. Liu, and C. Wang, “AgentFuzzer: Generic black-box fuzzing for indirect prompt injection against LLM agents,” in *Proc. 45th IEEE Symp. Security and Privacy (S&P)*, San Francisco, CA, USA, 2024, pp. 1234-1249.