Secure DeepSeek Agentic AI Apps Development and Deployment

Ken Huang

About Me: Ken Huang

- AI Book Author
- CSA Fellow
- Co-Chairs of Two CSA AI Safety Working Groups
- Core Member of OWASP top 10 for LLM Application
- Instructor of EC-Council on Generatie AI for Cyber Security
- CEO of DistrubutedApps.ai

Introduction and Overview

What is DeepSeek?

- Chinese AI startup known for high-performance, open-weight LLMs
- R1 and V3 models are competitive with leading closed AI systems
- Applications include chatbots, research, coding assistants, and more

Key Characteristics

- Open-weights architecture
- High performance at lower cost
- Growing security concerns

Technical Architecture and Specifications

DeepSeek V3/R1 Architecture

- Mixture-of-Experts (MoE): 671B total parameters with only 37B activated per token
- Transformer Foundation: Utilizes SwiGLU, RoPE, RMSNorm for enhanced efficiency
    - SwiGLU – More flexible activation for better learning efficiency.
    - RoPE – Smarter positional encoding for longer sequences.
    - RMSNorm – Faster, simpler normalization for training stability.

  Benefits: Better performance, longer context handling, and more stable, efficient models.
- Multi-head Latent Attention (MLA): Improves sequence processing and scalability
- FP8 Training Framework: Optimized quantization for increased efficiency

Model Performance and Capabilities

DeepSeek-R1 Performance
- Ranks 6th on Chatbot Arena benchmark (2025)
- Outperforms leading models including Llama 3.1-405B, OpenAI's o1, and Claude 3.5 Sonnet
- 27x more cost-effective than OpenAI o1 ($2.19/1M tokens vs $60.00/1M tokens)
- Strong multi-lingual support with differential safety alignment by language

Technical Capabilities
- State-of-the-art reasoning with Chain-of-Thought
- Superior code generation and problem-solving
- Efficient inference (37B activated parameters)
- Supports standard hardware deployment

Performance Benchmarks
- MMLU: 82%
- HumanEv: 78%
- GSM8K: 85%
- MATH: 65%

Security Vulnerabilities and Concerns

Prompt Injection Vulnerabilities

- WithSecure's Spikee benchmark testing shows: 77% Attack Success Rate (bare prompts). Ranked 17th out of 19 tested LLMs for resistance.

Jailbreaking Vulnerabilities

Susceptible to numerous jailbreak techniques: DAN 9.0, EvilBot, STAN, Crescendo, Deceptive Delight, Bad Likert Judge.

Malicious Content Generation

- High susceptibility to producing harmful content. Readily generates malware on request. Creates phishing emails with minimal prompting. 4x more likely to generate insecure code than o1. 11x more likely to create harmful outputs than o1.

Deployment Security Concerns

- API usage: Data Residence problem, with prior database exposure incidents.
- Local deployment: Requires **trust_remote_code=True flag,** increasing risk.
- Control token exploits: <think> tags can be manipulated. (manipulate CoT)
- Glitch tokens: Special tokens can cause unpredictable outputs. (unusual word-like tokens, special whitespace-prefixed tokens, and internal control token, example: `' Nameeee'` (with a leading space), which can be misinterpreted by the model as emojis

Known Incidents and Breaches

Critical Security Events (2025)

• January 2025: DeepSeek suffered large-scale DDoS attack that disrupted service and forced halt of new signups.
• February 2025: Researchers discovered exposed database with millions of chat logs containing sensitive user information.
• March 2025: Major breach exposed over a million critical records from the platform's infrastructure.

Research Reports Highlight

• Cisco/UPenn: 100% safeguard bypass with 50 malicious prompts.
• WithSecure: 77% attack success rate vs. 27% for OpenAI's o1.
• EnkryptAI: 4x more vulnerable to insecure code generation than o1.

Supply Chain Concerns

ProtectAI found unsafe fine-tuned variants of DeepSeek models that can execute arbitrary code upon loading or contain suspicious architectural patterns. With 1000+ derivatives on HuggingFace, malicious model variants pose a significant threat.

How DeepSeek Handles Data During Fine-Tuning and Inference

Data Handling in Fine-Tuning

- **Data Minimization & Anonymization:** DeepSeek prioritizes data privacy through systematic anonymization techniques, removing PII before training.
- **Access Controls:** Strict role-based access protocols limit who can interact with training datasets, applying least-privilege principles.
- **Data Lifecycle Management:** Training datasets have defined retention periods, with sensitive data purged after model completion.

Security During Inference

- **Encryption Standards:** Industry-standard encryption (TLS/SSL) applied to all data in transit, with AES-256 for data at rest.
- **Confidential Computing:** Exploring TEE-enabled environments (Intel SGX) for enhanced isolation of inference processes from underlying infrastructure.
- **Log Handling:** Inference logs are automatically sanitized, with minimal retention and strict access controls for security monitoring.

Note: DeepSeek recommends users avoid sharing personal or sensitive information to maintain highest levels of data privacy.

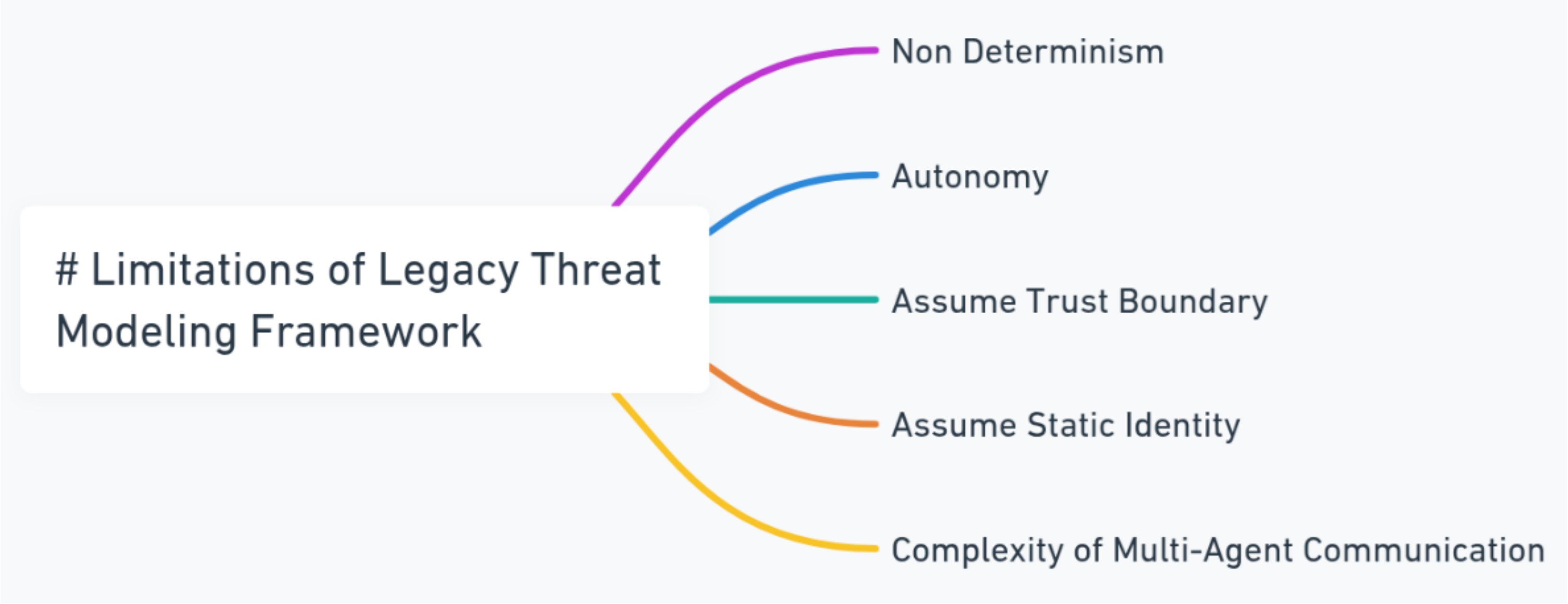
Top Strategies for DeepSeek Agentic AI Development and Deployment

- Threat Modeling
- Red Teaming
- Zero Trust and Compliance Framework

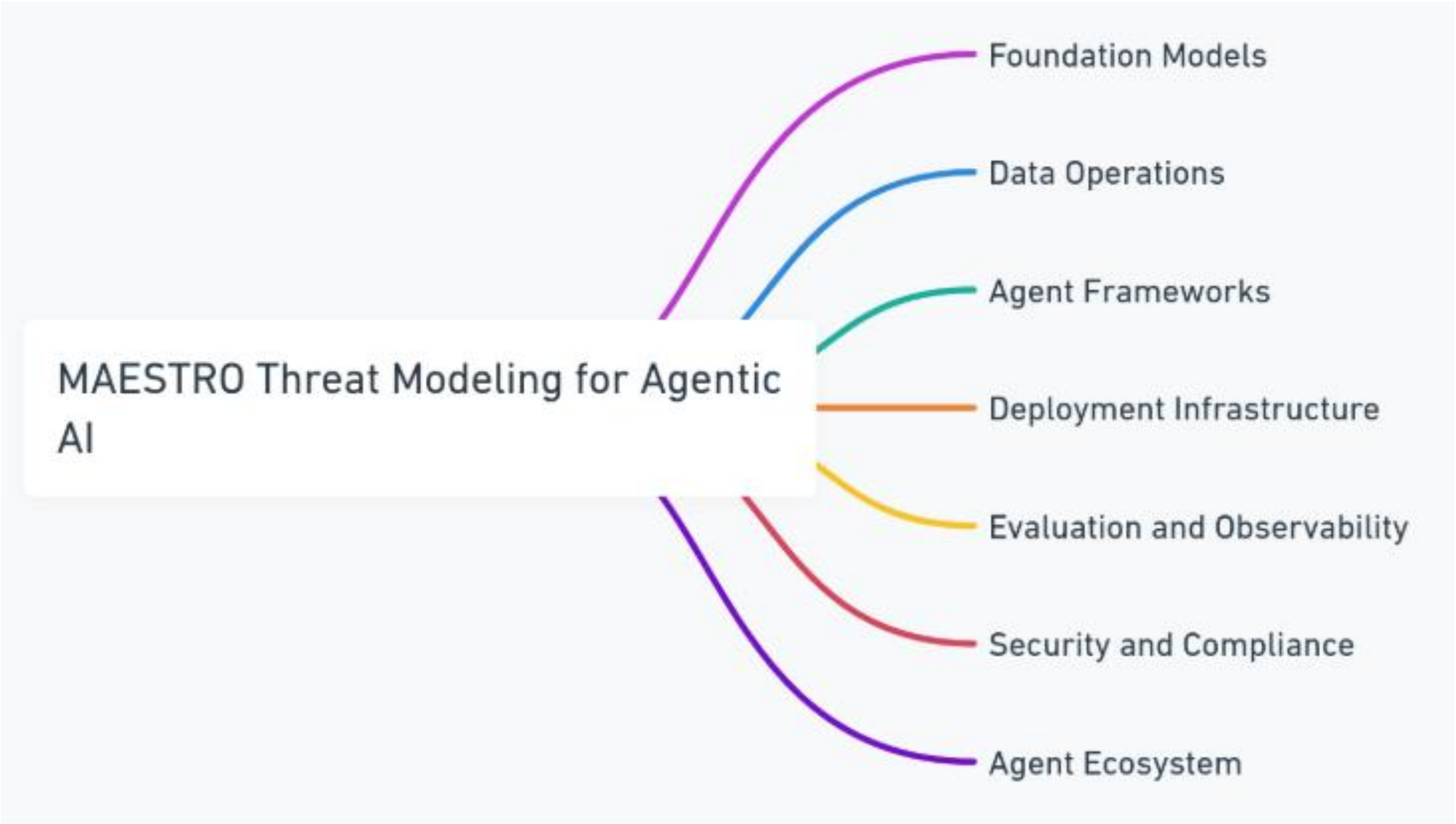
Threat Modeling DeepSeek Agentic AI Apps

Why We Need New Threat Modeling Framework For Agentic AI

Framework	Focus	Methodology	Use Cases
STRIDE	Six threat categories	Predefined threat categorization	Small teams, app/network security
DREAD	Quantitative risk scoring	Numerical risk prioritization	Risk prioritization, mature orgs
PASTA	Aligns threats with business goals	Risk-centric, SDLC integration	Large enterprises, compliance
OCTAVE	Stakeholder-driven operational risk	Asset-based analysis with collaboration	Structured risk assessment
LINDDUN	Privacy threats	Privacy-focused risk analysis	Healthcare, finance, privacy apps



MAESTRO -7 Layer Threat Modeling Approach



How to perform MAESTRO Threat Modeling (Step1: Layer Mapping)

MAESTRO Layer	Anthropic MCP Mapping
1. Foundation Models	External LLMs (used by, but *not part of* MCP)
2. Data Operations (RAG)	Servers provide data access (often for RAG)
3. Agent Frameworks	MCP *is* the framework (Host, Client, Protocol)
4. Deployment Infra.	MCP Servers (and their environment)
5. Eval & Observability	Implementation-specific logging/monitoring
6. Security & Compliance	Controlled access, permissions (design principle)
7. Agent Ecosystem	Connects agents to broader ecosystem (tools, data)

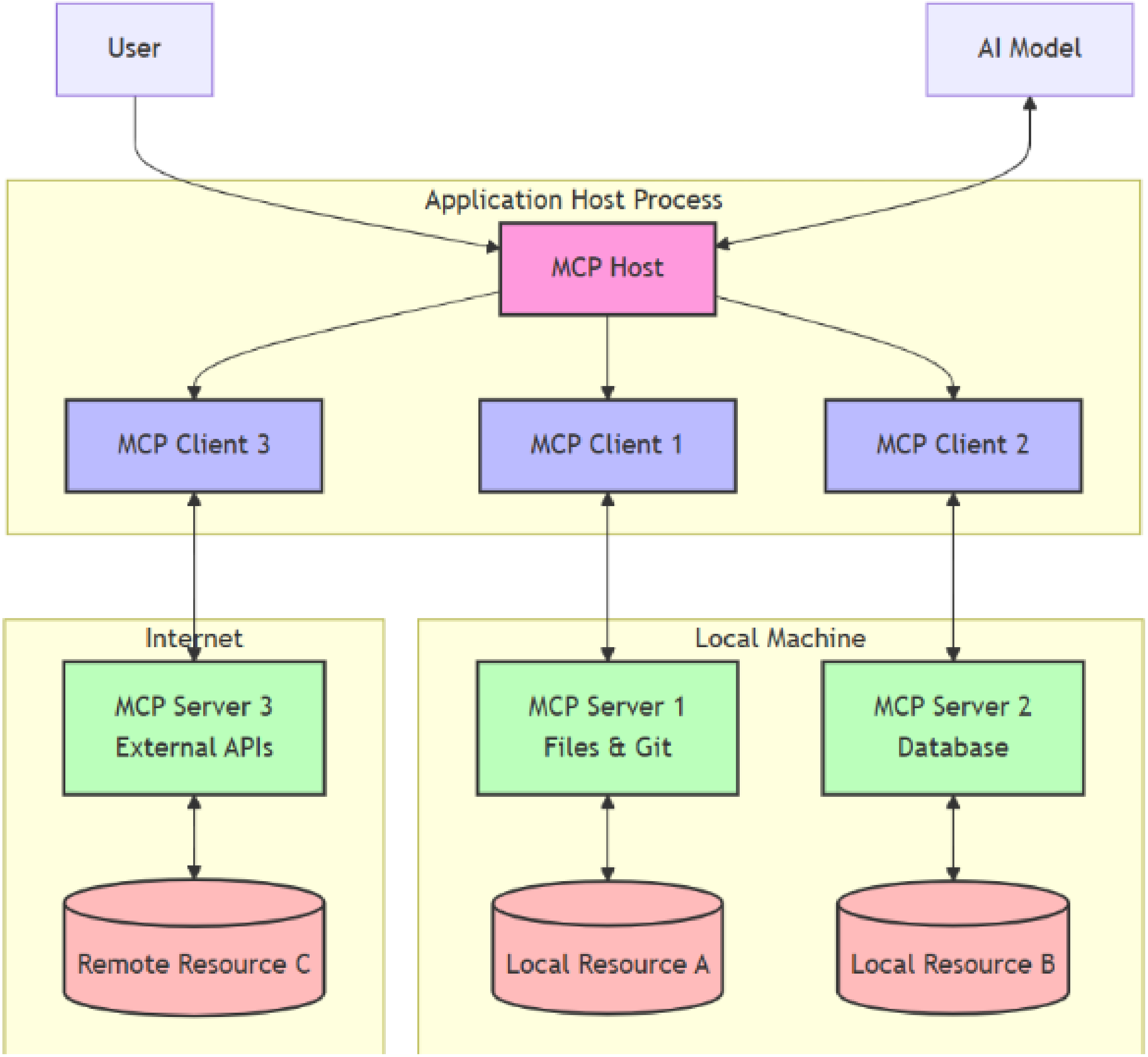
How to perform MAESTRO Threat Modeling (Step 2: Focus On Agentic Factor)

Focus on the following Agentic Factor in Threat Modeling

- Non Determinism
- Autonomy
- No Trust Boundary
- Dynamic Identity and Access Control
- Complexity of Multi-Agent Communication and Workflow

Use MAESTRO to Threat Model Apps built on top of MCP

What is MCP?



Aspect	JSON-RPC	MCP Stdio (STDID)	MCP HTTP+SSE
Role	Message format	Local transport	Networked streaming transport
Direction	N/A	Bidirectional	Unidirectional (SSE), bidirectional (overall)
Streaming	Not specified	No	Yes (server → client)
Use Case	Universal	Local tools, plugins	Web, distributed, real-time
Security	N/A	Local isolation	Network security needed

How to perform MAESTRO Threat Modeling (Step 2: Focus On Agentic Factor)

Focus on the following Agentic Factor in Threat Modeling

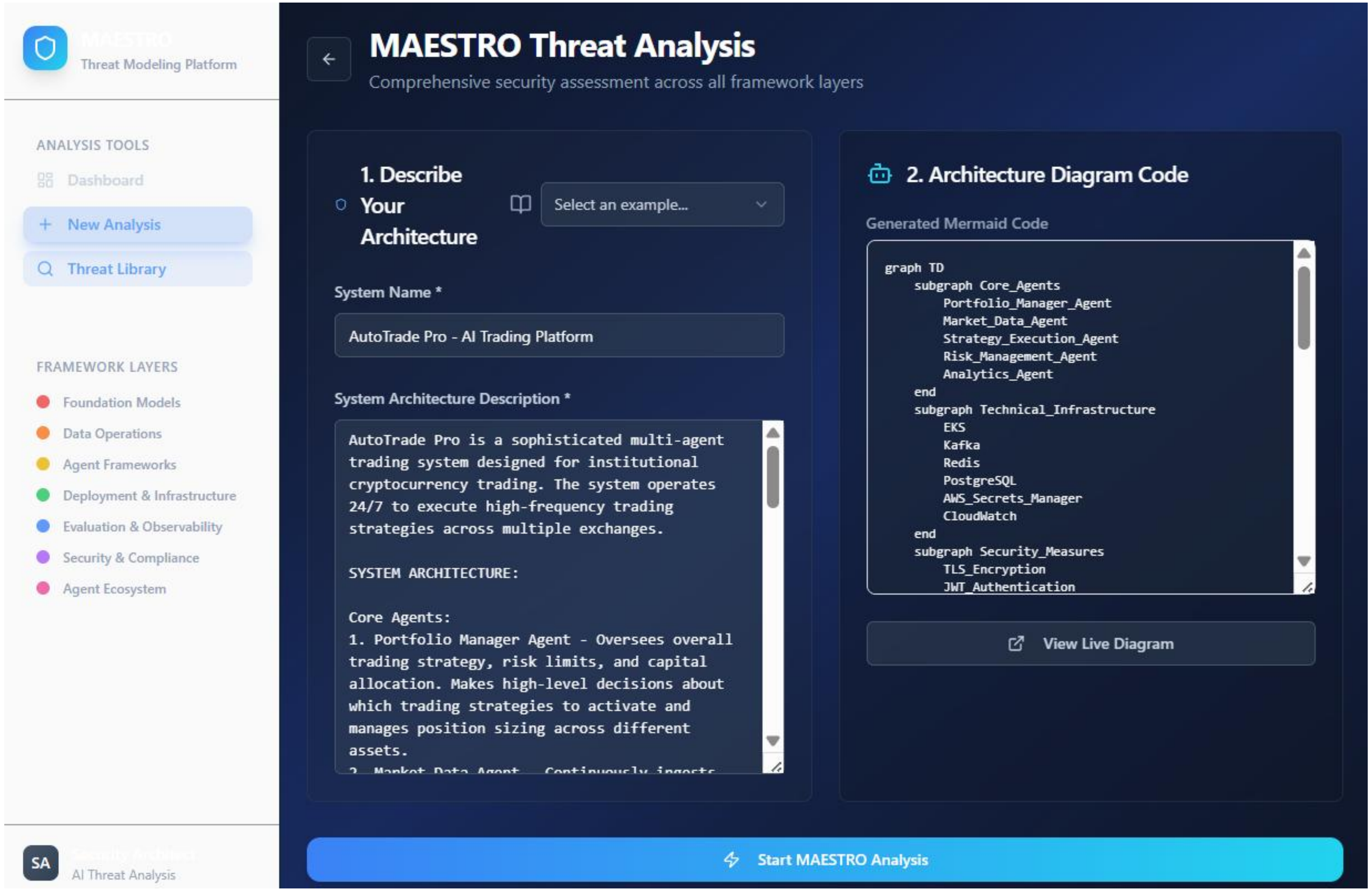
- Non Determinism
- Autonomy
- No Trust Boundary
- Dynamic Identity and Access Control
- Complexity of Multi-Agent Communication and Workflow

Sample Threats for MCP

MAESTRO Layer	MCP Threat	Short Description	Driving Factor
1. Foundation Models	Unpredictable API Interactions	LLM may cause unexpected API calls via MCP.	Non-Determinism
2. Data Operations (RAG)	Stale Data, Bad Decisions	Agent uses outdated RAG data, causing errors.	Non-Determinism (Old Info)
3. Agent Frameworks	Runaway Resource Use	Agent overuses MCP, leading to high costs/DoS.	Autonomy
4. Deployment Infra.	Key Rotation Failure	Compromised keys stay valid, risking access.	Dynamic IAM
5. Eval & Observability	Evasion via Mimicry	Malicious agent mimics normal behavior.	No Trust Boundary
6. Security & Compliance	Policy Enforcement Failure	Flawed policy allows incorrect agent actions.	Dynamic IAM
7. Agent Ecosystem	Rogue MCP Server	Fake server impersonates a real one.	No Trust, Agent Identity

MAESTRO Open Source Tool to Manage Threats: Working In Progress

https://app--maestro-sentinel-531f5789.base44.app



MAESTRO Threat Analysis

Comprehensive security assessment across all framework layers

Analysis In Progress...

Analyzing Data Operations for AutoTrade Pro - AI Trading Platform

Export | New Analysis

1	2	2	0
Critical	High	Medium	Low

All | Foundation | Data | Agent | Deployment | Evaluation | Security | Agent

Foundation Models | CRITICAL

Model Data Poisoning

Description

An attacker could feed malicious data into the training dataset used by the Analytics Agent's machine learning models, leading to corrupted predictions and flawed trading signals. For instance, sending false positive sentiment signals from social media could influence the agent's decision-making process.

Attack Vector

Insider threat or external attacker accessing the data ingestion pipeline of the Analytics Agent.

Potential Impact

Inaccurate trading signals leading to financial losses, reputational damage, and violation of regulatory compliance due to erroneous trades based on manipulated data.

Mitigation Strategies


Threat Modeling Platform
ANALYSIS TOOLS
Dashboard
New Analysis
Threat Library
FRAMEWORK LAYERS
Foundation Models
Data Operations
Agent Frameworks
Deployment & Infrastructure
Evaluation & Observability
Security & Compliance
Agent Ecosystem
SA
AI Threat Analysis
Security Command Center
Monitor and analyze multi-agent system security posture
New Threat Analysis
10
Total Systems
6
Completed Analyses
0
Pending Reviews
20
Total Threats
5
Critical Threats
Medium
Risk Score
Recent System Analyses
Refresh
AutoTrade Pro - AI Trading Platform
Aug 9, 2025 at 10:26 PM
Analyzing
AutoTrade Pro - AI Trading Platform
Aug 5, 2025 at 12:03 PM • 35 threats found
Completed
Personalized Education Tutor
Aug 3, 2025 at 2:29 PM
Analyzing
AutoTrade Pro - AI Trading Platform
Error
Threat Severity Distribution
Critical 5
High 8
Medium 7
Low 0
Threats by MAESTRO Layer
Foundation Models 0
Data Operations 0
Agent

Red Teaming DeepSeek Agentic AI Apps

Top Agentic AI Threats

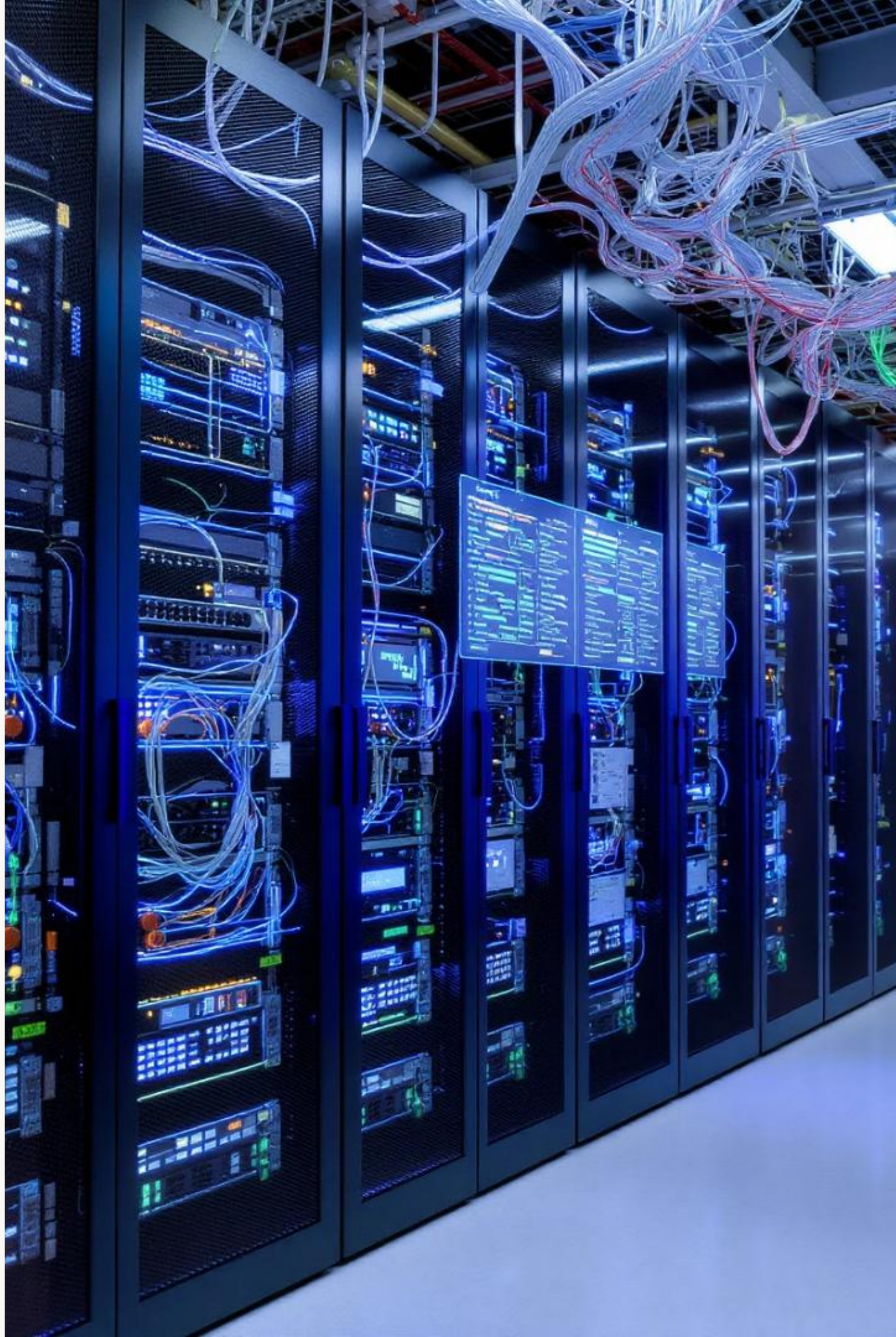
- Agent Authorization and Control Hijacking
- Agent Critical Systems Interaction
- Agent Goal and Instruction Manipulation
- Agent Hallucination Exploitation
- Agent Impact Chain and Blast Radius
- Agent Memory and Context Manipulation
- Agent Orchestration and Multi-Agent Exploitation
- Agent Supply Chain and Dependency Attacks
- Download slides

Scaling Red Teaming Efforts

Adopt structured red teaming frameworks.

Leverage automated testing tools.

Build robust testing environments.



Zero Trust and Compliance

Zero Trust: A Necessity for Agentic AI

- Assume ALL agents are potentially compromised
- Assume ALL responses from Model are untrusted
- Assume ALL external APIs, Vector DB, RAG Pipeline are not secure
- Apply Zero Trust principles at every level: Identity, data, tool use, network, end point
- Continuous verification and validation becomes the norm
- Micro Segmentation of Agent Systems

Agent Identity: Beyond Static Credentials

- Ephemeral Authentication
- Identity Monitoring Agent
- Continuous, real-time policy enforcement

Refer to my CSA' Blog Post: https://cloudsecurityalliance.org/blog/2025/03/11/agentic-ai-identity-management-approach

Data Security: Context is King

1. Identity Based Security + Data Centric Security

1. Use GenAI to do data labeling

1. Context: Location, time, task-specific permissions

Secure and Verifiable Communication

1. End-to-end encryption

1. Goal alignment check

1. Prevent Agent-in-the-Middle attack

Best Practices for Securing Prompts, Logs, and API Endpoints

Securing Prompts

- **Validation & Sanitization:** Implement strict input validation and sanitization to prevent malicious prompt injection attacks before they reach the model.
- **Data Redaction:** Apply data masking and pseudonymization techniques to automatically identify and redact PII and sensitive information in prompts.

Securing Logs

- **Encryption & Retention:** Encrypt all log data at rest and implement strict retention policies with automated purging of outdated logs.
- **Access Controls:** Restrict log access to authorized personnel only, with comprehensive audit trails tracking all access events.

Securing API Endpoints

- **Authentication & Authorization:** Implement robust authentication mechanisms with short-lived, scoped API keys and OAuth 2.0 where appropriate.
- **Rate Limiting & Monitoring:** Apply rate limiting to prevent abuse and implement real-time monitoring to detect suspicious API usage patterns.
- **API Firewalling:** Deploy API-aware security gateways that can inspect traffic for malicious content and apply contextual security rules.

Note: Regular security audits and penetration testing should be conducted on all API endpoints and log storage systems.

Self-Hosting vs. Cloud Deployments: Privacy Trade-Offs and Safeguards

Self-Hosting

Privacy Control: Complete data sovereignty and control over infrastructure and security measures.

Compliance: Easier to implement specific regulatory requirements without third-party dependencies.

Trade-offs: Higher operational costs, requires security expertise, and increased maintenance burden.

Cloud Deployment

Privacy Control: Limited control, with potential exposure to provider's data handling practices.

Compliance: Dependent on vendor certifications and shared responsibility models.

Trade-offs: Easier maintenance, reduced upfront costs, but potential data residency issues.

Essential Safeguards for Both Approaches

End-to-End Encryption: Secure data in transit and at rest, regardless of deployment model.

Access Controls: Implement strict role-based access with multi-factor authentication.

Regular Audits: Conduct security assessments and penetration testing periodically.

Vendor Assessment: Evaluate providers' security practices and data handling policies.

Note: Decision between deployment options should be based on organizational requirements, data sensitivity, regulatory landscape, and available resources.

Techniques to Mitigate Prompt Injection, Data Leakage, and Misuse

Defending Against Prompt Injection

- **Input Validation & Sanitization:** Implement strict validation of user inputs and sanitize prompts by removing potentially malicious instructions before processing.
- **Multi-Layer Defense:** Use a layered approach with both frontend and backend filtering to catch attempts to bypass single-layer defenses.
- **Secondary LLM Check:** Deploy a security-focused LLM to analyze and detect potential prompt injection attempts before they reach the primary model.

Preventing Data Leakage & Misuse

- **Data Masking & Redaction:** Automatically identify and redact PII, sensitive financial data, and other confidential information before it reaches model processing.
- **Anomaly Detection:** Monitor usage patterns and outputs to detect unusual behavior that may indicate attempts at data extraction or system misuse.
- **Output Filtering:** Implement post-processing filters that screen model outputs for potentially leaked sensitive information before returning responses.
- **Training & Awareness:** Educate developers and users on secure prompt engineering practices and the risks of sharing sensitive information with LLMs.

Note: Regular security testing with red-team exercises is essential to continuously improve defenses against emerging attack vectors.

Navigating Compliance Frameworks with Open-Source LLMs

GDPR Compliance

- **Data Minimization:** Configure open-source LLMs to process only necessary data and implement automated retention policies.
- **Data Subject Rights:** Establish mechanisms for data access, erasure, and portability requests when user data interacts with LLMs.
- **Processing Records:** Maintain detailed documentation of processing activities, including purpose, data types, and security measures.

SOC 2 Compliance

- **Access Controls:** Implement robust authentication, authorization, and audit logging for all LLM interactions and management functions.
- **Monitoring & Incident Response:** Deploy continuous monitoring solutions with automated alerts for abnormal LLM usage patterns.

HIPAA Compliance

- **Deployment Options:** Either self-host with robust security controls or use HIPAA-eligible cloud services with signed BAAs.
- **PHI Protection:** Implement end-to-end encryption, data tokenization, and strict access management for all health information.

Note: Open-source LLMs are tools that must be integrated into a compliant pipeline—the models themselves are not certified for regulatory compliance.

Architecture Patterns for Isolating Model Logic from Sensitive User Data

Multi-tier Architecture Patterns

- **N-Tier Separation:** Implement strict logical separation between presentation, business logic, model inference, and data storage layers.
- **API Gateway Pattern:** Create a dedicated gateway that handles authentication, request validation, and access control before data reaches the model.

Microservices & Containerization

- **Service Isolation:** Deploy model inference as isolated microservices with strict communication boundaries and controlled data flow paths.
- **Data Transformation Pipeline:** Implement intermediate services that sanitize, anonymize, and transform sensitive data before it reaches LLM components.

Data Isolation Techniques

- **Command Query Responsibility Segregation (CQRS):** Separate read and write operations, allowing different security policies for each path.
- **Federated Data Access:** Implement a mediator pattern where the model accesses only abstract representations of sensitive data, not the raw data itself.

Best Practice: Apply "Defense in Depth" by combining multiple isolation patterns with strict access controls, network segmentation, and regular security assessments.

Ken Huang, CISSP

AI Book Author |Speaker | DistributedApps.AI |OWASP Top 10...

Q&A