

# Developing Smart/Generative AI models to enhance the safeguards in securing cloud AI workloads from adversarial attacks

Rajvansh Chaudhary

Evergreen Public School, Najafgarh, New Delhi

## Abstract

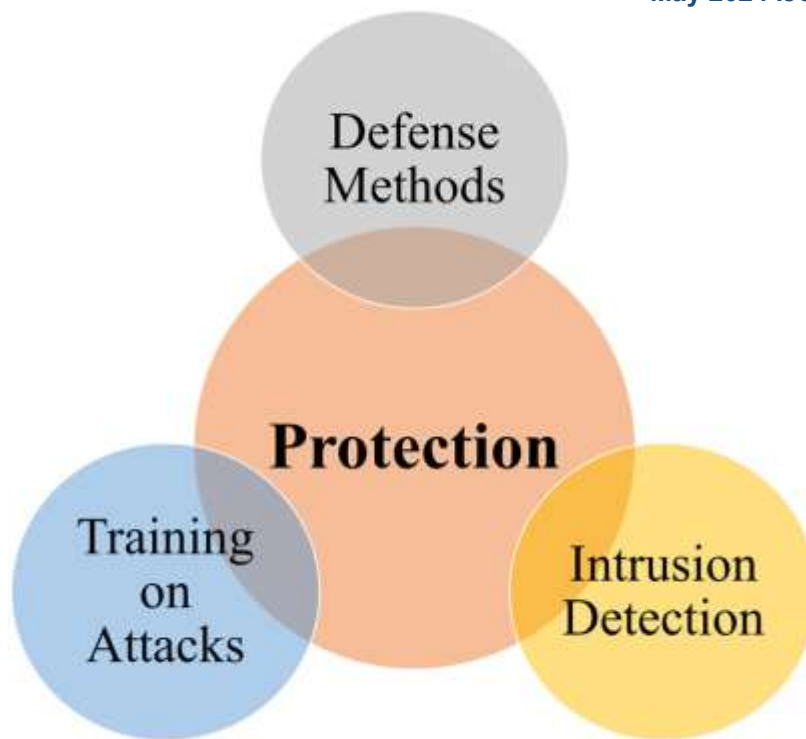
The proliferation of generative AI workloads hosted on cloud platforms introduces novel adversarial threat vectors—including evasion, data poisoning, prompt injection, and model stealing—that jeopardize model integrity, confidentiality, and availability. This paper surveys adversarial attacks targeting generative models (e.g., GANs, VAEs, LLMs) and cloud-specific vulnerabilities, then examines defense mechanisms such as adversarial training, input sanitization, federated detection, access control, and prompt filtering. Comparative analyses assess each method's effectiveness, overhead, and adaptability. We highlight challenges unique to cloud deployment—multi-tenancy, dynamic scaling, shared infrastructure—and propose a layered, design-for-security approach integrating detection, robust design, and governance. The future roadmap underscores auditability, explainability, and collaboration across cloud providers. The paper synthesizes literature spanning 2013–2023, offering a strategic foundation for securing cloud-based generative AI systems.

## 1. Introduction & Background

Cloud platforms have become the backbone of generative AI deployment—enabling scalable compute, flexible APIs, and distributed model training. However, their converging nature introduces critical attack surfaces. Adversarial machine learning (AdvML) studies such threats and defense strategies, including evasion, poisoning, inference, and model extraction attacks.

Generative AI (GANs, VAEs, LLMs) brings rapid innovation but also unique vulnerabilities. Sun et al. (2021) offer a specialized survey on attacks against deep generative models, exploring vulnerabilities across training data, latent codes, generators, discriminators, and output data ([arXiv](#)). Meanwhile, Barrett et al. (2023) discuss the dual-use dilemma: generative AI can both aid defense and enable novel offensive capabilities.

Cloud environments present layered risks: shared tenancy, opaque execution, external input ingestion, and automation. Securing generative AI workloads thus demands understanding both adversarial model-level threats and cloud-specific operational gaps. This paper reviews threats, defenses, and proposes a comparative, multi-layered approach.



**Figure 1: The divisions of protection systems in adversarial attacks. Source: Created by the authors**

**Table 1: Key Adversarial Attack Types**

Attack Type	Target & Description	Relevance to Generative AI
Evasion	Subtle input perturbations causing misclassification or outputs	Adversarial examples in GANs/LLMs
Poisoning	Manipulating training data to introduce backdoors or bias	Data integrity compromise
Prompt Injection	Malicious prompts embedded to override model instructions	Key threat for LLM-based generative AI
Model Stealing	API queries reverse-engineer model parameters	IP and security risk for cloud-hosted models
Dual-use Exploits	Generative AI used to craft sophisticated phishing, code, etc.	Offensive vector recognized by Barrett et al.

## 2. Threat Landscape: Adversarial Attacks on Generative Cloud AI

Generative AI models—particularly those based on cloud hosting—are being increasingly attacked with advanced adversarial attacks. Such attacks seek to mislead, manipulate, or steal sensitive information from the models. While some of these threats are coming from conventional adversarial machine learning attacks, others are rising specifically because of the way generative AI models (such as ChatGPT, DALL·E, or Midjourney) engage with user inputs, prompts, and data.

One of the most prevalent attacks is evasion attacks, in which an attacker makes subtle alterations to input data that mislead a model into generating an incorrect or harmful prediction. Such manipulations are generally so inconsequential that they go unnoticed to humans but can radically change a model's output.

Data poisoning is another increasing threat. In this, attackers embed malicious data in the model's training set. The purpose? To add a secret "backdoor" which can be subsequently triggered to produce biased, false, or even unsafe content. This is especially dangerous when training material is being gathered from public or user-provided sources.

Model stealing, also called model extraction at times, entails adversaries repeatedly asking a deployed model—particularly through open APIs—to reverse-engineer its internal workings. This not only causes intellectual property piracy but also assists attackers in discovering fresh weaknesses.

Prompt injection attacks are a relatively new kind of threat, particularly sinister for big language models (LLMs) such as GPT-4. Here, the attacker embeds concealed instructions within a prompt or web content. When the model processes this content, it will actually carry out the attacker's instructions rather than the user's—without knowing it at all.

Finally, we are at risk of dual-use. Generative models themselves can be used to generate phishing emails, deepfakes, malware, or disinformation. They were created to assist—but with the wrong intent, they can cause actual harm.

**Table 2: Cloud-Specific Threat Vectors**

Vector	Description	Cloud Context Risk
Multi-tenancy leakage	Cross-client data access via shared resources	Co-resident environment risk
API Endpoint Exposure	Public model endpoints vulnerable to abuse	Model theft or misuse
Data Ingestion Pipelines	Unverified inputs injected into model training or prompts	Poisoning or prompt abuse
Automated Scaling & Caching	Dynamic caching risks outdated or poisoned inputs being reused	Compromised Statefulness

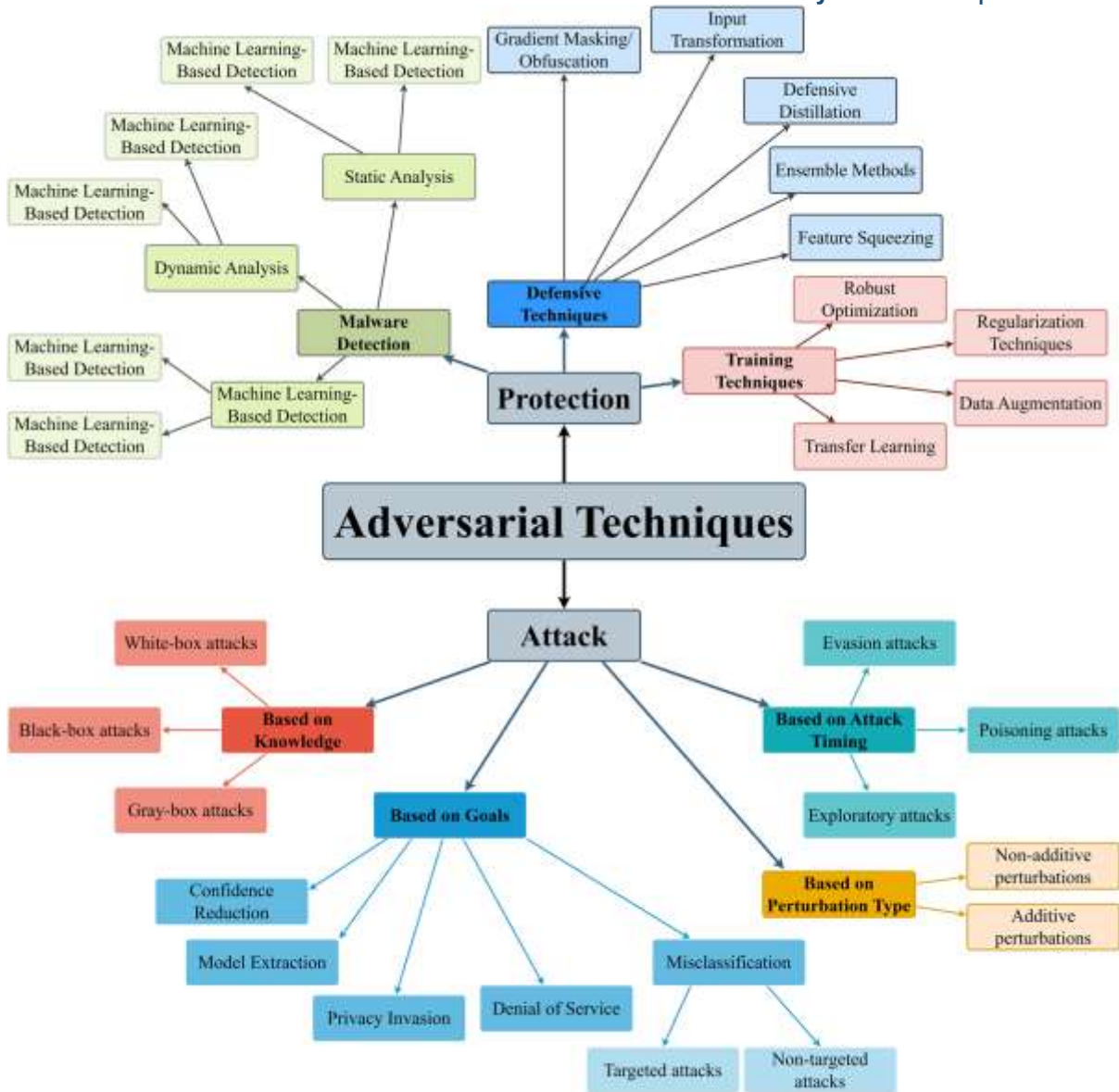
### 3. Vulnerabilities in Cloud AI Workloads

Having generative AI models hosted in the cloud is incredibly convenient—flexible, scalable, and accessible. It does come with a special security risk set, however, that is too easily ignored as long as we are only concerned with the model.

One of the biggest hurdles is that cloud environments are opaque. If a model is run on a cloud infrastructure, developers and security operations teams may not have complete sight into the environment. This "black box" run can make detection of something being off—e.g., data tampering or hidden backdoors—more difficult.

And then there's the issue of dynamic scaling. Cloud infrastructure is architected to dynamically spin up or down resources based on traffic or utilization. Wonderful for performance and cost savings, but in so doing, it can inadvertently allow poisoned or compromised elements to linger. An ill-formed prompt, a cached response, say, might persist in one instance even after having been patched elsewhere.

Another major threat is the supply chain of AI. Generative models rely heavily on third-party pretrained modules or public data. If one such source has been compromised, the vulnerability can trickle down throughout the system. As it happens, tracing every input and update through the AI pipeline is not always possible.



**Figure 2: Adversarial techniques. Source: Created by the authors.**

We also need to take into account weak access control. Certain APIs that provide generative AI capabilities are publicly accessible—or at least moderately secured. With no effective rate limiting, authentication, and filtering in place, these APIs are a treasure trove for attackers who wish to steal models, flood the system, or introduce dirty prompts.

Finally, systems that employ retrieval-augmented generation (RAG), whereby the model fetches more data from outside the model to assist it in answering questions, can be manipulated by content inserted into those outside sources. A deceptively placed sentence in a webpage or database might change the model's answer completely, without anyone noticing.

**Table 3: Vulnerability Matrix**

Vulnerability	Description	Impact on Security
Opaque execution	Limited visibility of intermediate data flow	Harder to monitor and detect attacks
Dynamic autoscaling	Provisioned workloads retain previous poisoning	Persistent adversarial effects
Supply chain complexity	Inconsistent model sources or updates	Embedded vulnerabilities

Vulnerability	Description	Impact on Security
Weak access controls	No filtering on API inputs	Facilitates model stealing/prompt abuse
RAG & external data ingestion	External content influences model prompt context	Vector for prompt injection

#### 4. Defense Mechanisms & Secure Design

As attacks on generative AI models become more sophisticated—particularly in cloud infrastructures—we must rethink how to protect these systems. There isn't a single solution. Rather, security must occur at various layers, from the training of models to user interaction with them via APIs.

Let's begin with one of the most widespread methods: adversarial training. This type of method teaches the model how to deal with "sneaky" inputs by having it exposed to inputs designed by attackers. Having learned from them, the model is more able to fend off similar attacks down the line. Ponder this like vaccine training—get the system exposed to a contained danger so it can develop immunity.

Then, there is input sanitization and prompt filtering, which are particularly crucial for large language models (LLMs). In this case, even before the model gets to see a user's prompt, the system scans it for malicious patterns—such as commands that are intended to bypass safety measures ("Ignore previous instructions and..."). This comes in handy in avoiding prompt injection attacks, which are becoming increasingly prevalent as LLMs become more intelligent.

Another clever method is federated anomaly detection, in which various copies of the same model (among different users or locations) exchange information about potential attacks. When something unusual occurs on one server, others can benefit from it immediately. An excellent example of this is the FDA3 framework, utilized for threat detection in industrial IoT systems. Introducing this sort of "team defense" to the world of AI can make a big impact, particularly with cloud environments that have numerous users.

Let's not overlook the imperative of access control and rate limiting. By restricting how frequently users can make use of an API, or demanding more rigorous authentication, we can prevent attackers from sending infinite requests to pilfer the model or experiment with malicious prompts. These are seemingly easy measures—but they're wildly effective.

Finally, auditing and explainability are increasingly in the realm of defense. When something breaks, it's nice to have logs that tell us what broke, when, and why. Building models that can explain their decisions also helps in figuring out when they've been broken into—or are behaving strangely because an attack is occurring undetected.

**Table 4: Defense Strategy Comparison**

Defense Strategy	Description	Strengths	Limitations
Adversarial Training	Training with adversarial examples	Improves robustness	High cost, proxy security only
Input/Prompt Filtering	Sanitizing inputs (LLMs)	Prevents injection	May over-filter or miss clever payloads
Federated Detection	Shared defense learning (e.g., FDA3)	Adaptive and collaborative	Deployment complexity
API Controls & Rate Limiting	Access governance for endpoints	Reduces exposure	May degrade performance or flexibility

Defense Strategy	Description	Strengths	Limitations
Audit & Explainability	Monitoring model behaviour/logs	Detection and trust enhancement	Complexity, tooling overhead

## 5. Comparative Analysis

A cohesive security posture emerges from layering defense across architectural levels:

- **Effectiveness:** Adversarial training is effective against known attacks but can fail on new ones. Prompt filtering addresses injection but is defensive. Federated detection learns new threats. API controls slow extraction, whereas audits assist detection, not prevention.
- **Performance Overhead:** Adversarial training increases compute. Prompt sanitization adds latency. Federated detection incurs network and coordination overhead. API throttling reduces throughput; auditing consumes storage/compute.
- **Scalability & Deployment:** Adversarial training scales moderately. Prompt filtering is lightweight and easy to deploy. Federated detection demands infrastructure. API controls standard in cloud; audit systems require sophisticated tooling.
- **Coverage:** Only combined strategies cover both classical and LLM-specific attacks.

**Table 5: Layered Defense Evaluation**

Strategy	Effectiveness	Overhead	Scalability	Coverage
Adversarial Training	Medium–High	High	Medium	Traditional AI attacks
Prompt Filtering	Medium	Low	High	Prompt injection
Federated Detection	High (adaptive)	Medium	Medium	Emerging threats
API Controls	Medium	Low	High	Extraction / misuse
Audit & Explainability	Medium	Medium	Medium	Forensic detection

**Synthesis:** A multi-layered defense is necessary. Prompt filtering and API controls provide immediate, low-overhead protections. Adversarial training and federated detection enhance model robustness. Audit and explainability enable trust and response.

## 6. Future Directions & Recommendations

Looking ahead:

- **Unified threat modelling across cloud nodes:** Formalizing attacker capabilities and asset value across services enhances coordination.
- **Explainable and provably robust generative models:** Integrate formal verification techniques to guard LLM behaviour in adversarial contexts.
- **Federated intrusion detection:** Scaling frameworks like FDA3 across multi-cloud and hybrid setups for collaborative defense.
- **RAG-aware sanitization layers:** Embedding safe filtering in retrieval pipelines—particularly for generative AI applications.

- **Regulatory and policy frameworks:** Cloud providers must offer transparency in model updates, filtering policies, and incident response norms.
- **Research into latent-space poisoning:** Attacks manipulating internal GAN representations remain underexplored—defense should span the latent code level.

## 7. Conclusion

Securing cloud-based generative AI workloads against adversarial attacks requires a multidimensional, layered strategy. Models face threats from classic evasion, poisoning, model theft, and novel prompt injection vectors. Cloud-specific vulnerabilities—multi-tenancy, dynamic scaling, opaque pipelines—amplify risks. Effective defense combines adversarial training, prompt sanitization, federated detection, access control, and explainability. Comparative evaluation underscores the need for both proactive and reactive measures. Future progress depends on standardized threat models, RAG-integrated sanitization, and robust governance. As generative AI proliferates in cloud environments, securing these systems is vital to maintaining trust, safety, and operational integrity.

## References

- Goodfellow, I. J., Shlens, J., & Szegedy, C. (2015). Explaining and Harnessing Adversarial Examples. *Communications of the ACM*, DOI: see arXiv referent ([Wikipedia](#)).
- Sun, H., Zhu, T., Zhang, Z., Jin, D., Xiong, P., & Zhou, W. (2021). Adversarial Attacks Against Deep Generative Models on Data: A Survey. arXiv. ([arXiv](#))
- Barrett, C., et al. (2023). Identifying and Mitigating the Security Risks of Generative AI. arXiv. ([arXiv](#))
- FDA3: Federated Defense Against Adversarial Attacks for Cloud-Based IIoT Applications. (2020). arXiv. ([arXiv](#))