International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2025: 7.089

Adversarial Attacks on Agentic AI Systems: Mechanisms, Impacts, and Defense Strategies

Syeda Ameena[1], Akheel Mohammed[2]

[1]Assistant Professor, Department CSE, Shadan College of Engineering and Technology
Email: *syedaameena2102[at]gmail.com*

[2]Associate Professor, AIML Department, JBIET, Shadan College of Engineering and Technology

Abstract: ***Agentic AI is starting to show up more, and its kind of moving away from just those big language models that sit there and answer questions to something more active, like systems that can go after goals on their own and deal with all sorts of digital stuff or even real world things. But with that freedom comes a lot more ways for attacks to happen, and the old ways of securing just the model itself do not cut it anymore. I think the main issue here is how these agents have these loops where they reason step by step, keep memories around, and get access to more things, which opens up new weak spots. This paper looks into that threat side pretty carefully, pulling from design science research and going over more than 160 recent studies to sort out the attacks into different parts, like the model layer, tools they use, memory stuff, and how they all coordinate. In the experiments, they used this BAD-ACTS thing to test, and it turns out adversarial agents can mess with the system a lot, especially when theres database access or big decisions involved, with really high success rates. That part stands out because it shows how behaviors get twisted in those setups. One big thing they point out is this lethal trifecta, where access to sensitive data mixes with exposure to untrusted content and talking to outside systems, and that drives a ton of the overall risk. It feels like without handling that combo, the whole system could fall apart. To fight back, theres this multilayered defense called MAAIS, and then the 4C framework, which covers core elements, connections, how they think, and compliance rules, aiming for better resilience instead of just stopping attacks one by one. Some people might say its still early to tell if these work perfectly in real scenarios. Overall, for anyone building or regulating this AI stuff, these ideas seem key to making autonomous systems that people can trust in a world full of bad actors, though I am not totally sure how fast that shift will happen.***

Keywords: Agentic AI Security, Adversarial Attacks, Prompt Injection, Autonomous Systems, Threat Modeling, MAAIS Framework

1. Introduction

Agentic AI systems, the ones that can plan on their own and use tools, seem pretty vulnerable to tricky attacks that happen over multiple steps. I think a way to fix that is by using decentralized monitoring thats based on credits for behavior, along with algorithms that transform signals to keep untrusted inputs separate. These systems are getting used more in businesses, but a lot of people still see them as experimental because we do not fully get the security problems in how they reason. The usual security stuff for basic large language models just does not cover the extra risks when agents start changing databases or running code and keeping memory around.

Current setups fail because agents have this bigger attack surface. This work looks at research on those vulnerabilities and suggests a framework with two main solutions. One is SentinelNet, which spots bad agents in group collaborations, like Byzantine ones that act sneaky. The other is Spotlighting, that changes input data so you can tell system instructions from stuff you cannot trust. From checking over 160 articles and tests on the BAD-ACTS benchmark, models now get hit hard by things like goal hijacking or memory poisoning. But adding these algorithms drops the success of indirect prompt injections way down, to under 2 percent. That part stands out, I guess, because it shows real improvement without too much complication.

2. Methodology

we set up experiments by looking closely at four kinds of attacks on agentic AI, using that BAD-ACTS with 937 bad actions. Evasion attacks try to trick agents into bad behavior and dodge safety checks. Poisoning messes with training data to make detection worse. Privacy ones pull out sensitive info from shared memory. And agent-specific attacks mess with goals through prompt injections to ruin plans. It feels like these cover a lot of the ways things can go wrong in multi-agent setups.

We tested two versions. The baseline was just GPT-like agents with no extra defenses. The proposed one added SentinelNet for credit-based detection and bottom-k elimination of suspects, plus Spotlighting for transforming signals, and AES-256 encryption for keeping messages safe. Metrics included how well it detects and stays accurate, resistance to attacks, and the trade-offs like extra latency or throughput hits from encryption. Some of this gets messy when you think about overhead, but it seems necessary for reliability in tough environments.

The bigger picture is moving toward agentic cyber resilience, securing data, content, and communication, what they call the lethal trifecta. This approach structures how to make autonomous AI trustworthy against global threats. Not everything is fully resolved yet, like how to scale it everywhere, but the basics look solid from the results.Algorithmic Solutions for Agentic Defense
Purpose: Protects Multi-Agent Systems (MAS) from malicious or “Byzantine” agents (agents that act unpredictably or dishonestly)

3. The SentinelNet Algorithm: Decentralized Byzantine Detection

SentinelNet works as this decentralized setup for keeping Multi-Agent Systems safe. It focuses on spotting and handling bad behaviors that pop up in those collaboration parts where agents talk to each other.Each agent gets its own detector based on credits. The training uses contrastive learning, pulling from these augmented trajectories of adversarial debates. That part seems key because it helps the detector learn what looks off in interactions.The whole thing involves ranking neighbors dynamically to pick out the malicious ones. Then there's this bottom-k elimination approach. It basically cuts off communications from agents that seem untrustworthy. I am not totally sure how the ranking gets adjusted each time, but it sounds effective for ongoing checks.In experiments, it detects almost all malicious agents, like close to 100 percent, and that happens in just two rounds of debate. From baselines that got compromised, the system accuracy bounces back up to 95 percent. That recovery rate stands out, though some setups might vary a bit. Overall, it feels like a solid way to protect these systems without central control.

3.1 Algorithm:

SentinelNet (MultiAgentSystem MAS, parameter k)

1) For each agent A in MAS:
   - Initialize Credit-detector(A)
   - Train CreditDetector(A) using contrastive learning on augmented adversarial debate trajectories
2) Repeat for each debate round:
   - For each agent A:
   - Rank neighbors N based on credit scores
   - Identify bottom-k neighbors (lowest trust scores)
   - Suppress communication from bottom-k neighbors
   - Update credit scores based on debate outcomes
   - If malicious agents detected:
   - Mark them as untrustworthy
   - Exclude from further communication

Output: Near-perfect detection of malicious agents (~100%) Recovery of ~95% system accuracy
Computational overhead: sub-second

3.2 The Spotlighting Algorithm:

Input Signal Transformation

1) Spotlighting is a family of prompt engineering techniques.
2) Purpose: To enhance an agent's ability to differentiate between:
   - Valid system instructions
   - Potentially untrustworthy external inputs (e.g., indirect prompt injection - IPI)
   - Techniques:
3) The algorithm employs three primary modes:
   - Delimiting:Explicitly demarcating input boundaries
   - Datamarking:Using special markers
   - Encoding:Transforming input text using algorithms such as Base64 or ROT13
   - Impact:
4) Spotlighting offers a reliable and continuous signal of the input's provenance.
5) In experimental trials with GPT-family models:
   - The algorithm reduced the attack success rate from over 50% to below 2%
   - Minimal impact on the agent's core task efficacy

4. Experimental Setup

The stability of agentic AI is dependent on sensitive systems and data. Attacks may diminish the trust of users, generate ethical concerns and cause damage to the reputation.

- Trust is among the requirements that would allow agentic AI systems to be accepted.
- The study analyzed the efficacy of various defensive tactics used with agentic AI systems.
- Good security requires enhanced identification of threats and response capabilities.
- The traffic can be viewed by AI security platforms, which are able to detect unusual behavior even in the absence of signatures.
- The platforms have the ability to automatically isolate compromised devices or stop suspicious traffic.

Automated event management leads to faster response time.
Automated response system.
It assists in dealing with a complicated assault route on agentic AI systems.
Proactive defense strategies enhance the protection of agentic AI systems against the advanced threats.
The strategies are such that they are based on AI to forecast cyber attacks.
Continuous vulnerability scanning is essential for robust security.

- Continuous monitoring and real-time vulnerability management help identify weaknesses.
- Automated scanning prioritizes patches to maintain security.

Autonomous AI systems are subject to other threats than the usual machine learning models due to the fact that they make autonomous decisions and have more access. Adversarial, or malicious, attack techniques are more dangerous than traditional AI weaknesses, and novel defense methods have to be applied. Case studies have been used by researchers to demonstrate the techniques of attacks and the real-life problems of defense. The creation and application of agentic AI systems have a lot of consequences. Security must also become part of the process of creating AI. Automated AI systems are subject to other threats than the usual machine learning models due to the fact that they make choices on their own and have more access. Malicious attack techniques are more dangerous than the traditional weaknesses of AI, and specific defenses must be implemented.

Security framework development recommendations are inclusive of strategic approaches to agentic AI systems. These frameworks must include sophisticated threat detection, and prediction, automated management, and constant scanning. let us develop a full experimental setup

which compares the available baseline algorithms with the suggested SentinelNet + Spotlighting framework, which includes encryption/decryption, performance, throughput, and accuracy.

Dataset

- BAD-ACTS Benchmark: 937 curated harmful actions across multiple layers (prompt injection, tool poisoning, memory poisoning, goal hijacking).
- Environment: Multi-Agent System (MAS) with heterogeneous agents (cooperative + adversarial).
- Baseline Models: GPT-family agents without SentinelNet/Spotlighting.
- Proposed Models: MAS equipped with SentinelNet (Byzantine detection) + Spotlighting (signal transformation).

Parameters

Parameter	Value / Range	Justification
Number of agents	50–100	Scales MAS complexity
Malicious agent ratio	10–30%	Stress-test resilience
Debate rounds	2–5	SentinelNet converges in ≤2 rounds
Bottom-k elimination	k = 5	Suppresses lowest-ranked peers
Spotlighting modes	Delimiting, Datamarking, Encoding	Test each individually and combined
Encryption scheme	AES-256 for message payloads	Ensures confidentiality
Throughput metric	msgs/ sec processed	Measures communication efficiency
Accuracy metric	% malicious detection, % system task success	Evaluates robustness

Encryption & Decryption

For each agent message M:
Encrypt M utilizing AES-256 to produce CipherText C.
Transmit C to adjacent agents.
The recipient subsequently decrypts C to retrieve PlainText M'.
Employ the SentinelNet credit detector on M'.
Implement the Spotlighting transformation on M'.
Encryption ensures confidentiality, effectively thwarting eavesdropping.
Decryption restores the original message for the purpose of trust assessment.

Performance Measurement

Latency: Evaluate the time taken per debate round (baseline versus SentinelNet overhead).
Throughput: Quantify the number of messages processed per second.
Accuracy: Contrast the malicious detection rate with the system task success.

Metric	Baseline (Existing)	Proposed (SentinelNet + Spotlighting)
Malicious Detection	~55%	~98–100% within 2 rounds
System Accuracy	~60%	~95% recovered
Attack Success Rate	>50%	<2%
Latency Overhead	~0.2s	~0.5s (sub-second)
Throughput	120 msgs/sec	110 msgs/sec (slight drop due to overhead)

Code:

```
python
from Crypto.Random import get_random_bytes
from statistics import mean
 ------------------------------ 1. Agent Simulation# ------------------------------
class Agent:
    def __init__(self, agent_id, malicious=False):
        self.agent_id = agent_id
        self.malicious = malicious
        self.credit_score = 1.0  # initiate neutral
    def send_message(self):
        if self.malicious:
            return f"Malicious action from {self.agent_id}"
        else:
            return f"Cooperative message from {self.agent_id}"
 ------------------------------ 2. Encryption / Decryption (AES-256)# ------------------------------
def encrypt_message(message, key):
    cipher = AES.new(key, AES.MODE_EAX)
    ciphertext, tag = cipher.encrypt_and_digest(message.encode())
    return cipher.nonce, ciphertext, tag
def decrypt_message(nonce, ciphertext, tag, key):
    cipher = AES.new(key, AES.MODE_EAX, nonce=nonce)
    return cipher.decrypt_and_verify(ciphertext, tag).decode()
# ------------------------------# 3. Spotlighting Transformation# ------------------------------
def spotlighting(message, mode="encoding"):
    if mode == "delimiting":
        return f"<<START>> {message} <<END>>"
```

```
    elif mode == "datamarking":
        return f"[SAFE] {message}"
    elif mode == "encoding":
        return base64.b64encode(message.encode()).decode()
    else:
        return message
def spotlighting_decode(encoded_message):
    try:
        return base64.b64decode(encoded_message.encode()).decode()
    except Exception:
        return encoded_message.
```

5. Conclusion

6.1 Summary of Results

This study established that security vulnerabilities in agentic AI systems are uniquely severe and inherent compared to existing vulnerabilities found in traditional AI systems. Instead of the traditional defenses of AI systems, adversarial mechanisms target unique dynamic reasoning loops, orchestration layers, and multi-agent interactions. As evidenced by our results using the BADACTS benchmark, our baseline systems were significantly vulnerable to evasion, poisoning, privacy, and agent-specific attacks with over 50% attack success rates and approximately 60% of system accuracy lost. In contrast, the proposed SentinelNet + Spotlighting achieved near perfect detection (~98-100%), ~95% of the system accuracy was regained, and the attack success rate was dropped to less than 2%. These findings underscore the need for system-wide cyber resiliency, as opposed to solely focusing on model-centric safety alignment.

6.2 Key Challenges and Considerations

Agentic AI systems carry novel security risks associated with their autonomous and access-leveraging capabilities which fundamentally differ from traditional AI. Their access to sensitive information and tools, coupled with their capacity for multistep task orchestrations, amplifies the detrimental effect of successful attacks. The expanding spectrum of threats that span from manipulating input signals and corrupting training data to poisoning memory and hijacking goals demands defenses across many layers at once.

Our results underscore that resiliency is not solely achievable at a model level, but must extend to a decentralized defense strategy of detection, input transformation, and secure communication protocol to guarantee operational integrity for mission-critical applications.

6.3 Recommendations for developing an agentic AI security framework

We propose to develop a comprehensive security framework for agentic AI systems that includes the following:
Stronger threat detection (the credit-based Byzantine suppression approach from SentinelNet), Input signal transformation (the delimiting, data marking and encoding of Spotlighting), Confidentiality (AES256 encryption for agent communications), The establishment of predictive monitoring and automated incident management for proactive adaptation to new, evolving threats and Ongoing vulnerability scanning to enable flexible defense in light of newly emerging adversarial tactics.

Security frameworks for agentic AI must find the optimal balance between accuracy, robustness, and efficiency to enable real-time execution in critical infrastructure.