

LLM and Gen AI Data Security Best Practices

OWASP Top 10 for LLM Apps & Gen AI
Data Security Initiative



The information provided in this document does not, and is not intended to, constitute legal advice. All information is for general informational purposes only. This document contains links to other third-party websites. Such links are only for convenience and OWASP does not recommend or endorse the contents of the third-party sites.

License and Usage

This document is licensed under Creative Commons, CC BY-SA 4.0

You are free to:

- Share – copy and redistribute the material in any medium or format
- Adapt – remix, transform, and build upon the material for any purpose, even commercially.
- Under the following terms:
 - Attribution – You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner but not in any way that suggests the licensor endorses you or your use.
 - Attribution Guidelines - must include the project name as well as the name of the asset Referenced
 - OWASP Top 10 for LLMs - LLM and Gen AI Data Security Best Practices Guide 1.0
- ShareAlike – If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.



Table of Content

Acknowledgements.....	3
Audience.....	4
Introduction.....	5
Traditional vs. LLM-Specific Data Security.....	7
Key Risks Overview: Data Security in LLMs.....	11
Data.....	16
Practical LLM Data Security Principles.....	24
Monitoring and Auditing Guidelines.....	28
Securing Data Flows in LLM Agents.....	30
Future Trends & Challenges in LLM Data Security.....	37
Secure Deployment Strategies for LLMs.....	43
LLM Data Security Governance.....	50
Conclusion & Call to Action.....	55
OWASP Top 10 for LLM Project Sponsors.....	56
Project Supporters.....	57
References.....	58



Acknowledgements

Authors

[Emmanuel Guilherme Junior](#) - Data Gathering Methodology Initiative Lead

[Scott Clinton](#) - Project Co-lead, Industry Outreach, Collab and Growth

Contributors

Chris Hughes
Rock Lambros
Manuel Villanueva
Ben Moreland
Vineeth Sai Narajala
Evgeniy Kokuykin
Trent Holmes

Reviewers

Ken Huang
Vineeth Sai Narajala
Bill Glennon



Audience

This document addresses three critical roles in the secure deployment and management of data security in Large Language Models (LLMs):

1. **Technical Cybersecurity Leaders:**

Roles: Security architects, security engineers, and SOC teams.

Focus:

- Adopt defense-in-depth strategies (e.g., layered encryption, secure enclaves).
- Enforce industry-standard frameworks (NIST SP 800-53, ISO/IEC 27001/20547/5338/38507/23894/24028/23053/22989/42001, MITRE ATLAS) and ensure compliance (GDPR, CCPA, HIPAA, EU AI Act, AI Action Plan).
- Implement robust monitoring (SIEM/XDR) and incident response for LLM pipelines.

2. **Developers (Engineers & Data Scientists):**

Roles: ML engineers, software developers, and data scientists.

Focus:

- Integrate secure coding and data validation into CI/CD pipelines.
- Utilize artifact signing and verified model registries (e.g., MLflow) to ensure model integrity.
- Implement adversarial testing frameworks and secure data ingestion methods.

3. **CISOs (Chief Information Security Officers):**

Roles: Executive leaders overseeing cybersecurity strategy and risk management.

Focus:

- Align LLM usage with organizational risk appetite and compliance mandates.
 - Institutionalize zero-trust models and robust access governance.
 - Foster cross-functional collaboration (Legal, Privacy, Compliance) for holistic AI security policies.
-



Introduction

The rapid proliferation of Large Language Models (LLMs) across various industries has highlighted the critical need for advanced data security practices. As these AI systems become more sophisticated, they bring with them unprecedented risks, including potential breaches of sensitive information and challenges in meeting stringent data protection regulations. This white paper outlines a comprehensive set of best practices for LLM data security, designed to address these emerging vulnerabilities and mitigate the associated risks effectively.

This initiative complements the OWASP Top 10 for LLM AI Applications 2025 list, reinforcing our ongoing commitment to a secure and responsible AI ecosystem. The objective of this paper is to inform, educate, and provide insights into protecting data in the context of LLMs and their best practices. We bring a new perspective with additional content made by and for the community. Complimenting the other useful documents created by the project such as the LLM Cybersecurity and Governance Checklist, LLM GenAI Security Center of Excellence, The Guide for Preparing and Responding to Deepfake Events, The AI Security Solution Landscape Guide and GenAI Red Teaming Guide. Authored and presented by the Data Gathering Methodology Team, this white paper brings a rigorous and methodical approach to data collection and analysis, ensuring that our recommendations are both practical and aligned with global security frameworks.

Why Data Security is Critical for LLMs

Data security is critical for Large Language Models (LLMs) due to the significant risks and vulnerabilities associated with their use and development. Data is the *"lifeblood"* of all LLMs; ensuring their protection includes:

Protecting Sensitive Information

LLMs are often trained on vast datasets that may contain sensitive or confidential information. Ensuring data security is crucial to:

- Prevent unauthorized access to personal, financial, or proprietary data
- Maintain user privacy and trust
- Comply with data protection regulations

Implementing strong encryption, access controls, and anonymization techniques helps safeguard sensitive data used in LLM training and operation.



Ensuring Data Integrity and Reliability

Maintaining the integrity and reliability of data used by LLMs is essential for:

- Producing accurate and trustworthy outputs
- Preventing the spread of misinformation or biased content
- Ensuring the model's performance aligns with its intended purpose

Implementing data validation processes and continuous monitoring helps detect and respond to potential threats that could compromise data integrity.

Addressing Privacy Concerns

LLMs raise significant privacy concerns due to their ability to process and generate human-like text. Robust data security measures are necessary to:

- Protect user inputs and prevent unauthorized disclosure of personal information
- Ensure compliance with privacy regulations like GDPR
- Maintain user trust in AI-powered applications

Adopting privacy-enhancing technologies such as differential privacy and federated learning can help protect individual privacy while allowing LLMs to learn from broad data insights.

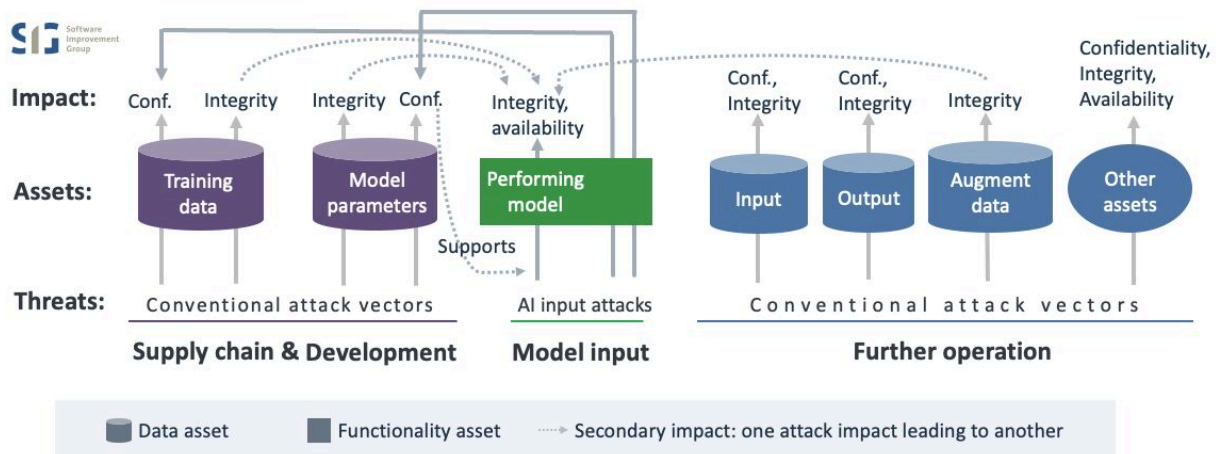
Mitigating Emerging Threats

As LLMs become more prevalent, they introduce new security challenges:

- Sophisticated phishing attacks using AI-generated content
- Manipulation of online information at scale
- Automated cyber-attacks leveraging LLM capabilities
- Misinformation
- Unbounded consumption

Robust data security is essential for LLMs to ensure their responsible development and deployment. By implementing comprehensive security strategies, organizations can harness the power of LLMs while safeguarding sensitive information and maintaining user trust. Implementing advanced security measures and staying vigilant against evolving threats is crucial to protect LLMs and their users.

Traditional vs. LLM-Specific Data Security



https://www.linkedin.com/posts/robvanderveer_ai-aisecurity-activity-7274736168255074304-g9yq?utm_source=share&utm_medium=member_desktop

Rob van der Veer (Software Improvement Group)

Traditional Data Security Measures

Traditional security practices such as encryption, access control, data masking, and network security constitute the foundational layer for safeguarding data in Large Language Models (LLMs). These time-tested methods ensure baseline protection and help maintain compliance in any robust system. For instance, encryption at rest (e.g. AES-256) and in transit (e.g. TLS 1.3) secures data during storage and communication, while role-based access control (RBAC) and multi-factor authentication (MFA) limit unauthorized entry. Data masking or anonymization techniques mitigate the risk of accidental disclosure, and meticulous auditing and logging provide an audit trail for troubleshooting or compliance reporting. Firewalls, VPNs, and intrusion detection systems further strengthen network perimeters to ensure that only legitimate traffic can pass through. Together, these controls form the bedrock for applying additional, more specialized protocols for LLMs.


LLM-Specific Security Adaptations

These traditional necessary measures must be augmented to address unique challenges that arise throughout the AI lifecycle. Model lifecycle management and governance frameworks that encompass version control, rollback capabilities, and adherence to corporate standards, help monitor training data provenance and maintain traceability. Secure development processes and verified supply chains minimize vulnerabilities introduced by external libraries and custom scripts. Threat assessments such as adversarial testing, model inversion analysis, and penetration exercises reveal system weaknesses before they lead to compromise. For daily operations, incident response protocols, supported by continuous monitoring and real-time anomaly detection, are essential for rapid containment of breaches or data leakage events. These controls also intersect with broader legal and ethical considerations, including adherence to data residency rules and mitigation of unintended biases.

POTENTIAL RISKS & IMPACTS

The OWASP Top 10 for LLM GenAI Applications 2025 list, organized and curated by a group of experts, presents the top security risks and the potential impacts associated with Large Language Models. Six out of ten are specific to data security:

No	Risk	Explanation	Statistic
1	LLM01:2025 Prompt Injection	Malicious actors can manipulate prompts to alter the model's behavior, resulting in the generation of unauthorized content or performing unintended actions.	Research shows that even sophisticated defenses like Retrieval Augmented Generation (RAG) do not fully prevent these vulnerabilities.
2	LLM02:2025 Sensitive Information Disclosure	LLMs risk revealing personally identifiable information (PII), proprietary algorithms, or confidential business data through their output. For example, poorly sanitized data can lead to unintentional leaks.	Research shows that even sophisticated defenses like Retrieval Augmented Generation (RAG) do not fully prevent these vulnerabilities.
3	LLM04:2025 Data and Model Poisoning	Attackers can corrupt training data or the model itself, causing it to generate misleading outputs or compromise its integrity.	Adversarial examples have shown a 35% success rate in influencing model outputs, even with defensive mechanisms in place.
4	LLM03:2025	Insecure dependencies and libraries can introduce risks into LLM	The "Proof Pudding" attack (CVE-2019-20634)



	Supply Chain Vulnerabilities	deployments. Attackers can compromise these external sources to gain access to models and data.	demonstrated how disclosed training data facilitated model extraction, leading to significant privacy and security violations.
5	LLM05:2025 Improper Output Handling	LLMs may inadvertently generate harmful content or leak sensitive information if their outputs are not properly validated and controlled.	Recent breaches revealed that unfiltered LLM responses contributed to data exposure incidents in 12% of analyzed cases.
6	LLM07:2025 System Prompt Leakage	If attackers gain access to system prompts, they can extract details about the model's configuration and exploit them for further attacks.	Adversarial examples have shown a 35% success rate in influencing model outputs, even with defensive mechanisms in place.

BROADER IMPACT

Data security risks associated with large language models (LLMs) extend across multiple dimensions, with significant broader impacts including:

1. Misinformation and Manipulation: LLMs can be exploited to generate and spread false or misleading information, leading to:

- Social engineering attacks
- Erosion of trust in AI systems
- Potential societal impacts
- Link traps

2. Over Reliance on AI-generated Information: Trusting LLM outputs without proper validation can result in:

- Security breaches
- Legal issues
- Reputational damage

3. Ethical Concerns: The use of LLMs raises questions about:

- Bias in AI decision-making
- Fairness and discrimination
- Accountability for AI-generated content
- Transparency

The mitigation of these risks requires organizations to implement robust security measures, including data anonymization, input validation, output sanitization, continuous monitoring of LLM systems and more.



Additionally, developing ethical guidelines and maintaining human oversight in critical decision-making processes is crucial for responsible AI deployment.

This effort is closely aligned with the broader OWASP LLM and Generative AI Security Project, which provides in-depth guidance on mitigating these risks and establishing secure development and deployment practices. The [OWASP AI Security and Privacy Guide](#) offers a valuable broader perspective, addressing data privacy, ethical implications, and responsible AI development. The [OWASP AI Exchange](#) serves as a central repository for information and facilitates collaboration within the field.

While these initiatives directly address LLM security, established OWASP projects offer valuable, transferable insights. The [OWASP Application Security Verification Standard \(ASVS\)](#) provides a framework for verifying the security of applications integrating LLMs, while the [OWASP Testing Guide and Web Security Testing Guide \(WSTG\)](#) offers methodologies applicable to LLM-based systems.

The [OWASP Cheat Sheet Series](#) provides concise guidance on specific security techniques, and the [OWASP Threat Modeling Project](#), along with the [Threat Dragon tool](#), enable effective threat analysis for LLM deployments.

Supply chain security for LLM dependencies can be managed using [OWASP Dependency-Check and Dependency-Track](#). Furthermore, applying the [OWASP Software Assurance Maturity Model \(SAMM\)](#) facilitates the development of robust software security programs encompassing LLM lifecycles, and the [OWASP DevSecOps Maturity Model \(DSOMM\)](#) supports the integration of security into operational aspects of LLM infrastructure and data management.

Principles from the [OWASP Code Review Guide](#) are essential for scrutinizing code interacting with LLMs, and the [OWASP Security Knowledge Framework \(SKF\)](#) provides a broad foundation of security knowledge applicable across various LLM security domains.

This comprehensive set of OWASP resources is crucial for establishing and implementing effective LLM data security best practices.

For more information on the OWASP Top 10 for LLM 2025 list visit: <https://genai.owasp.org/llm-top-10/>



Key Risks Overview: Data Security in LLMs

The rapid adoption of Large Language Models (LLMs) has introduced unprecedented risks in data security. This white paper presents our view on data security and is dependent on the other content produced by the project. As organizations scale their reliance on LLMs, the challenges of protecting sensitive information, ensuring compliance, privacy and mitigating evolving threats become more acute. This section explores the critical risks associated with data in LLMs and offers forward-looking solutions to safeguard data integrity and security in 2025.

1. Sensitive Information Disclosure

LLMs are trained on extensive datasets, some of which may inadvertently contain sensitive or personally identifiable information (PII). Even with stringent data preparation, models can "memorize" sensitive information and reproduce it in their outputs, posing severe risks.

Key Risks:

- **Data memorization:** Retention of sensitive or proprietary information, leading to unintentional disclosures during inference.
- **Insecure data outputs:** Poorly designed output handling mechanisms may expose confidential information or trade secrets.

Impact:

- Non-compliance with data protection regulations like GDPR or CCPA.
- Reputational damage from privacy violations.
- Erosion of user trust in AI systems.

Mitigation Strategies:

- **Anonymization:** Remove direct identifiers and use advanced techniques such as k-anonymity or synthetic data generation.
 - **Differential privacy:** Inject statistical noise into datasets and outputs to obscure sensitive information while retaining utility.
 - **Output filtering:** Implement robust filters to detect and remove potentially sensitive content before model outputs are shared.
-



2. Data Breaches

LLMs handle large datasets, making them lucrative targets for cybercriminals. Breaches can result from insufficient storage security, weak encryption practices, or vulnerabilities in supply chains.

Key Risks:

- **Security misconfiguration:** Data storage exposed publicly provides attackers an entry point.
- **Unencrypted storage:** Failure to secure data at rest increases exposure to theft.
- **Supply chain vulnerabilities:** Open-source models and third-party dependencies can be entry points for attackers.

Impact:

- Loss of intellectual property, financial penalties, and operational disruptions.
- Exposure of sensitive training datasets containing PII or proprietary business data.

Mitigation Strategies:

- **Encryption:** Encrypt data at rest and in transit using robust algorithms like AES-256 and TLS 1.3.
 - **Zero trust architecture:** Restrict data access to authenticated and authorized users only.
 - **Supply chain security:** Conduct regular audits of third-party components, using tools to monitor for vulnerabilities in open-source dependencies.
-

3. Data and Model Poisoning

Adversaries can inject malicious data into training pipelines, compromising the integrity of LLMs. Poisoning attacks aim to bias model behavior or introduce vulnerabilities that can be exploited later.

Key Risks:

- **Training data corruption:** Insertion of deceptive data that influences model outputs.
- **Hidden backdoors:** Trigger phrases embedded into the model to elicit specific responses.

Impact:

- Biased or harmful content generation, compromising user trust.
- Security vulnerabilities in downstream applications reliant on the model.

Mitigation Strategies:

- **Data validation and cleaning:** Establish automated pipelines to detect and remove anomalies in training data.
 - **Adversarial training:** Include adversarial examples during training to build model resilience.
 - **Behavior monitoring:** Continuously analyze model outputs for unexpected patterns or behaviors that may indicate poisoning.
-



4. Unauthorized Access and Model Theft

Without robust access controls, LLMs are vulnerable to theft and unauthorized use. Malicious actors may attempt to clone proprietary models through extensive API interactions or steal intellectual property.

Key Risks:

- **API exploitation:** Attackers manipulate APIs to extract sensitive outputs or overwhelm systems with queries.
- **Model extraction:** Using black-box techniques, adversaries replicate LLM behavior, creating near-identical clones.

Impact:

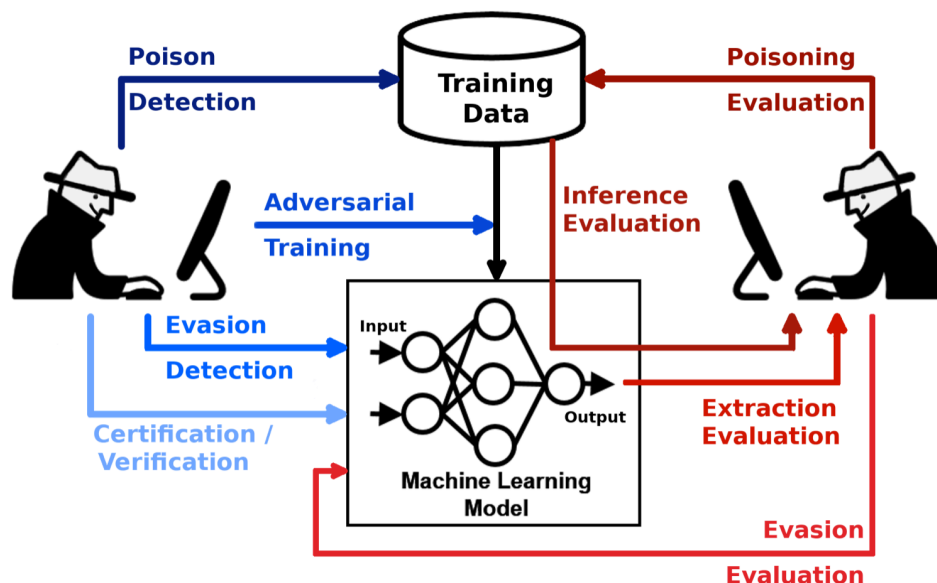
- Loss of competitive advantage and revenue.
- Creation of unregulated, potentially harmful clones that exploit LLM vulnerabilities.

Mitigation Strategies:

- **Access controls:** Enforce multi-factor authentication (MFA), role-based access controls (RBAC), and fine-grained permissions.
 - **Rate limiting:** Restrict the frequency and volume of API calls to deter extraction attempts.
 - **Watermarking:** Embed identifiers in models to trace unauthorized distribution of model as is, otherwise identifiers provide limited mitigation against threats.
-

5. Adversarial Attacks

Adversarial inputs can exploit model vulnerabilities, leading to harmful outputs or the extraction of sensitive data. These attacks include prompt injection, data inversion, and output manipulation.



<https://www.labellerr.com/blog/what-are-adversarial-attacks-in-machine-learning-and-how-can-you-prevent-them/>

Key Risks:

- **Prompt injection:** Crafting inputs that bypass safeguards to extract information or manipulate behavior.
- **Model inversion:** Reconstructing sensitive training data by analyzing model outputs.
- **Output manipulation:** Generating harmful or unintended content.

Impact:

- Exposure of private information embedded in training data.
- Amplification of misinformation or malicious behavior in downstream applications.

Mitigation Strategies:

- **Input validation:** Employ robust sanitization methods to filter malicious inputs.
- **Adversarial training:** Train LLMs on adversarial examples to improve robustness.
- **Output monitoring:** Use automated systems to detect and block harmful or inappropriate outputs.



6. Future Considerations in Data Security

Emerging Challenges:

As LLMs integrate more deeply into critical operations, new challenges emerge:

- **Privacy risks:** Increased reliance on user interactions heightens exposure to sensitive information.
- **Dynamic threat landscape:** Attackers continuously innovate, exploiting both known and unforeseen vulnerabilities.
- **AI agent and multi-agent systems:** Complex systems of planning, reasoning, followed by actions which greatly increase the attack surface.

Forward-Looking Solutions:

- **Privacy-enhancing technologies (PETs):** Federated learning, secure multi-party computation, and homomorphic encryption will play critical roles in securing sensitive data.
- **AI governance frameworks:** Establish clear guidelines for ethical AI use, ensuring alignment with global standards and regulations.
- **Continuous monitoring and updates:** Regularly audit and update security measures to address evolving threats.



Data

In today's data-driven landscape, securing large language models (LLMs) demands a holistic approach that goes beyond simply knowing your data, verifying versioning, and ensuring high standards of data quality, standardization, and efficient storage across silos and data lake architectures. For software developers, cybersecurity professionals, and CISOs alike, a robust data governance framework is essential—one that clearly defines data ownership and accountability while maintaining integrity throughout the entire lifecycle, from data ingestion and preprocessing to model training and inference. Integrating continuous risk assessments, strict adherence to cryptographic and access standards, and automated anomaly detection fortifies your defenses against data poisoning, leakage, and compliance violations. Moreover, leveraging advanced encryption, secure access protocols, immutable audit trails, and proactive incident response and disaster recovery plans ensures that security is not an afterthought but a core element of your strategic framework. By systematically embedding these best practices into your operational model, organizations can achieve a resilient, agile, and transparent security posture that meets both technical and executive requirements in today's dynamic threat landscape.

Key Stages

Data Sources

What Happens: Acquire raw data from databases, APIs, web scraping, or third-party providers.

Security Controls:

- **Data provenance & integrity:** Use metadata tracking (Apache Atlas, Data Catalog) to establish auditable lineage. Apply HMAC (HMAC-SHA-256) to verify integrity.
- **Network & transport security:** Enforce TLS 1.3 or newer with mutual TLS for external sources. Implement zero-trust network policies to isolate data ingestion endpoints.
- **Authentication & authorization:** Integrate with centralized IAM, employing short-lived credentials (e.g. AWS STS, GCP IAM) and hardware-backed key storage (HSMs).

Testing:

- **API vulnerability scans:** Apply DAST tools against ingestion endpoints.
- **Fuzzing & threat emulation:** Inject malformed data to test schema validation and error-handling mechanisms.

Data Loaders

What Happens: Ingest validated data into secure pipelines.

Security Controls:

- **Containerized & immutable environments:** Run ETL jobs in hardened, ephemeral containers with pre-approved golden images scanned by SCA tools (Trivy, Gype).
 - **Schema validation & sanitization:** Strictly enforce schemas (e.g. Avro, Protobuf) to prevent injection attacks.
 - **Key management & access control:** Use HashiCorp Vault or cloud-native KMS for credential rotation and key wrapping.
- Testing:**
- **CI/CD integration:** Automate data validation tests in CI pipelines.
 - **Malicious input simulation:** Submit known malicious payloads to confirm detection and rejection.

Data Lake

What Happens: Store large volumes of raw and semi-structured data.

Security Controls:

- **Encryption at rest:** Use AES-256 with hardware-based key management (HSM/KMS). Ensure crypto-agility to support future transitions, including quantum-safe keys.
 - **Fine-grained access controls:** Enforce RBAC/ABAC and row-level security via native cloud services (AWS Lake Formation, Azure Purview).
 - **Least privilege:** Apply zero-trust data plane segmentation. Restrict network paths using service meshes with mTLS (Istio, Linkerd).
- Testing:**
- **Periodic audits & logging:** Regularly review access logs via SIEM. Monitor for unusual read patterns and large-scale exports.
 - **Configuration security checks:** Use IaC scanning (Terraform Scan, CloudFormation Guard) to maintain secure baselines.

Preparation/Computation

What Happens: Transform and clean data, apply privacy-preserving techniques, and prepare training datasets.

Security Controls:

- **Data minimization & masking:** Redact or tokenize PII using NIST pseudonymization standards.
 - **Differential privacy & K-anonymity:** Introduce statistical noise or grouping techniques to prevent re-identification.
 - **Confidential computing:** Employ trusted execution environments (Intel SGX, AMD SEV) to ensure data remains encrypted in-use.
- Testing:**
- **Data leak simulations:** Verify no sensitive fields appear after transformations.

- **Privacy audits:** Regularly assess compliance with GDPR or HIPAA using privacy impact assessment tools.

Data Warehouse

What Happens: Store processed, query-ready datasets.

Security Controls:

- **Granular RBAC & policy enforcement:** Use attribute-based controls and conditional access policies.
- **Integrity & availability:** Checksum entire datasets and maintain cryptographic signatures. Implement geo-redundant backups for resilience.
- **Lifecycle management:** Align data retention, archival, and deletion policies with legal and compliance requirements.
Testing:
- **Penetration tests & red teaming:** Attempt unauthorized queries to confirm privilege boundaries hold.
- **Data integrity verification:** Validate cryptographic checksums and run periodic block-level integrity scans.

Data Sharing

What Happens: Share data internally across teams or externally with partners or analysts.

Security Controls:

- **Secure data-sharing protocols:** Leverage platforms with built-in RBAC, watermarking, and anonymization (Snowflake, BigQuery).
- **Digital watermarking & fingerprinting:** Tag shared datasets for traceability and auditability.
- **Legal & regulatory compliance:** Implement DLP solutions and integrate with Information Rights Management (IRM) to prevent unauthorized exports.
Testing:
- **API security testing:** Evaluate token scopes, rate limits, and response filtering.
- **DLP efficacy checks:** Test for exfiltration attempts and ensure detection/prevention triggers are effective.
- **Federated learning:** Use it to enable privacy while model is updated without data being passed.



Add-Ons

Insight Stores

What Happens: Store derived insights, model outputs, and metadata.

Security Controls:

- **Quantum-resistant cryptography:** Begin preparing for post-quantum standards (e.g. hybrid key exchange schemes) to protect long-lived data.
 - **Immutable ledgers & auditing:** Implement tamper-evident technologies (e.g. blockchain-based logging) for high-integrity audit trails.
- Testing:**
- **Forensic readiness:** Simulate data tampering scenarios to validate the ability to reconstruct event timelines.
 - **Compliance audits:** Map controls to frameworks like ISO/IEC 27001 and ensure continuous attestation.

AI Store

What Happens: Maintain models, associated datasets, code, and configuration for reproducibility and versioning.

Security Controls:

- **Artifact signing & verification:** Use SLSA or Sigstore's Cosign to sign model weights, training code, and config files.
 - **Dependency & vulnerability management:** Continuously scan for CVEs in frameworks (e.g. PyTorch, TensorFlow) and underlying libraries.
 - **Model poisoning prevention:** Maintain a known-good baseline and hash manifests. Continuously compare new artifacts to detect unauthorized changes.
- Testing:**
- **Adversarial robustness testing:** Use adversarial example frameworks (CleverHans, IBM ART) to verify model robustness.
 - **Supply chain security checks:** Periodically verify the provenance of all data and code artifacts in CI/CD.

Additional Recommendations

- **Infrastructure & orchestration security:**
Leverage IaC scanning (tfsec, checkov) and OPA/Gatekeeper policies for Kubernetes to ensure containers and pods adhere to security best practices. Use network policies, eBPF-based runtime analysis, and container image signing (Cosign) to secure workloads.
- **Zero-trust and micro-segmentation:**
Adopt zero-trust principles across LLM pipelines. Require mutual TLS for every service-to-service call, and implement micro-segmentation with service meshes and policy-based access controls.
- **Real-time security analytics:**
Integrate SIEM (Splunk, Elastic SIEM) and XDR platforms to process logs, model outputs, and telemetry. Train anomaly detection systems to spot unusual inference patterns or data requests that may indicate adversarial behavior.
- **Framework alignment & compliance mapping:**
Map each control to recognized frameworks (NIST AI RMF, MITRE ATLAS) and regulations (GDPR, HIPAA), providing a clear compliance roadmap.
- **Continuous testing & model lifecycle management:**
Incorporate continuous integration/testing that includes SAST, DAST, IAST, and software composition analysis. Define secure model update processes and rollbacks to swiftly revert to a known-good model state if compromise is detected.


Data Anonymization and Pseudonymization

Data anonymization and pseudonymization are important techniques for protecting sensitive information when working with large language models (LLMs). In this context, data anonymization involves removing or obfuscating personally identifiable information (PII) from datasets used to train or interact with LLMs. The purpose is preventing re-identification of individuals from the data. It can involve techniques like:

- Removing direct identifiers (names, addresses, etc.).
- Generalizing quasi-identifiers (age ranges instead of exact ages).
- Adding noise to numerical values.
- Suppressing rare or unique data points.

Pseudonymization replaces identifying information with artificial identifiers or pseudonyms. It allows data to be linked to an individual indirectly through additional information kept separately. It can involve techniques like:

- Tokenization - replacing sensitive data with non-sensitive tokens.
- Encryption - encoding identifiers that can only be decrypted with a key.
- Hashing - generating fixed-length codes to represent data.



Aspect	Anonymization	Pseudonymization
Definition	Data is transformed in a way that irreversibly removes identifiers, making it impossible to link data back to an individual.	Data is altered to remove identifiers, but a reversible process allows for re-identification with the use of additional information (e.g. a key).
Purpose	To protect privacy by ensuring that data cannot be traced back to an individual, even with additional information.	To reduce risk of data exposure while still allowing re-identification for legitimate purposes.
Risk Level	Generally considered safer, as the data cannot be easily linked back to individuals.	Lower privacy protection compared to anonymization, as there remains a risk of re-identification if the key is accessed.
Use Cases	Publishing open datasets for research, statistics, or public health purposes.	Internal processes where identifying information may need to be recovered, such as in medical or financial institutions.
Compliance	Often required by data protection laws when releasing data to the public.	Used in compliance with regulations that allow reversible privacy measures for processing.
Data Utility	Often reduced, as linking or correlating records is difficult without identifiers.	Maintains more utility since data can be re-identified for analysis or verification.
Techniques Used	Data masking, data aggregation, or k-anonymity.	Tokenization, encryption, or use of reference tables.
Reversibility	Irreversible; once anonymized, data cannot be restored to its original state.	Reversible with the use of a key or process to re-identify the data.
Example	Removing all personally identifiable information from a dataset used for public research.	Replacing names with unique identifiers in a medical database, with the ability to use a key to restore original data.

Benefits for LLMs:

- Enables use of real-world data for training while protecting privacy.
- Reduces risks of exposing sensitive information in model outputs.
- Helps comply with data protection regulations like GDPR.

Challenges:

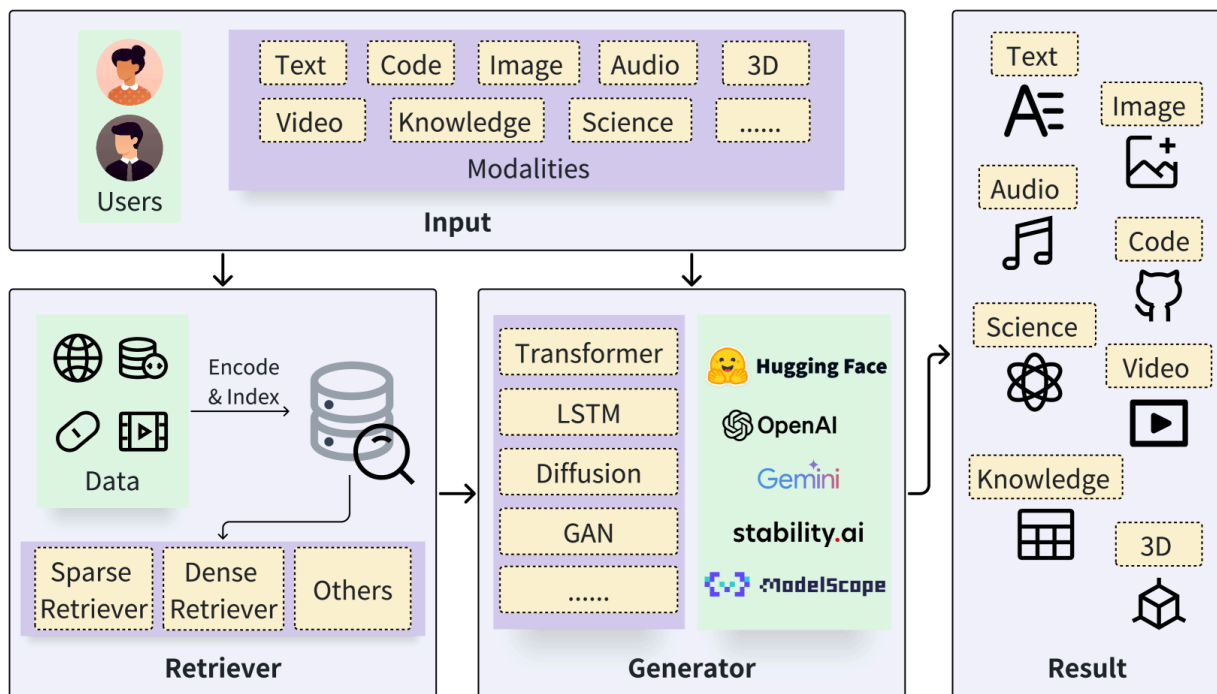
- LLMs may still infer or generate sensitive information from patterns in anonymized data.
- Balancing privacy protection with maintaining data utility for model performance.
- Anonymization can potentially introduce biases or reduce diversity in training data.

- Measuring metrics of anonymization efficiency.

Best Practices:

- Use differential privacy techniques to add statistical noise.
- Implement access controls and encryption for sensitive data.
- Regularly audit models for potential privacy leaks.
- Combine multiple anonymization techniques for stronger protection.
- Consider synthetic data generation as an alternative to using real data.

Overall, robust anonymization and pseudonymization are crucial for responsible development and deployment of LLMs. Their use helps protect individual privacy while leveraging the power of large-scale language data. Ongoing research aims to improve these techniques specifically for LLM applications.



<https://arxiv.org/html/2402.19473v4#S7>

Security in LLM data ecosystems must be comprehensive, spanning from data ingestion to model deployment and insight generation. Cryptographic best practices adoption should include preparing for quantum-resistant standards, while leveraging confidential computing, implementing robust IAM and network segmentation, maintaining a secure software supply chain, and performing continuous adversarial testing. In this manner, organizations can establish a trustworthy and resilient LLM infrastructure.



Integrating these recommendations allows stakeholders—cybersecurity leaders, developers, and CISOs—to align operational needs with stringent security controls and compliance requirements. As a result, they can confidently harness the full potential of LLMs while upholding stringent data protection, privacy, and integrity standards.



Practical LLM Data Security Principles

Securing Large Language Models (LLMs) requires a multi-layered approach that encompasses robust policies, cutting-edge technologies, and continual adaptation to evolving threats. This section provides practical and forward-looking principles to address critical data security challenges and ensure resilient, compliant, and trustworthy LLM operations.

1. Data Minimization and Purpose Limitation

Minimizing data collection and restricting its use to clearly defined purposes are fundamental to protecting sensitive information and ensuring regulatory compliance.

Implementation:

- Define precise data requirements and exclude unnecessary or high-risk datasets.
- Use data solely for specified objectives, enforcing restrictions on secondary use.

Tools & Best Practices:

- Implement differential privacy to anonymize datasets while preserving utility.
 - Use synthetic data where possible to replace sensitive real-world data.
 - Employ tools like Google's Differential Privacy Library or Snorkel for dataset preparation.
-

2. Robust Data Encryption

Encryption ensures data remains secure throughout its lifecycle—during storage, transmission, and processing.

Implementation:

- Encryption in transit: Use TLS 1.3 for securing API communications and external data flows.
- Encryption at rest: Apply AES-256 encryption for data storage, including backups.
- Encrypted computation: Leverage homomorphic encryption or secure multi-party computation for sensitive processing tasks.

Tools & Best Practices:

- Integrate AWS KMS, Azure Key Vault, or HashiCorp Vault for secure key management.
 - Automate encryption audits to detect misconfigurations and unauthorized access.
-

3. Granular Access Controls

Enforce strict role-based and attribute-based access control (RBAC and ABAC) to ensure only authorized users can access sensitive LLM data and systems.

Implementation:

- Assign roles based on least privilege principles.
- Enforce MFA for accessing critical systems and APIs.

Tools & Best Practices:

- Use identity management platforms like Okta or Azure Active Directory.
 - Regularly review and revoke access rights as roles or responsibilities change.
 - Employ tools such as AWS IAM Access Analyzer to automatically review your permissions.
-

4. Comprehensive Input Validation

Validating and sanitizing inputs protects against injection attacks and malicious data exploitation.

Implementation:

- Implement input filters to remove commands that could manipulate model behavior.
- Restrict inputs to predefined formats and types.

Tools & Best Practices:

- Use regex-based validation for structured inputs.
 - Deploy OWASP ZAP to test input vulnerabilities in pre-production environments.
 - Leverage LLMs to perform input fuzzing testing to increase your test coverage.
-

5. Output Monitoring and Moderation

LLMs can inadvertently generate sensitive, harmful, or unauthorized content. Moderating outputs helps prevent misuse and compliance violations.

Implementation:

- Use dynamic filters to flag sensitive data, misinformation, or harmful content in outputs.
- Implement post-processing tools to anonymize or mask flagged outputs before distribution.



Tools & Best Practices:

- Integrate the OpenAI Moderation API or Azure Content Safety, Azure Purview, Protect AI MLSecOps tools.
 - Continuously refine moderation algorithms with adversarial testing.
-

6. Privacy-Enhancing Technologies (PETs)

Privacy-enhancing technologies safeguard user data while maintaining model utility. A list of PETs will be provided on the future trends chapter.

Implementation:

- Use federated learning to train models without exposing raw data.
- Implement secure multi-party computation for collaborative training across organizations.

Tools & Best Practices:

- Leverage libraries like PySyft for privacy-preserving workflows.
 - Regularly evaluate PET configurations to adapt to emerging threats.
-

7. Secure Development and Deployment

Security must be integral to the entire LLM lifecycle, from development to deployment.

Implementation:

- Use isolated development environments (e.g. containers) for training and fine-tuning.
- Deploy LLMs in secure runtime environments with network segmentation.

Tools & Best Practices:

- Use Docker or Kubernetes for isolated deployments.
 - Conduct pre-deployment audits with vulnerability scanners like Nessus or Aqua Security.
-

8. Incident Response and Breach Preparedness

Effective incident response minimizes the impact of data breaches or security incidents.

Implementation:

- Develop a detailed incident response plan tailored to LLM-specific risks.
- Conduct regular breach simulations and tabletop exercises.

Tools & Best Practices:

- Use monitoring platforms like Splunk or IBM QRadar for real-time breach detection.

- Maintain compliance with regulations like GDPR's 72-hour breach notification rule.
-

9. Supply Chain Security

Third-party dependencies introduce vulnerabilities in LLMs. Securing the supply chain ensures resilience.

Implementation:

- Audit pre-trained models and third-party libraries for vulnerabilities.
- Use only verified and signed packages.

Tools & Best Practices:

- Employ Snyk, GitHub Dependabot, or OWASP Dependency-Check for continuous monitoring.
 - Set up automated alerts for updates to critical dependencies.
 - Sign packages/containers with in-toto or cosign
 - Create Software Attestations with Build Providence to implement an appropriate SLSA level
-

10. Continuous Monitoring and Auditing

Ongoing monitoring ensures LLM systems adapt to evolving threats and remain compliant with policies.

Implementation:

- Monitor all interactions with the LLM, tracking input, output, and API usage. Configure appropriate Access Controls for logs access to avoid data leakage.
- Conduct regular security audits to identify misconfigurations or gaps.

Tools & Best Practices:

- Deploy SIEM tools like Splunk, ELK Stack, or Azure Sentinel.
- Perform quarterly penetration tests with tools like Burp Suite or Metasploit.

Monitoring and Auditing Guidelines

The following suggested audit guidelines table is organized by risk level, starting with high, and then by audit frequency within each risk level:

Type of Audit	Audit Frequency	Risk Level	Benefits
Input Data Audits (LLMs)	Weekly	High	Identify and prevent biases, ensure data quality, and maintain relevance of training data.
Log Monitoring and Analysis (Traditional)	Weekly	High	Identify suspicious activities, ensure system stability, and detect potential breaches early.
Vulnerability Assessment (LLMs)	Monthly	High	Detects and remediates potential security flaws in the model infrastructure.
Network Security Audit (Traditional)	Monthly	High	Evaluate the security of network infrastructure and ensure robust defenses.
Incident Response Drill (LLMs)	Annually	High	Prepare for potential security incidents and ensure the effectiveness of response plans.
Data Privacy Audit (LLMs)	Annually	High	Assess data handling practices to protect user privacy and comply with data protection laws.
Penetration Testing (LLMs)	Quarterly	Medium	Simulate attacks to test the security posture and identify exploitable vulnerabilities.
Compliance Audit (LLMs)	Quarterly	Medium	Ensure adherence to regulations like GDPR or HIPAA and avoid legal issues.
Access Control Review (LLMs)	Annually	Medium	Verify that only authorized personnel have access to sensitive data and resources.
Patch Management Audits (Traditional)	Quarterly	Medium	Ensure timely updates of software, reduce vulnerabilities, and maintain system integrity.
Business Continuity and Disaster Recovery Testing (Traditional)	Annually	Medium	Validate recovery procedures, ensure business resilience, and minimize downtime during incidents.
Access Control Review (Traditional)	Annually	Medium	Assess permissions and privileges to prevent unauthorized access.
Configuration Audit (LLMs)	Bi-Annually	Low	Review system settings to maintain optimal security configurations.
Configuration Audit (Traditional)	Quarterly	Low	Review and optimize hardware and software configurations for security.



11. Regulatory Compliance and Governance

Compliance with global data protection laws ensures LLMs operate within legal frameworks and build trust.

Implementation:

- Align security measures with GDPR, CCPA, and ISO/IEC 27001 among others.
- Conduct privacy impact assessments (PIAs) to evaluate risks and controls.

Tools & Best Practices:

- Use platforms like OneTrust for automating compliance tracking.
 - Develop internal governance teams to oversee AI ethics and security policies.
-

12. Ethical and Responsible AI Governance

Establishing governance frameworks ensures ethical, secure, and transparent LLM deployment.

Implementation:

- Define ethical guidelines for AI use aligned with organizational values.
- Document training data sources, biases, and usage limitations.

Tools & Best Practices:

- Use frameworks like the OWASP Top 10 for LLM Applications to guide governance efforts.
- Regularly update governance policies based on technological advancements.

Organizations following these principles can address the multifaceted challenges of securing LLMs. Adopting a layered approach that integrates advanced technologies, proactive governance, and continuous monitoring ensures the protection of sensitive data, regulatory compliance, and trustworthiness of AI systems in an increasingly complex threat landscape.



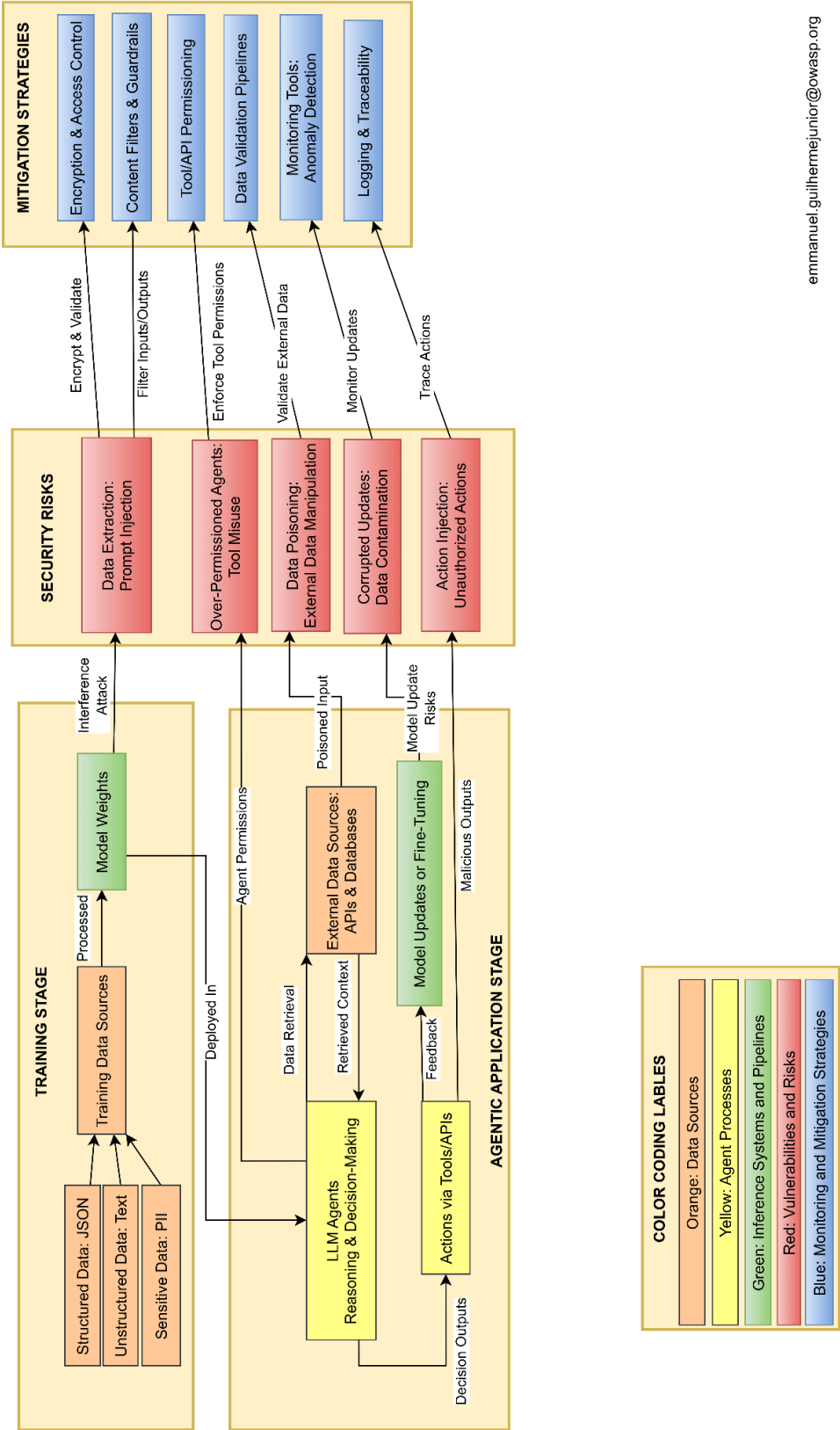
Securing Data Flows in LLM Agents

LLM Agents are stateful, autonomous components that leverage Large Language Models to interpret instructions, reason through multi-step tasks, access external tools, and maintain evolving internal memory or context. While these capabilities enable powerful automation and decision-making, they also expand the potential attack surface. Threats include prompt injection, data poisoning, malicious tool usage, and inadvertent data leakage. It appears that in 2025, AI agents and agentic applications will sharply accelerate.

This section provides a technical blueprint for securing data in LLM Agents. It maps recommendations to industry standards, and offers concrete configurations, tools, and policies to ensure data confidentiality, integrity, and compliance with frameworks such as the NIST AI Risk Management Framework (AI RMF) and MITRE ATLAS for adversarial ML tactics.

Underneath is a diagram illustrating the data flow and stages in LLM agents.

Data Security in LLM Agents and Agentic Applications



emmanuel.guilhermejunior@owasp.org



Key LLM Security Objectives

1. **Integrity of Inputs and Outputs**
Ensure that agent prompts, policies, and responses remain free from tampering or injection attempts.
 2. **Confidentiality of Sensitive Data**
Protect personally identifiable information (PII), proprietary content, and regulated data (GDPR, HIPAA) from unauthorized exposure.
 3. **Least-Privilege Access to Tools and Data**
Grant the agent minimal permissions, tightly scope API keys, and enforce micro-segmentation so that the agent can only interact with explicitly authorized data sources and actions.
 4. **Traceability and Compliance Alignment**
Maintain robust, tamper-evident logs and reference authoritative frameworks (e.g., NIST AI RMF) to ensure continuous compliance, and support rapid incident response and forensic analysis.
-

Data Flow Stages in LLM Agents

1. **Prompt/Instruction Ingestion:** The agent receives and processes user instructions.
2. **Context Retrieval & Memory Management:** The agent queries vector stores, knowledge graphs, or databases to gather relevant facts.
3. **Tool Invocation:** The agent leverages external APIs, scripts, or plugins to fetch additional data or perform actions.
4. **Stateful Reasoning & Output Generation:** The agent stores intermediate reasoning steps and produces a final, contextually-rich response.

Each stage poses distinct security risks and requires targeted controls.

Threat Models & Attack Vectors

- **Prompt Injection Attacks**
Adversaries craft inputs that override agent policies or cause data leakage.
MITRE ATLAS Reference: Prompt manipulation aligns with adversarial technique TA0040.
- **Data Poisoning (Compromised Vector Stores)**
Attackers insert malicious embeddings or altered facts, skewing the agent's reasoning.
NIST AI RMF: Emphasize data provenance and model robustness controls.

¹ <https://github.com/emmanuelgjr/2025WPdatasecurity/blob/main/PoCAgentDataSecurity.md>

- **Tool Misuse**

The agent might be tricked into invoking forbidden APIs or performing restricted actions.

Compliance Note: GDPR and HIPAA demand strict controls over data access and disclosure.

- **Inadvertent Data Leakage**

Sensitive data stored in agent memory surfaces in outputs, violating privacy regulations.

Security Controls for LLM Agents

Prompt and Instruction Security

- **Contextual Sanitization & Policy Embedding**

- **Implementation:** Pre-process user prompts with sanitizers that strip credentials, PII, and known disallowed patterns.
- **Policy embedding:** Incorporate internal compliance and security policies directly into system-level prompts. For instance, “NEVER output PII” is embedded at a high-priority level.
- **Integrity verification:** Apply HMAC (e.g., HMAC-SHA-256) to system prompts. If the system prompt is tampered with, the agent aborts.

- **Adversarial Prompt Testing**

- **Tools:** Use AI-specific fuzzing frameworks and prompt-injection simulation toolkits to test resilience against crafted malicious inputs.
- **Continuous integration (CI):** Integrate prompt security tests into CI/CD pipelines, rejecting new code that weakens prompt integrity protections.

Memory and State Management

- **Encrypted & Isolated Storage**

- **Confidential computing:** Store vector embeddings and chain-of-thought data inside TEEs (Intel SGX, AWS Nitro Enclaves) or use transparent database encryption managed by a cloud KMS.
- **Access control:** Implement Attribute-Based Access Control (ABAC) at the vector store layer. For example, only the “Researcher” role can access embeddings tagged with “de-identified_PII.”
- **Periodic integrity checks:** Hash embeddings (SHA-256) and periodically verify them to detect tampering. Use Merkle trees for scalable integrity checks.

- **Data Minimization & Redaction**

- **Implementation:** Before storing user data, redact or tokenize sensitive fields.
- **Compliance mapping:** Aligned with GDPR’s data minimization principle and HIPAA’s de-identification standard.



Secure Integration With External Tools and Data Sources

- **Scoped and Rotated Credentials**
 - **Short-lived tokens:** Provide the agent only with narrowly scoped OAuth2 tokens or API keys that expire frequently.
 - **IAM integration:** Configure enterprise IAM solutions (e.g., AWS IAM, GCP IAM) to dynamically grant and revoke permissions based on real-time policies.
- **Tool Execution Sandboxes**
 - **Containerization & isolation:** Run tool integrations in locked-down containers (gVisor, Kata Containers) with no outbound internet access except to whitelisted APIs.
 - **Schema validation:** Enforce strict JSON schema validation on all tool outputs. If the tool returns unexpected fields, abort processing and raise an alert.
- **Vector Store Security**
 - **Encrypted queries:** Use client-side encryption or mTLS for queries to vector stores.
 - **Authenticated data retrieval:** Require that each query include a cryptographic token ensuring the request comes from an authorized agent instance.

Cryptographic and Post-Quantum Readiness

- **Quantum-Resistant Key Management**
 - **Hybrid KEM approaches:** Explore hybrid key exchange (e.g. combining ECDH with a NIST PQC candidate) for long-lived, sensitive agent memory.
 - **Crypto-agility:** Maintain versioned keys and support rolling upgrades to new ciphers. Periodically rotate keys managed by KMS solutions, and re-encrypt stored embeddings as required.
 - **Artifact Signing & Integrity**
 - **Sigstore integration:** Use Cosign to sign prompt templates, policy files, and tool scripts. On startup, the agent verifies signatures.
 - **Tamper-evident logs:** Apply cryptographic timestamping (RFC3161) and blockchain or immutability services (immudb) to ensure that logs are tamper-evident.
-

Testing and Verification

- **Red Teaming & Adversarial Simulation**

Emulate known adversarial techniques from MITRE ATLAS. Test prompt injection, poisoning attempts, and unauthorized tool invocation. Record findings and integrate improvements.
- **SAST, DAST & IAST in CI/CD**
 - **SAST:** Use CodeQL to scan agent code for logic flaws that could enable data exposure.
 - **DAST:** Test APIs the agent uses with OWASP ZAP or Burp Suite.

- **IAST:** Runtime instrumentation tools ensure that agent behavior matches expected security policies during integration tests.
 - **Behavioral Monitoring & Anomaly Detection**
 - **SIEM/XDR integration:** Forward all logs and telemetry (agent requests, tool outputs, vector store queries) to SIEM solutions (Splunk, Elastic). Use ML-based anomaly detection to flag unusual access patterns or output distributions.
 - **Chaos Engineering for Security**

Introduce controlled faults: corrupt an embedding index or provide malformed tool outputs, ensuring the agent's security controls catch and mitigate these anomalies.
-

Auditing, Traceability, and Governance

- **Comprehensive Structured Logging**
 - **Schema-based logging:** Store all events as JSON with fields such as request_id, user_role, data_sensitivity_level, and tool_invoked.
 - **Immutable storage:** Append logs to tamper-evident storage (WORM or blockchain-based), enabling forensic reconstructions.
 - **Forensic Readiness**
 - **Event correlation:** Use correlation IDs across prompts, memory retrievals, and tool calls. If an incident occurs, reconstruct the agent's entire decision chain.
 - **Compliance Mapping**
 - **GDPR/CCPA:** Demonstrate that personal data is encrypted, minimized, and access-controlled.
 - **HIPAA:** Show that PHI is never output without authorization, and memory with PHI is stored in FIPS-validated encryption modules.
-

Policy Enforcement and Framework Alignment

- **Policy-as-Code with OPA**

Define OPA or Conftest policies that restrict the agent's capabilities. For example, a policy might state: "The agent can only call getCustomerData() if the request is during business hours and includes a valid JWT from the HR system."
- **Standards and Frameworks**
 - **NIST AI RMF:** Align controls with the NIST AI RMF's guidance on trustworthy and transparent AI systems.
 - **MITRE ATLAS:** Continuously monitor for tactics and techniques related to data poisoning, prompt injection, and model evasion.
 - **SLSA:** Supply-chain Levels for Software Artifacts



Performance and Scalability Considerations

- **Latency Management**
Use ephemeral memory caches, hardware acceleration (HSMs or TEEs), and efficient encryption algorithms to reduce added latency from security controls.
 - **Scalable Security**
Distribute cryptographic operations and IAM checks across multiple nodes. Implement autoscaling for security services (e.g., dedicated KMS proxies) to maintain throughput.
-

Scenario Example

Use Case: Customer Support Agent

- **Data:** Customer ticket info, possibly containing PII.
 - **Controls:** The agent's prompt policy forbids outputting exact PII. The vector store embedding is encrypted and only accessible with a time-limited token. Tool integration (CRM API) uses a read-only API key with IP allowlisting.
 - **Testing & Compliance:** Red team attempts prompt injection to extract sensitive details fail because the agent's prompt includes strict redaction policies and the schema validation rejects abnormal tool outputs.
-

By integrating robust cryptographic protections, least-privilege policies, sandboxed tool execution, alignment with recognized security frameworks, and continuous testing, organizations can confidently operate LLM Agents that handle sensitive data securely and compliantly. This comprehensive, standards-aligned approach ensures that as LLM Agents grow more capable, they remain trustworthy, resilient, and aligned with both technical best practices and regulatory mandates.



Future Trends & Challenges in LLM Data Security

The evolution of Large Language Models (LLMs) is transforming industries, yet it also presents complex data security challenges. Addressing these challenges calls for advanced techniques, vigilance with respect to regulatory developments, and foresight regarding emerging threats. This section examines key trends, practical technical insights, and strategies for securing LLMs in 2025 and beyond. Although the future is difficult to predict, we aim to highlight the most promising developments shaping the field.

1. Emerging Privacy-Enhancing Technologies (PETs)

Technical Enhancements:

- **Homomorphic Encryption**
 - Tool: Microsoft SEAL or IBM HELib for performing encrypted computations.
 - Insight: While computationally intensive, advancements in hybrid encryption (partially homomorphic combined with symmetric encryption) are reducing overhead.
- **Federated Learning**
 - Tool: TensorFlow Federated or PySyft.
 - Insight: Techniques like adaptive federated averaging and differential privacy integration enhance scalability and security for distributed datasets.
- **SMPC**
 - Tool: MP-SPDZ for multi-party computation protocols.
 - Insight: Combine SMPC with oblivious transfer techniques to minimize bandwidth requirements during computations.



PETs Advanced Use Case:

Privacy-preserving multi-institutional AI: Hospitals collaboratively train LLMs for diagnostics without sharing raw patient data using federated learning combined with differential privacy.

See a list of promising PETs:

- | | |
|---|---|
| 1. Differential Privacy | 14. Privacy-Preserving Data Sharing Protocols |
| 2. Federated Learning | 15. Encrypted Computation Frameworks |
| 3. Homomorphic Encryption | 16. Data Minimization Principles |
| 4. Secure Multi-Party Computation (SMPC) | 17. Federated Analytics |
| 5. Data Anonymization and Masking | 18. Privacy-Aware Machine Learning Models |
| 6. Access Control and Audit Trails | 19. Policy-Based Data Access |
| 7. Ethical Guidelines and Compliance Frameworks | 20. Secure Enclaves in Cloud Computing |
| 8. Trusted Execution Environments (TEEs) | 21. Encrypted Model Parameters |
| 9. K-Anonymity and L-Diversity | 22. Privacy-Preserving Record Linkage |
| 10. Synthetic Data Generation | 23. Use of On-Device Processing |
| 11. Privacy-Preserving Machine Learning Frameworks | 24. Consent Management Platforms |
| 12. Zero-Knowledge Proofs (ZKPs) | 25. Anomaly Detection for Privacy Breaches |
| 13. Differentially Private Stochastic Gradient Descent (DP-SGD) | 26. Secure Logging and Monitoring |
| | 27. Endpoint Security Measures |
| | 28. Integration of Privacy by Design Principles |
| | 29. Privacy Impact Assessments (PIAs) |

PETs Case Studies:

- **Google's federated learning in mobile applications:** Improved models on devices without compromising user data.
 - **OpenAI's policies for GPT-3:** Implemented guidelines to prevent the generation of sensitive or personal information.
 - **IBM's homomorphic encryption research:** Advanced techniques for encrypted computations in AI systems.
 - **Healthcare federated learning consortiums:** Hospitals collaborating on AI models without sharing patient data.
-



2. Advanced Adversarial Threats

Technical Enhancements:

- **Model Extraction Detection**
 - Insight: Implement watermarking systems that embed unique identifiers in model outputs, detectable during unauthorized model usage.
 - Tool: Use OpenMined for embedding imperceptible watermarks.
- **Dynamic Defense Against Adversarial Attacks**
 - Insight: Deploy adversarial training with on-the-fly perturbation injection to improve model robustness.
 - Tool: Libraries like ART (Adversarial Robustness Toolbox) for generating and defending against adversarial examples.
- **Automated Threat Modeling**
 - Tool: Microsoft Threat Modeling Tool to simulate attack vectors like prompt injection or poisoning.

Advanced Use Case:

- **Active Defense:** Implementing honeytokens within training data to detect and trace back unauthorized model queries or extractions.
-

3. Regulatory Complexity and Compliance

Technical Enhancements

- **Automated Compliance Validation**
 - Tool: OneTrust Data Mapping for continuous alignment with GDPR, CCPA, and upcoming AI Act requirements.
 - Insight: Combine compliance tools with natural language processing (NLP) models to automatically flag non-compliant clauses in agreements or datasets.
- **Explainable Compliance Mechanisms**
 - Insight: Use explainable AI (XAI) frameworks like SHAP or LIME to provide regulators with transparency into decision-making processes.

Advanced Use Case

- **Real-time cross-border compliance:** Implementing AI-driven dynamic data routing systems that ensure LLM data storage and processing stay within jurisdictional boundaries.
-



4. Scalability of Security Solutions

Technical Enhancements

- **Serverless Security:**
 - Insight: Use serverless frameworks like AWS Lambda integrated with AWS Macie for scalable data protection and anomaly detection.
- **Dynamic Workload Segmentation**
 - Tool: VMware NSX or Cisco Tetration for micro-segmentation of data flows between model training and inference.
- **Scaling Encryption**
 - Insight: Transition to lightweight encryption schemes like ChaCha20-Poly1305 for low-latency environments.

Advanced Use Case:

- **On-the-fly data encryption:** Using cloud-based real-time encryption for sensitive streaming data in multi-cloud hybrid deployments.
-

5. Integration with Critical Infrastructure

Technical Enhancements:

- **Zero Trust in AI Workflows**
 - Insight: Extend Zero Trust principles to LLM pipelines using real-time identity verification and dynamic policy enforcement.
 - Tool: Use Google BeyondCorp for end-to-end Zero Trust architecture.
- **Domain-Specific LLMs**
 - Insight: Use pre-trained domain-specific models fine-tuned in isolated, air-gapped environments to prevent cross-domain contamination.
 - Example: Healthcare-specific LLMs secured with FHIR-compliant data handling.

Advanced Use Case:

- **LLMs in public safety:** Deploying secure LLMs for emergency response systems that integrate live data streams with strict regulatory oversight.
-



6. Ethical and Responsible AI Development

Technical Enhancements:

- **Bias Detection and Mitigation**
 - Tool: AI Fairness 360 (AIF360) or Fairlearn for auditing datasets and outputs.
 - Insight: Combine unsupervised clustering with explainable AI techniques to detect hidden biases in large datasets.
- **Reproducibility**
 - Insight: Use containerization with tools like Docker and MLflow to ensure consistent, auditable AI workflows.

Advanced Use Case

- **Proactive bias mitigation:** Real-time auditing of LLM outputs for fairness compliance using integrated feedback loops from diverse user inputs.
-

7. Proliferation of AI-as-a-Service Models

Technical Enhancements

- **API Security**
 - Tool: Use API Gateway services with OAuth 2.0 and JSON Web Tokens (JWT) for secure LLM API interactions.
 - Insight: Implement dynamic throttling and rate limiting to prevent API abuse.
- **Third-Party Plugin Security**
 - Insight: Use static analysis tools like SonarQube to scan plugins for vulnerabilities before integration.

Advanced Use Case

- **Tokenized API access:** Offering granular access levels to LLMs using tokenization, ensuring minimal privileges for third-party plugins.
-



8. Future Innovations in Data Security

Technical Enhancements:

- **Quantum-Resilient Cryptography**
 - Insight: Adopt lattice-based cryptographic protocols like NTRU and Kyber for long-term data security. NIST's PQC algorithm: <https://csrc.nist.gov/projects/post-quantum-cryptography>
 - Tool: OpenQuantumSafe (OQS) for exploring and implementing post-quantum encryption schemes.
- **Self-Healing AI Systems**
 - Insight: Implement self-healing models that automatically retrain to counteract detected poisoning attacks.
 - Tool: Leverage TensorFlow Model Remediation for dynamic adjustments in response to identified vulnerabilities.
- **AI-Driven Incident Response**
 - Tool: Use Cortex XSOAR for automated incident response driven by AI-powered playbooks.

Advanced Use Case

- **Quantum-secure LLM pipelines:** Deploying end-to-end quantum-resilient workflows for critical infrastructures, ensuring forward compatibility with quantum computing advancements.

The future of LLM data security lies in integrating advanced technologies with adaptive strategies to mitigate evolving threats. Collaboration among cross-functional teams is paramount addressing data security. The use of cutting-edge tools, embracing regulatory frameworks, and fostering collaboration across sectors will enable organizations to keep their LLM deployments secure, compliant, and impactful. The ability to proactively address these trends will define leaders in the AI security domain, paving the way for trust and innovation in the age of intelligent systems. Constant adaptability to the ever evolving threat landscape and the use of new products/services is the best approach to protect your data in LLM environments.



Secure Deployment Strategies for LLMs

Deploying Large Language Models (LLMs) requires robust strategies to address security, performance, and compliance. As deployment scenarios grow more complex with on-premises, cloud, and hybrid environments, organizations must implement scalable and secure architectures tailored to their operational needs. This section outlines practical and technical deployment strategies to safeguard LLMs while maximizing efficiency and trust.

Aspect	On-Premises	Cloud	Hybrid
Data Control	High	Low	Medium
Scalability	Low	High	Medium
Cost	High upfront, Low ongoing	Low upfront, Variable ongoing	Medium upfront, Medium ongoing
Compliance	Easier for strict regulations	May require additional measures	Flexible, but complex
Maintenance	Full responsibility	Managed by provider	Shared responsibility
Network Security	Internal focus	Provider + Customer responsibility	Complex, requires careful design

When deploying large language models (LLMs), organizations must carefully evaluate different architectural options.

1. On-Premises Deployment

On-premises deployments are ideal for organizations handling highly sensitive data or operating under strict regulatory requirements. These environments provide maximum control over infrastructure but demand rigorous security practices.

Technical Strategies

- **Isolated Compute Environments**
 - Deploy LLMs in air-gapped systems to eliminate exposure to external networks.
 - Use tools like VMware vSphere or Red Hat OpenShift to create isolated virtual environments.

- **Hardware Security Modules (HSMs)**
 - Use HSMs to securely store encryption keys for protecting model weights and data.
 - Tools: AWS CloudHSM (on-premises-compatible) or Thales Luna HSM.
- **Secure Model Serving**
 - Use inference-serving frameworks such as NVIDIA Triton to optimize performance while ensuring secure access.

Gold Nuggets:

- Implement role-based segmentation for accessing compute resources, ensuring only authorized personnel interact with the system.
- Use Intel SGX (Software Guard Extensions) to isolate sensitive model computations within secure enclaves.

On-Premises Deployment Checklist	
Aspect	On-Premises
Data Security	Encrypt data at rest and in transit; limit physical access to servers; implement DLP
Access Control	Use role-based access control (RBAC); enforce MFA; audit and review access logs
Network Security	Secure the internal network; apply firewalls and intrusion detection; isolate LLM servers
Compliance and Governance	Ensure adherence to local regulations; regularly audit for regulatory compliance
Backup and Disaster Recovery	Schedule regular backups; test disaster recovery procedures
Model Security	Secure model files and configurations; regularly update and patch software; conduct security audits
Operational Security	Perform regular security assessments; train staff on secure LLM practices
Incident Response	Develop an on-prem incident response plan; conduct regular incident response drills

2. Cloud-Based Deployment

Cloud-based deployments enable scalability and flexibility but require a comprehensive approach to mitigate risks associated with multi-tenant environments.

Technical Strategies:

- **Zero Trust Architecture:**
 - Implement identity and access management (IAM) with tools like AWS Identity Center or Azure AD Conditional Access.
 - Enforce dynamic authentication policies based on user behavior and location.
- **Encryption in Multi-Tenant Environments:**
 - Use AWS KMS or Google Cloud KMS for encrypting data and model weights.
 - Employ envelope encryption to manage encryption keys securely.
- **Serverless Deployment for Inference:**
 - Use serverless solutions like AWS Lambda or Google Cloud Functions to dynamically scale inference workloads while maintaining minimal attack surfaces.

Gold Nuggets:

- Leverage confidential computing with services like Azure Confidential VMs, NVIDIA Confidential Computing or Google Cloud Confidential Computing to secure data and computation in shared environments.
- Use secure container orchestration with Kubernetes tools like Kyverno for enforcing pod-level security policies.

Cloud Security Deployment Checklist	
Aspect	Cloud
Data Security	Ensure end-to-end encryption; use cloud provider's encryption services; enable DLP
Access Control	Configure IAM; use cloud-native security tools; monitor and audit cloud access logs
Network Security	Use VPC settings; implement network security groups; enable DDoS protection
Compliance and Governance	Verify compliance with cloud-specific standards; use compliance tools from the vendor
Backup and Disaster Recovery	Utilize cloud backup solutions; plan for data restoration in case of outage
Model Security	Use cloud-native model security features; enable automated updates; restrict API access
Operational Security	Use cloud monitoring tools; keep cloud security certifications up-to-date
Incident Response	Use cloud-native incident response tools; automate threat detection and response

3. Hybrid Deployment

Hybrid deployments combine the control of on-premises systems with the scalability of the cloud, offering flexibility for diverse workloads.

Technical Strategies

- **Data Partitioning**
 - Split sensitive workloads between on-premises systems and less critical tasks in the cloud.
 - Use Apache Kafka for real-time data synchronization between environments.
- **Edge Computing for LLMs**
 - Deploy lightweight versions of LLMs at the edge using frameworks like ONNX Runtime.
 - Ensure local encryption with hardware-accelerated tools such as Intel QAT (QuickAssist Technology).
- **Hybrid Identity Management**
 - Centralize authentication across on-premises and cloud environments with solutions like Okta Hybrid Access.

Gold Nuggets:

- Implement multi-cloud key orchestration using tools like HashiCorp Vault to unify key management across hybrid setups.
- Optimize hybrid workloads with AWS Outposts or Azure Arc to extend cloud services into on-premises environments securely.

Hybrid Security Checklist	
Aspect	Hybrid
Data Security	Implement encryption for both on-prem and cloud; review encryption policies; monitor data flows
Access Control	Integrate IAM across environments; apply consistent access controls; consolidate logs
Network Security	Secure communication between environments; use VPNs/firewalls; monitor for unusual activity
Compliance and Governance	Maintain cross-architecture governance policies; create a compliance management plan
Backup and Disaster Recovery	Implement a hybrid backup strategy; test failover processes regularly
Model Security	Protect models with encryption; sync updates; implement unified monitoring and alerts
Operational Security	Deploy cross-environment assessments; train teams on hybrid complexities
Incident Response	Integrate response plans across platforms; test hybrid incident response scenarios



4. API Security for LLM Deployment

APIs are the primary interface for interacting with LLMs, making their security crucial to prevent unauthorized access, misuse, and data breaches.

Technical Strategies

- **Authentication and Authorization**
 - Use OAuth 2.0 and JSON Web Tokens (JWT) to secure API access.
 - Implement fine-grained authorization policies with tools like Kong Gateway or NGINX API Gateway.
- **Rate Limiting and Throttling**
 - Prevent abuse with dynamic rate limiting using tools like Cloudflare Rate Limiting.
- **Input Sanitization**
 - Filter API inputs for malicious payloads using regex and pre-defined templates.
 - Automate threat detection with OWASP ModSecurity.

Gold Nuggets

- Implement per-client API keys with unique scopes to enforce granular access control.
 - Monitor API usage patterns with tools like Splunk Observability or Datadog API Monitoring to detect anomalies in real time.
-

5. Model Monitoring and Logging

Continuous monitoring of LLMs in production ensures operational reliability and detects anomalous behavior.

Technical Strategies

- **Telemetry and Metrics Collection**
 - Use Prometheus or Grafana Loki to collect and visualize performance metrics.
 - Focus on latency, throughput, and error rates for real-time insights.
- **Anomaly Detection**
 - Deploy AI-driven monitoring systems like Dynatrace to identify irregular input or output patterns.
 - Use time-series anomaly detection with libraries like Kats (Facebook).
- **Audit Logging**
 - Record all API interactions, model inferences, and administrative activities.
 - Store logs securely in tamper-proof systems like Amazon S3 with Object Lock or Azure Data Lake.



Gold Nuggets

- Integrate Explainability Dashboards using frameworks like SHAP to correlate performance issues with specific model behaviors.
 - Enable dynamic logging to scale the level of detail captured based on observed anomalies, reducing noise without losing key insights.
-

6. Edge Security for Decentralized Deployments

Deploying LLMs on edge devices introduces unique challenges due to limited resources and exposure to physical tampering.

Technical Strategies

- **Model Compression**
 - Use quantization techniques with tools like TensorRT to optimize model size for edge deployment.
- **Secure Boot and Firmware Protection**
 - Ensure devices boot only trusted firmware using TPM (Trusted Platform Module).
- **Local Data Encryption**
 - Protect data on edge devices with lightweight cryptographic schemes such as AES-GCM.

Gold Nuggets

- Combine edge computing with federated learning to enable decentralized training without exposing raw data.
 - Deploy geo-fencing policies to restrict edge model usage to specific physical locations.
-

7. Future-Proofing Deployment Strategies

As AI technologies evolve, deployment strategies must adapt to emerging threats and innovations.

Technical Strategies

- **Quantum-Resistant Encryption**
 - Transition to quantum-safe algorithms such as NTRU or SIKE for long-term security.
- **Self-Healing Deployments**
 - Implement systems that automatically roll back to secure states upon detecting anomalies.
 - Use Kubernetes' Helm for automated rollbacks and updates.



Gold Nuggets

- Integrate AI-driven vulnerability management platforms like Tenable.io for proactive risk mitigation.
- Leverage AI Orchestration tools to automate model updates, scaling, and security patching dynamically.

Securing LLM deployments requires a multifaceted approach tailored to the operational environment. Robust adoption of on-premises, cloud, hybrid, and edge strategies, allows organizations to mitigate risks while harnessing the full potential of their LLMs. Proactive measures, such as dynamic monitoring, advanced encryption, and adaptive security policies, ensure that deployments remain resilient against evolving threats. These strategies not only protect sensitive data but also provide the foundation for scaling LLMs securely in the future while recognizing there is not a “one size fits all” solution for every organization.



LLM Data Security Governance

Governance is critical for ensuring the secure, ethical, and compliant deployment of Large Language Models (LLMs). A robust governance framework establishes clear policies, accountability structures, and mechanisms for oversight, enabling organizations to protect sensitive data, mitigate risks, and maintain trust. This section outlines advanced strategies for LLM governance, incorporating technical tools, metrics, and real-world scenarios to address evolving challenges.

1. Establishing Data Security Policies

Comprehensive policies guide the secure management of data throughout the LLM lifecycle. These policies must address data privacy, acceptable use, and regulatory compliance while considering the ethical implications of AI deployment.

Key Components

- **Data Privacy and Access**
 - Restrict data access to authorized users with role-based access control (RBAC).
 - Enforce data minimization practices to reduce exposure to sensitive information.
 - Tools: Implement RBAC with platforms like Okta or Azure Active Directory.
- **Acceptable Use Policies**
 - Define permitted LLM applications (e.g., customer support) and explicitly prohibit high-risk use cases (e.g. decision-making in critical infrastructure without human oversight).
- **Data Retention and Lineage**
 - Maintain records of data origins, transformations, and uses to ensure traceability.
 - Tools: Use Apache Atlas or DataHub for data lineage tracking.

Real-World Example

- A healthcare provider implemented strict data minimization policies using synthetic data generation for training LLMs, ensuring compliance with HIPAA while preserving data utility.

Best Practices

- Automate policy enforcement using machine-readable policies with tools like Open Policy Agent (OPA).
 - Conduct annual policy reviews to align with emerging regulations such as the EU AI Act or NIST AI RMF.
-



2. Defining Accountability and Oversight Structures

Governance frameworks require clear accountability to enforce policies and respond to violations. Oversight structures ensure consistent application and continuous improvement.

Key Components

- **Data Stewards**
 - Assign data stewards to oversee datasets, ensuring compliance with organizational policies and regulatory requirements.
 - Tools: Use Collibra for stewardship management.
- **AI Ethics and Governance Committees**
 - Establish cross-functional committees to oversee AI applications, including IT, legal, compliance, and business leaders.
 - Metrics: Define KPIs such as the percentage of reviewed LLM outputs flagged for compliance or ethical concerns.
- **Incident Response Accountability**
 - Assign specialized teams to address policy violations and data breaches involving LLMs.
 - Tools: Use ServiceNow or Splunk SOAR for incident management.

Real-World Example

- A financial institution formed an AI governance committee to monitor LLM compliance with Basel III requirements, leveraging audit trails generated by AI monitoring tools.

Best Practices

- Incorporate governance structures into enterprise risk management frameworks like ISO 31000 or NIST RMF.
 - Use collaborative platforms like Confluence or Slack to coordinate governance activities.
-

3. Continuous Monitoring and Governance Automation

Governance must adapt to changing risks, requiring real-time monitoring and automated enforcement mechanisms.

Key Components

- **Automated Monitoring**
 - Use AI-driven tools to monitor LLM interactions and detect violations in real-time.
 - Tools: Deploy Dynatrace AI Ops or Azure Monitor for anomaly detection and compliance tracking.
- **Audit Trails and Documentation**

- Record all data access, model training, and usage activities to maintain transparency and accountability.
 - Tools: Store logs in tamper-proof systems like Amazon S3 with Object Lock or Azure Data Lake.
 - **Dynamic Policy Updates**
 - Continuously align policies with emerging regulations using automation tools like TrustArc or OneTrust.
-

4. MLSecOps

Implement MLSecOps pipeline to encompass comprehensive data protection strategies that span the entire machine learning lifecycle, including data collection, preprocessing, training, and deployment. Implementing an MLSecOps pipeline requires a holistic approach that integrates security practices throughout the machine learning development lifecycle.

Key Components

- Secure Data Management Framework
 - Begin with a framework that includes data encryption, access controls, and privacy-preserving techniques. Implement continuous integration and continuous deployment (CI/CD) pipelines.
 - Tools: Use integrated security checks, such as automated vulnerability scanning, model validation, and anomaly detection.
- Infrastructure-as-code (IAC)
 - Use IaC with security-first principles.
 - Tools: Incorporate runtime monitoring tools that can detect model drift, performance degradation, and potential security breaches.
- Authentication and Authorization
 - Implement robust authentication and authorization mechanisms.
 - Tools: Use secure model packaging and deployment techniques, and establish comprehensive logging and auditing processes.
- Modeling and Testing
 - Regularly conduct threat modeling, penetration testing, and security assessments specific to machine learning systems.
- Awareness and Response
 - Develop incident response protocols tailored to ML-specific security challenges, and maintain a culture of security awareness through ongoing training and cross-functional collaboration between data scientists, security professionals, and DevOps teams.

Real-World Example:

- A technology company automated governance checks with Continuous Control Monitoring (CCM) to ensure compliance with ISO 27001 and reduce manual audits by 40%.

Best Practices:

- Use explainable AI tools (e.g. SHAP, LIME) to provide insights into model decisions and enhance trust in governance decisions.
 - Integrate governance automation with security information and event management (SIEM) platforms like Splunk.
-

5. Ensuring Ethical and Transparent AI Practices

Ethical considerations are integral to LLM governance, fostering trust and minimizing risks of harm from unintended consequences.

Key Components

- **Bias Detection and Mitigation**
 - Audit datasets and outputs for bias using frameworks like AI Fairness 360 or Fairlearn.
 - Tools: Combine unsupervised clustering with bias mitigation algorithms for large-scale datasets.
- **Explainability and Transparency**
 - Deploy frameworks such as Explainable Boosting Machines (EBM) to make model outputs interpretable.
 - Publish transparency reports detailing LLM use cases, datasets, and governance measures.
- **Stakeholder Engagement**
 - Involve external stakeholders (e.g., customers, regulators) through public workshops or consultations.
 - Tools: Use SurveyMonkey to gather stakeholder feedback on AI governance practices.

Real-World Example

- A global retailer used SHAP dashboards to demonstrate model fairness and explainability to regulators during compliance reviews.

Best Practices

- Establish grievance mechanisms for stakeholders to report concerns about LLM use.
 - Conduct ethical AI training sessions for employees, focusing on responsible use and decision-making.
-



6. Governance KPIs and Metrics

Measuring the effectiveness of governance frameworks is essential for continuous improvement and risk mitigation.

Key Metrics

- **Compliance Rates**
 - Percentage of LLM interactions audited and found compliant with policies.
- **Incident Response Times**
 - Average time to detect, investigate, and resolve policy violations or breaches.
- **Transparency Scores**
 - Frequency of publishing transparency reports and stakeholder feedback.

Tools

- Use Power BI or Tableau for visualizing governance performance metrics.
 - Automate metric collection with platforms like Elastic Stack.
-

Governance of LLM data security is a dynamic process requiring clear policies, robust oversight, and continuous adaptation. While integrating advanced tools, automation, and ethical principles, offer organizations great capabilities to protect sensitive data, ensure compliance, and build trust in their AI systems. A proactive governance framework is not just a safeguard—it is a strategic enabler that empowers organizations to innovate responsibly in the age of AI.



Conclusion & Call to Action

This white paper has provided a comprehensive examination of the data security challenges inherent in Large Language Model (LLM) deployments—spanning inadvertent data exposure, adversarial manipulation, model theft, and supply chain vulnerabilities. It underscores a multi-layered approach to security that includes robust encryption, privacy-preserving technologies (e.g. differential privacy, homomorphic encryption, secure multi-party computation), tightly controlled access, adversarial training, supply chain validation, and strict adherence to compliance frameworks like GDPR, CCPA, NIST AI RMF, and ISO.

The principles and best practices outlined here are not theoretical constructs; they are actionable, adaptable strategies designed to reinforce trust, maintain compliance, and uphold data integrity. The integration of detailed tool references and regulatory mappings serves as a practical guide for stakeholders—ranging from technical leaders and developers to CISOs and compliance officers—enabling them to proactively secure their LLM systems against an evolving threat landscape.

For additional resources, find Detailed Tool References [here](#) and Regulatory Mappings [here](#).

Key Takeaways

- **Layered security controls:** Employ strong data encryption, rigorous IAM policies, continuous monitoring, and privacy-enhancing techniques.
- **Robust governance:** Align with recognized frameworks, conduct regular audits, and treat security as a strategic priority rather than an afterthought.
- **Tailored approaches:** Recognize the diverse roles within organizations and provide role-specific guidance to ensure a unified security posture.
- **Future-forward mindset:** Anticipate emerging threats, stay informed about new adversarial tactics, and embrace quantum-resistant cryptographic practices as LLM capabilities advance.

Call for Collaboration

The complexity and dynamism of the LLM ecosystem demand a collective response. We invite:

- **Industry experts:** To share insights, methodologies, and tools that refine and strengthen these best practices.
- **Open-source contributors:** To engage in community-driven efforts to build secure frameworks, contribute code enhancements, and develop testing suites.
- **Organizations:** To adopt the recommended measures, provide feedback based on operational experiences, and influence future iterations of this guidance.

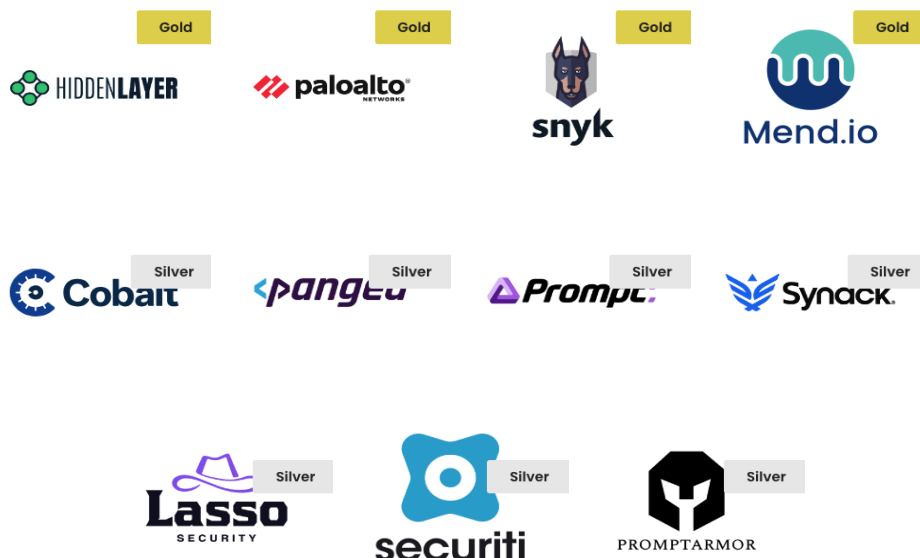
As we look ahead, the intersection of AI innovation and data protection will only grow more intricate. It is therefore incumbent upon us to remain vigilant, adaptable, and dedicated to safeguarding sensitive data. Through shared learning, collective effort, we can shape an AI landscape where large language models deliver transformative benefits without compromising privacy, integrity, or trust.

OWASP Top 10 for LLM Project Sponsors

We appreciate our Project Sponsors, funding contributions to help support the objectives of the project and help to cover operational and outreach costs augmenting the resources provided by the OWASP.org foundation. The OWASP Top 10 for LLM and Generative AI Project continues to maintain a vendor neutral and unbiased approach. Sponsors do not receive special governance considerations as part of their support. Sponsors do receive recognition for their contributions in our materials and web properties.

All materials the project generates are community developed, driven and released under open source and creative commons licenses. For more information on becoming a sponsor, [visit the Sponsorship Section on our Website](#) to learn more about helping to sustain the project through sponsorship.

Project Sponsors



Sponsor list, as of publication date. Find the full sponsor [list here](#).



Project Supporters


Project supporters lend their resources and expertise to support the goals of the project.

Accenture	Cobalt	Kainos	PromptArmor
AddValueMachine Inc	Cohere	KLAVAN	Pynt
Aeye Security Lab Inc.	Comcast	Klavan Security Group	Quiq
AI informatics GmbH	Complex Technologies	KPMG Germany FS	Red Hat
AI Village	Credal.ai	Kudelski Security	RHITE
aigos	Databook	Lakera	SAFE Security
Aon	DistributedApps.ai	Lasso Security	Salesforce
Aqua Security	DreadNode	Layerup	SAP
Astra Security	DSI	Legato	Securiti
AVID	EPAM	Linkfire	See-Docs & Thenavigo
AWARE7 GmbH	Exabeam	LLM Guard	ServiceTitan
AWS	EY Italy	LOGIC PLUS	SHI
BBVA	F5	MaibornWolff	Smiling Prophet
Bearer	FedEx	Mend.io	Snyk
BeDisruptive	Forescout	Microsoft	Sourcetoast
Bit79	GE HealthCare	Modus Create	Sprinklr
Blue Yonder	Giskard	Nexus	stackArmor
BroadBand Security, Inc.	GitHub	Nightfall AI	Tietoevry
BuddoBot	Google	Nordic Venture Family	Trellix
Bugcrowd	GuidePoint Security	Normalyze	Trustwave SpiderLabs
Cadea	HackerOne	NuBinary	U Washington
Check Point	HADESS	Palo Alto Networks	University of Illinois
Cisco	IBM	Palosade	VE3
Cloud Security Podcast	iFood	Praetorian	WhyLabs
Cloudflare	IriusRisk	Preamble	Yahoo
Cloudsec.ai	IronCore Labs	Precize	
Coalfire	IT University Copenhagen	Prompt Security	

Sponsor list, as of publication date. Find the full sponsor [list here](#).


References

1. Wikipedia. *Large language model*.
https://en.wikipedia.org/wiki/Large_language_model (Accessed: January 2, 2025).
2. Elastic. *What is a large language model?*
<https://www.elastic.co/what-is/large-language-models> (Accessed: January 2, 2025).
3. SAP. *What is a large language model?*
<https://www.sap.com/resources/what-is-large-language-model> (Accessed: January 2, 2025).
4. AWS. *Large language model*.
<https://aws.amazon.com/what-is/large-language-model/> (Accessed: January 2, 2025).
5. Databricks. *Glossary: Large language models (LLM)*.
<https://www.databricks.com/glossary/large-language-models-llm> (Accessed: January 2, 2025).
6. IBM. *Topics: Large language models*.
<https://www.ibm.com/topics/large-language-models> (Accessed: January 2, 2025).
7. Aqua Security. *LLM security; cloud-native vulnerability management for LLM security*.
<https://www.aquasec.com/cloud-native-academy/vulnerability-management/llm-security/>
(Accessed: January 2, 2025).
8. Sentra. *Safeguarding data integrity and privacy in the age of AI-powered LLMs*.
<https://www.sentra.io/blog/safeguarding-data-integrity-and-privacy-in-the-age-of-ai-powered-large-language-models-llms> (Accessed: January 2, 2025).
9. ScienceDirect. *Understanding LLM security risks*.
<https://www.sciencedirect.com/science/article/pii/S266729522400014X> (Accessed: January 2, 2025).
10. Krista AI. *How to protect your company data when using LLMs*.
<https://krista.ai/how-to-protect-your-company-data-when-using-llms/> (Accessed: January 2, 2025).
11. Forbes. *Will LLM adoption demand more stringent data security measures?*
<https://www.forbes.com/sites/hessiejones/2024/05/31/will-llm-adoption-demand-more-stringent-data-security-measures/> (Accessed: January 2, 2025).
12. Tigera. *LLM security: Guides*.
<https://www.tigera.io/learn/guides/llm-security/> (Accessed: January 2, 2025).
13. Master of Code. *LLM security threats*.
<https://masterofcode.com/blog/llm-security-threats> (Accessed: January 2, 2025).
14. OWASP. *OWASP Top 10 for Large Language Model Applications*.
<https://owasp.org/www-project-top-10-for-large-language-model-applications/> (Accessed: January 2, 2025).
15. Protecto. *LLM security risks and best practices*.
<https://www.protecto.ai/blog/llm-security-risks-best-practices> (Accessed: January 2, 2025).

- 
16. Lakera AI. *How to deploy an LLM*.
<https://www.lakera.ai/blog/how-to-deploy-an-llm> (Accessed: January 2, 2025).
 17. Metomic. *What are the top security risks of using large language models?*
<https://www.metomic.io/resource-centre/what-are-the-top-security-risks-of-using-large-language-models-llms> (Accessed: January 2, 2025).
 18. Nightfall AI. *Training data extraction attacks on LLMs*.
<https://www.nightfall.ai/ai-security-101/training-data-extraction-attacks> (Accessed: January 2, 2025).
 19. Semantic Scholar. *Extracting training data from LLMs*.
<https://www.semanticscholar.org/paper/Extracting-Training-Data-from-Large-Language-Models-Carlini-Tram%C3%A8r> (Accessed: January 2, 2025).
 20. Gaper. *The role of LLMs in data security*.
<https://gaper.io/role-of-llms-in-data-security/> (Accessed: January 2, 2025).
 21. LanguageWire. *LLM data security: Key considerations*.
<https://languagewire.com/en/blog/llm-data-security> (Accessed: January 2, 2025).
 22. Fuzzy Labs. *How someone can steal your large language model*.
<https://www.fuzzylabs.ai/blog-post/how-someone-can-steal-your-large-language-model>
(Accessed: January 2, 2025).
-

Additional References

23. National Institute of Standards and Technology (NIST). *Artificial Intelligence Risk Management Framework (AI RMF)*.
<https://csrc.nist.gov/projects/ai-risk-management> (Accessed: January 2, 2025).
24. European Union. *Proposal for a Regulation laying down harmonised rules on Artificial Intelligence (Artificial Intelligence Act)*.
<https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52021PC0206> (Accessed: January 2, 2025).
25. OpenAI. *OpenAI Security*.
<https://openai.com/security> (Accessed: January 2, 2025).
26. Google Cloud. *Securing AI/ML Workloads*.
<https://cloud.google.com/security/ai-ml> (Accessed: January 2, 2025).
27. Microsoft. *Security best practices for AI*.
<https://learn.microsoft.com/en-us/azure/security/fundamentals/security-best-practices-for-ai>
(Accessed: January 2, 2025).
28. Electronic Frontier Foundation (EFF). *AI and Privacy Insights*.
<https://www.eff.org/issues/ai> (Accessed: January 2, 2025).
29. MITRE. *MITRE ATLAS (Adversarial Threat Landscape for Artificial-Intelligence Systems)*.
<https://atlas.mitre.org/> (Accessed: January 2, 2025).

- 
30. International Organization for Standardization (ISO). *ISO/IEC 27001 Information Security Management Systems*.
<https://www.iso.org/isoiec-27001-information-security.html> (Accessed: January 2, 2025).
 31. ISACA. *COBIT 2019 Framework: Governance and Management Objectives*.
<https://store.isaca.org/s/communitylibrary?topic=COBIT&productCategory=COBIT2019> (Accessed: January 2, 2025).
 32. American Institute of CPAs (AICPA). *SOC 2® – SOC for Service Organizations: Trust Services Criteria*.
<https://us.aicpa.org/interestareas/frc/assuranceadvisoryservices/aicpasoc2report> (Accessed: January 2, 2025).
 33. Cloud Security Alliance (CSA). *Security Guidance for Critical Areas of Focus in Cloud Computing*.
<https://cloudsecurityalliance.org/research/security-guidance/> (Accessed: January 2, 2025).
 34. SANS Institute. *White Papers on Machine Learning Security*.
<https://www.sans.org/white-papers/?cat=machine-learning> (Accessed: January 2, 2025).
 35. Harvard Berkman Klein Center. *Research on AI Ethics and Governance*.
<https://cyber.harvard.edu/topics/ethics-governance-ai> (Accessed: January 2, 2025).
 36. Federal Trade Commission (FTC). *Guidance for AI and Data Privacy*.
<https://www.ftc.gov/business-guidance/resources> (Accessed: January 2, 2025).
 37. White House Office of Science and Technology Policy (OSTP). *Blueprint for an AI Bill of Rights*.
<https://www.whitehouse.gov/ostp/ai-bill-of-rights/> (Accessed: January 2, 2025).
 38. U.S. Department of Health & Human Services (HHS). *HIPAA for Professionals*.
<https://www.hhs.gov/hipaa/for-professionals/index.html> (Accessed: January 2, 2025).
 39. PCI Security Standards Council. *PCI DSS (Payment Card Industry Data Security Standard)*.
https://www.pcisecuritystandards.org/document_library (Accessed: January 2, 2025).
 40. OWASP Top 10 for LLM and Generative AI Security Project: <https://genai.owasp.org/>
 41. OWASP AI Security and Privacy Guide:
<https://owasp.org/www-project-ai-security-and-privacy-guide/#>
 42. OWASP AI Exchange: <https://owaspai.org/>
 43. OWASP Application Security Verification Standard (ASVS):
<https://owasp.org/www-project-application-security-verification-standard/>
 44. OWASP Testing Guide: <https://owasp.org/www-project-web-security-testing-guide/>
 45. OWASP Cheat Sheet Series: <https://cheatsheetseries.owasp.org/>
 46. OWASP Threat Modeling Project: <https://owasp.org/www-project-threat-dragon/>
 47. OWASP Dependency-Check: <https://owasp.org/www-project-dependency-check/>
 48. OWASP Software Assurance Maturity Model (SAMM): <https://owasp.org/www-project-samm/>
 49. OWASP DevSecOps Maturity Model (DSOMM):
<https://owasp.org/www-project-devsecops-maturity-model/>
 50. OWASP Code Review Guide: <https://owasp.org/www-project-code-review-guide/>
 51. OWASP Security Knowledge Framework (SKF): <https://www.securityknowledgeframework.org/>

