AI4DVault: A Registry Architecture for Securing the AI4D Supply Chain

Aminata SABANE
CITADEL, Université Joseph Ki-Zerbo
Ouagadougou, Burkina Faso
aminata.sabane@ujkz.bf

Tegawendé F. BISSYANDE
CITADEL, Université Joseph Ki-Zerbo
Ouagadougou, Burkina Faso
tegawende.bissyande@citadel.bf

***Abstract*—As artificial intelligence for development initiatives expand, ensuring secure and transparent supply chains for AI artifacts has become a critical challenge in emerging countries. Recent incidents of malicious models on repositories like Hugging Face demonstrate that machine learning model platforms are increasingly vulnerable to the same supply chain attacks that have plagued traditional software ecosystems. This paper presents the vision and architectural design for AI4DVault, a comprehensive registry architecture that addresses the unique security, provenance, and integrity requirements of AI4D ecosystems. While implementation remains a work in progress, our proposed architecture defines a cryptographically secure verification protocol that would allow stakeholders to trace model provenance from creation through deployment, protecting against namespace squatting, typosquatting, model confusion attacks, and unsafe serialization formats. The envisioned system would integrate with existing workflows through a standardized command-line interface while providing automated vulnerability scanning and integrity verification to prevent the distribution of compromised models. Our design adapts techniques from established supply chain security frameworks, including SLSA (Supply-chain Levels for Software Artifacts) and Sigstore, extending them to address AI-specific challenges such as dataset provenance tracking and model lineage verification. We present this architectural vision as a foundation for future implementation efforts, with the aim of establishing a roadmap toward more secure and trustworthy AI4D systems, particularly in resource-constrained environments where traditional security infrastructure may be limited.**

***Index Terms*—AI4D, AI artifacts, supply chain.**

I. Introduction

Artificial Intelligence for Development (AI4D) initiatives are transforming critical domains across the Global South, from healthcare delivery and agricultural optimization to financial inclusion and educational access [1]. As these AI systems become increasingly embedded in essential services, their security posture takes on heightened importance. However, the AI supply chain—encompassing data collection, model development, distribution, and deployment—presents unique security challenges that traditional software security approaches only partially address [2].

Recent security incidents have demonstrated that machine learning model repositories are becoming prime targets for supply chain attacks [3]. In March 2024, researchers identified malicious models on Hugging Face that, upon loading, executed arbitrary code giving attackers complete control of victim machines. One particularly concerning attack vector exploited the widely-used "*pickle*" serialization format to embed malware directly within model files. These attacks mirror the evolution of supply chain vulnerabilities that have successfully compromised traditional software ecosystems through package repositories like *npm* and *PyPI*, but with potentially greater impact given the complex, opaque nature of AI models and their widespread deployment in sensitive applications. The consequences of such attacks are particularly severe in development contexts where resource constraints often limit security infrastructure and expertise. When AI4D systems are compromised, the impacts extend beyond immediate security breaches to undermine trust in digital transformation efforts that are essential for sustainable development. Despite these high stakes, current AI model distribution mechanisms lack robust provenance tracking, verification protocols, and integrity guarantees that could mitigate these risks [4].

AI4DVault addresses this critical security gap by providing a comprehensive registry architecture specifically designed for securing the AI4D supply chain. Our approach builds upon established software supply chain security principles while extending them to address the unique characteristics of AI artifacts. Unlike traditional software where source code can be inspected and analyzed, AI models are effectively "black boxes" whose behaviors are determined by weights and architectures that cannot be easily verified through manual inspection. This opacity creates new attack surfaces that require novel security approaches. Drawing inspiration from industry frameworks such as Supply-chain Levels for Software Artifacts (SLSA) [5] and modern cryptographic signing solutions like Sigstore, AI4DVault implements end-to-end provenance tracking for AI artifacts. Our architecture encompasses cryptographic verification of model origin, tamper-proof documentation of dataset lineage, automated vulnerability scanning, and standardized interfaces for integration with existing AI development workflows. The key contributions of this paper include:

- A comprehensive threat model for AI supply chain vulnerabilities specific to development contexts
- An architecture for secure model and dataset registry with cryptographic verification
- A provenance protocol for tracking AI artifact lineage from creation through deployment
- An npm-style command-line interface for seamless integration with existing workflows

By establishing a secure foundation for AI artifact distribu-

tion and verification, AI4DVault enables the AI4D community to build more trustworthy systems while reducing security risk. Our work represents a significant step toward ensuring that artificial intelligence can be leveraged for development purposes without compromising on essential security requirements.

II. Background and Motivation

A. The Growing AI4D Ecosystem

Increasingly, AI technologies are being adapted for applications ranging from disease diagnosis in regions with limited healthcare infrastructure to optimizing agricultural practices in the face of climate change. This growing adoption is supported by an expanding ecosystem of open models, datasets, and tools specifically tailored for development contexts.

Unlike commercial AI deployments, AI4D initiatives often operate with significant constraints: limited computational resources, intermittent connectivity, and specialized requirements for localized data and models. These constraints have led to the development of lighter, more efficient models that can be deployed on edge devices and shared across multiple organizations working in similar domains. The result is a unique supply chain dynamic where AI artifacts are frequently transferred between organizations, customized for local needs, and deployed in sensitive applications with minimal security oversight.

B. Supply Chain Attacks in Machine Learning

Recent security research has revealed that machine learning model repositories are increasingly vulnerable to supply chain attacks. As reported by security researchers at Dropbox in their 2024 Black Hat Asia presentation "Confused Learning: Supply Chain Attacks through Machine Learning Models," ML repositories like Hugging Face provide attackers with opportunities similar to those that have been successfully exploited in traditional package repositories such as npm and PyPI.

These attack vectors include:

- **Namespace squatting attacks**: Researchers demonstrated how attackers could register namespaces that appear to belong to legitimate organizations. In one instance, they registered a namespace that appeared to be owned by a well-known company and found that authentic employees from the organization contacted them requesting to join the namespace to upload their models.
- **Typosquatting**: Similar to traditional software supply chain attacks, malicious actors can upload models with names that closely resemble popular models, exploiting typing errors to trick users into downloading compromised versions.
- **Model confusion attacks**: An attacker discovers the name of private dependencies within a project and creates public malicious models with identical names, causing internal projects to inadvertently use the compromised models.
- **Insecure serialization**: The JFrog security research team discovered a malicious ML model on Hugging Face that, when loaded, executed malicious code giving attackers full control of the victim's machine. This attack exploited the "pickle" file format, which is commonly used for serializing Python objects but can contain arbitrary code that executes when the file is loaded.

C. Current Security Limitations

Despite these growing threats, the current ecosystem for securing AI artifacts remains underdeveloped compared to traditional software security. Several critical limitations exist:

- **Inadequate model signing**: While some repositories offer cryptographic signing options (like GPG keys on Hugging Face), they are rarely used due to the complexity of key management and the friction they introduce to the model upload process.
- **Lack of provenance tracking**: Most AI repositories do not provide comprehensive tracking of an artifact's lineage, including which datasets were used in training and what transformations were applied.
- **Immature version control for datasets**: Unlike code version control systems, dataset management tools are less developed and often fail to track changes in a cryptographically verifiable manner.
- **Complex training processes**: AI training isn't fully scripted or deterministic like traditional software builds, making it harder to verify integrity throughout the development process.
- **Opacity of model contents**: Unlike source code that can be audited line by line, model weights and behaviors are not easily inspectable, making it difficult to detect malicious modifications.

These limitations create a substantial security gap in AI4D contexts, where organizations may lack the security expertise and infrastructure to implement their own protections. The consequences of compromised models could be severe, particularly when deployed in critical applications like healthcare diagnostics, financial services, or infrastructure management.

III. Threat Model

A. Attack Surface in AI4D Supply Chains

The AI4D supply chain contains multiple vulnerable touchpoints where security can be compromised. We identify the following key elements of the attack surface:

1) **Dataset Collection and Processing**: Initial data gathering, cleaning, and transformation steps where data could be poisoned or tampered with.
2) **Pre-trained Model Selection**: The process of selecting foundational models for transfer learning or fine-tuning.
3) **Training Infrastructure**: The compute environments, frameworks, and libraries used during model training.
4) **Model Storage and Distribution**: Repositories and hubs where models are published and from which they are downloaded.
5) **Deployment Environments**: The systems and platforms where models are ultimately installed and executed.

6) **Inference Pipelines**: The runtime components that prepare inputs for and process outputs from the model.

Each of these components presents distinct security challenges, made more complex by the distributed nature of AI4D ecosystems where resources and expertise are often shared across organizational boundaries.

B. Primary Threat Vectors

Based on our analysis of the attack surface and recent security incidents, we identify the following primary threat vectors in AI4D supply chains:

1) Data Poisoning Attacks: An adversary may manipulate training data to influence model behavior in subtle but harmful ways. This can be accomplished through **direct** poisoning, which involves injecting malicious examples into training datasets before they are used; **indirect** poisoning, where data sources (e.g., crowdsourcing platforms) that feed into training pipelines are compromised; and **targeted** poisoning, which subtly modifies data to create specific vulnerabilities or backdoors in resulting models. These attacks are particularly concerning in AI4D contexts where data may be gathered from diverse, less-secure sources and where data validation resources may be limited.

2) Model Tampering: Adversaries may modify models after training but before distribution. This can involve **weight manipulation**, which directly alters model parameters to introduce backdoors or vulnerabilities; **architecture injection**, which adds malicious components to model architectures that activate under specific conditions; and **checkpoint poisoning**, which modifies saved model checkpoints during long training processes. This form of tampering is especially difficult to detect since model weights don't have human-readable semantics that can be easily audited.

3) Malicious Model Publication: Attackers can publish entirely malicious models to repositories. This includes **trojan** models, which are designed to appear legitimate but contain **hidden** malicious functionality; **code execution** vectors, which exploit serialization formats to execute arbitrary code when loaded; and identity **spoofing**, where models are published under names that suggest authentic organizational origin. The current lack of robust verification in model repositories makes this attack vector particularly accessible to adversaries.

4) Supply Chain Compromise: Broader attacks targeting the infrastructure supporting AI development include **library compromission**, which involves injecting malicious code into popular ML libraries or frameworks; **development tool exploitation**, which compromises development environments or tools used in model creation; and **repository compromise**, which directly attacks model hub infrastructure to replace legitimate models.

5) Adversarial Runtime Manipulation: Threats that emerge when models are deployed include **input manipulation**, which involves crafting inputs that trigger hidden behaviors in compromised models; **configuration attacks**, which exploit configuration options to activate vulnerable model paths; and **model substitution**, which replaces deployed models with malicious versions during updates.

C. Threat Actors and Motivations

In the AI4D context, we consider several categories of potential adversaries. These include **criminal organizations** seeking financial gain through data theft, ransomware, or fraud enabled by compromised AI systems; **state-sponsored actors** targeting critical infrastructure or surveillance capabilities in developing regions; **competitors** seeking to undermine rival organizations by compromising their AI systems; **insiders**, who are individuals with legitimate access who may intentionally or unintentionally compromise security; and **opportunistic attackers** who exploit vulnerabilities without specific targeting. Each of these actors brings different capabilities, resources, and motivations, requiring a defense-in-depth approach to security.

D. Security Objectives for AI4DVault

Based on this threat model, we establish the following security objectives for AI4DVault. First, artifact integrity ensures that AI models and datasets remain unaltered from creation to deployment. Second, origin verification provides cryptographic guarantees about who created an artifact. Third, provenance tracking maintains tamper-proof records of an artifact's complete lineage. Fourth, access control limits artifact modification capabilities to authorized users. Fifth, vulnerability detection automatically identifies known security issues in artifacts. Sixth, usability delivers security features with minimal friction to encourage adoption. Finally, transparency enables auditing of security properties without compromising intellectual property. These objectives guide our architectural decisions and implementation priorities described in the following sections.

IV. AI4DVault Architecture

We first discuss the design principles that inform our architectural decisions, balancing security requirements with the practical constraints of AI4D environments (cf. Section IV-A). Then, we review the architecture (cf. Section IV-B) and describe the core components (cf. Section IV-C) before presenting the trust model (cf. Section IV-D).

A. Design Principles

AI4DVault architecture is guided by several key principles derived from our threat model and security objectives. We embrace *defense in depth*, implementing multiple layers of security to protect against the various attack vectors identified in our threat model. The architecture assumes **minimal trust**, acknowledging that individual components can be compromised and therefore requiring cryptographic verification at each stage. *Transparency* serves as another foundational principle, allowing verification of security properties without requiring trust in the registry itself. Throughout our design process, we prioritize *usability* to ensure that security features integrate smoothly into existing AI development workflows

without imposing a significant burden on developers. Finally, the architecture emphasizes *adaptability* to support resource-constrained environments common in AI4D contexts, where bandwidth, computing resources, and security expertise may be limited.

B. System Overview

AI4DVault implements a comprehensive registry architecture consisting of several integrated components that collectively secure the supply chain of AI artifacts. Figure 1 illustrates the high-level architecture and the interactions among components.

The architecture consists of three primary domains. First, the **Artifact Producers** domain includes components used by developers and organizations to create, sign, and publish AI artifacts. Second, the **Registry Core** domain provides centralized infrastructure for securely storing, verifying, and distributing artifacts. Third, the **Artifact Consumers** domain encompasses components used to verify, retrieve, and safely deploy AI artifacts in production environments. This three-part structure creates clear boundaries for security controls while maintaining the flexibility needed for various AI4D applications.

C. Core Components

1) Secure Storage Layer: The Secure Storage Layer provides tamper-evident storage for AI artifacts, including models, datasets, and their associated metadata. We implement content-addressable storage, where artifacts are identified by cryptographic hashes of their contents, ensuring that any modification results in a different identifier. This approach enables immutable versioning, preventing the overwriting of previous versions and maintaining a complete history of artifacts over time. To address the reliability concerns common in development contexts, the storage layer incorporates an optional replication mechanism that distributes critical artifacts across multiple storage nodes to prevent single points of failure. Understanding the storage constraints often present in AI4D environments, the system also implements deduplication to efficiently store artifacts with shared components, significantly reducing storage requirements for derivative models and related datasets.

2) Provenance Tracker: The Provenance Tracker maintains cryptographically verifiable records of each artifact's complete lineage throughout its lifecycle. For each artifact, it documents the data sources, recording the origin and transformations applied to the training data. It captures details about the build environment, documenting the training infrastructure, frameworks, and dependencies used during model creation. The tracker implements process attestation to capture cryptographic evidence of the training process itself. Additionally, it maintains a transformation history that tracks all modifications applied to artifacts throughout their lifecycle. This comprehensive approach extends the provenance format SLSA[1] [5] with AI-specific fields to accommodate the unique attributes and requirements of machine learning artifacts, creating a verifiable chain of custody from initial data collection through model deployment.

3) Signing Service: The Signing Service enables cryptographic verification of artifact identity and integrity, implementing approaches inspired by modern cryptographic signing solutions. The service uses ephemeral key generation, similar to Sigstore [6], which minimizes the key management burden that has hampered the adoption of previous signing approaches such as GPG [7]. To support diverse organizational structures, the service integrates with existing identity providers relevant to the AI4D ecosystem. All signing operations are recorded in transparent logs, maintained as an append-only ledger that enables public verification of signing events. Recognizing the connectivity challenges in many development contexts, the system supports offline verification in environments with limited internet access, allowing security verification even in remote or disconnected settings.

4) Verification Engine: The Verification Engine validates artifacts against security policies and provenance claims, serving as the cornerstone of AI4DVault's trust architecture. The engine performs comprehensive signature verification, confirming cryptographic signatures on both artifacts and their associated provenance documents. It conducts thorough provenance validation to verify that claimed provenance matches the cryptographic evidence provided. For organizational deployments, the engine enforces policy evaluation, assessing artifacts against customizable security policies specific to organizational or community requirements. Throughout this process, the engine validates the complete chain of custody from data collection to deployment, ensuring no gaps exist where tampering could occur undetected.

5) Vulnerability Scanner: The Vulnerability Scanner automatically detects known security issues in artifacts before they enter production environments. The scanner identifies models with configurations known to be vulnerable to published attacks, reducing the risk of exploitation. It flags models using potentially unsafe serialization formats, such as pickle [8], which has been exploited in recent supply chain attacks. Format validation ensures that models adhere to secure format specifications, promoting the use of safer alternatives such as SafeTensors. The scanner also analyzes model dependencies for security vulnerabilities, identifying risks in the underlying libraries and frameworks. This multifaceted approach to vulnerability detection provides developers with actionable means to secure their models before deployment.

6) Access Control System: The Access Control System manages the permissions for artifact operations, balancing security with collaboration needs. The system implements role-based access control, assigning permissions based on user roles within organizations. These permissions are fine-grained and control `read`, `write`, and `publish` operations at the individual artifact level. For particularly sensitive operations, the system supports approval workflows that require multiparty authorization before changes are applied. All access attempts

[1]Supply Chain Levels for Software Artifacts

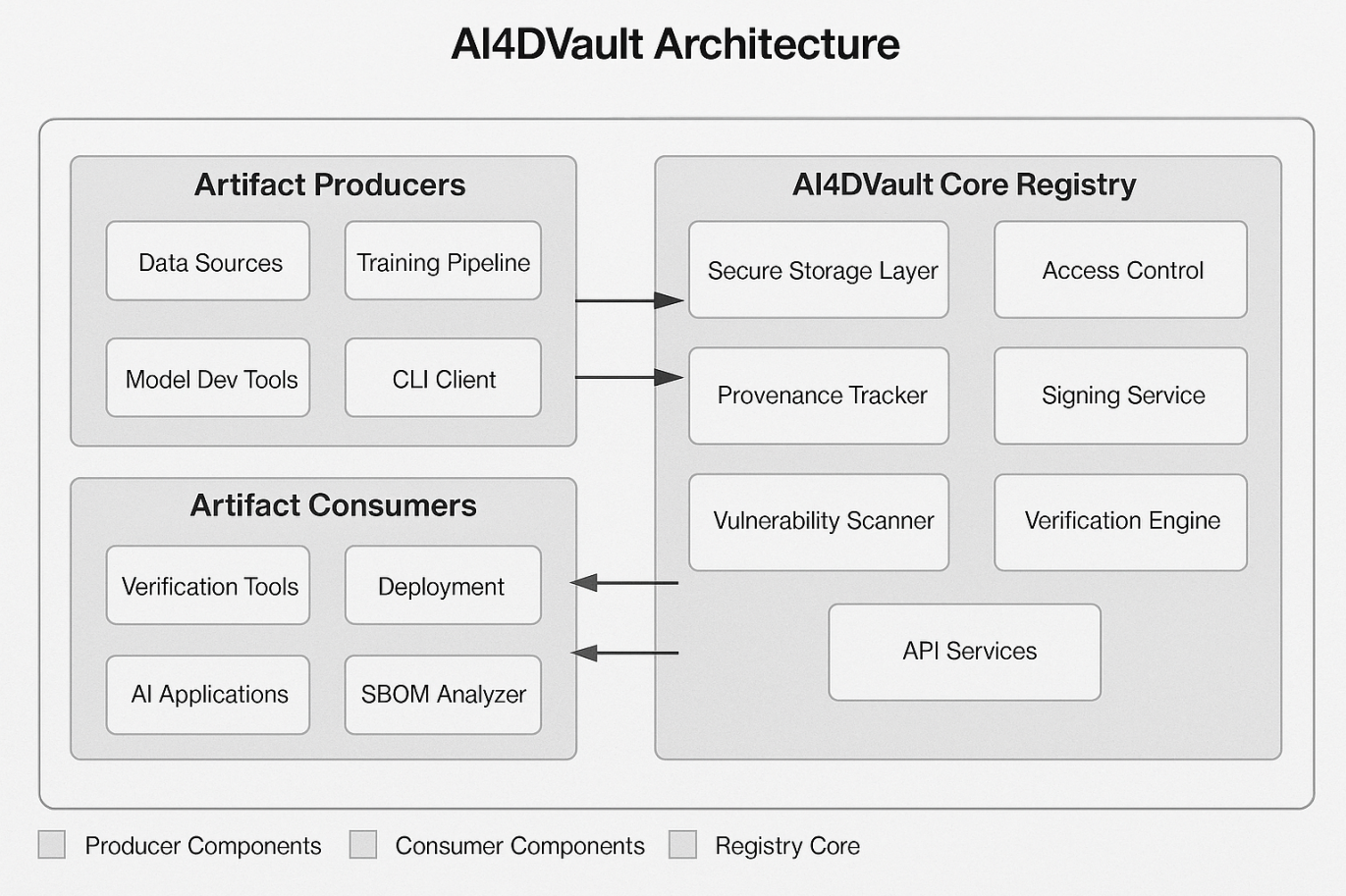

Fig. 1. AI4DVault Architecture Overview showing core components and their interactions

and administrative actions are recorded in comprehensive audit logs, enabling security teams to review system usage and investigate potential security incidents. This approach enables secure collaboration while maintaining the integrity of the artifact repository.

7) API Services: The API Services provide programmatic interfaces for interacting with the registry, enabling integration with diverse development workflows. A RESTful API enables straightforward integration with various tools and platforms in the AI ecosystem. For more complex data requirements, a GraphQL interface supports sophisticated queries of artifact metadata and relationships. Webhook notifications alert connected systems to relevant events, enabling automation of downstream processes. A comprehensive command-line interface provides developers with familiar, script-friendly access to registry functions. Together, these interfaces ensure that security features can be seamlessly incorporated into existing development workflows without disrupting productivity.

D. Trust Model

AI4DVault employs a multilayered trust model to accommodate varying security requirements across different deployment contexts. At the basic level, **trust based on the registry** allows the registry itself to enforce security policies and maintain the integrity of artifacts. For greater assurance, **verifiable credentials** provide cryptographic proof of the artifact properties independent of the trust of the registry. In collaborative environments, **community attestation** enables multiple independent parties to verify the properties and provenance of the artifacts. For the highest security requirements, a **zero-trust deployment** model allows consumers to verify all security properties without trusting the registry infrastructure itself. This flexible approach allows AI4DVault to operate effectively in environments with varying security infrastructure and trust requirements, from resource-constrained deployments to security-critical applications.

E. Integration with Current AI4D Workflows

AI4DVault integrates with existing AI4D workflows through multiple deployment models that address varying resource constraints and connectivity limitations.

Offline Deployment Model: For environments with limited connectivity, AI4DVault implements a "*sync-when-available*" architecture. Critical artifacts and verification data are cached locally, with periodic synchronization when connectivity permits. The system maintains cryptographic integrity through local verification using pre-distributed trust anchors.

Mobile-First Architecture: Recognizing that mobile devices often provide primary internet access in low-resource settings, AI4DVault implements a mobile-optimized interface with progressive web app capabilities. The system supports QR-code based artifact sharing and verification, enabling secure model distribution without requiring high-bandwidth connectivity.

Resource Optimization Strategies:

- *Compressed Verification*: Proof compression reduces verification data by 90% while maintaining security guarantees [9]
- *Incremental Synchronization*: Delta-sync protocols minimize bandwidth usage for model updates
- *Shared Infrastructure*: Community-operated nodes reduce individual organizational infrastructure burden
- *Adaptive Security*: Security levels adjust based on available computational resources and connectivity

V. Implementation Details

A. Command-line Interface

AI4DVault provides a comprehensive command-line interface (CLI) modeled after package managers like `npm` to ensure familiarity and ease of adoption for developers. The interface design prioritizes intuitive commands that naturally integrate with existing AI development workflows. By following established patterns from widely used package managers, we reduce the learning curve and cognitive overhead for security features. This approach makes security a seamless part of the development process rather than an additional burden that might escape.

The CLI supports the complete artifact lifecycle, from initialization and publishing to verification and deployment. Each command provides clear feedback and actionable error messages to guide developers toward secure practices. The interface also includes comprehensive local documentation and contextual help, ensuring that even users without constant internet connectivity can use the system effectively. This usability-focused design is critical for adoption in AI4D contexts, where competing priorities often result in security features being bypassed if they create friction in the development process.

```
# Initialize a new AI project with supply chain tracking
ai4dvault init

# Publish a model with provenance information
ai4dvault publish model --name "medical-classifier" --
    version "1"

# Verify a model's integrity and provenance
ai4dvault verify model://organization/medical-classifier@1

# Generate Software Bill of Materials (SBOM)
ai4dvault generate-sbom --output sbom.json

# Audit model for vulnerabilities
ai4dvault audit model://organization/medical-classifier@1

# Pull verified model for deployment
ai4dvault pull model://organization/medical-classifier@1
```

Listing 1. Example CLI Commands

B. Data Provenance Tracking

AI4DVault implements comprehensive tracking of the provenance of the data set using a structured metadata format that captures the complete lineage of data artifacts. Our approach records detailed information on the origin, transformation, and verification of the data, allowing a thorough assessment of the suitability and security of the data set. The provenance system maintains cryptographically verifiable records of who created the dataset, what source data it was derived from, and what processing steps were applied. This information is critical for evaluating whether a dataset is appropriate for use in sensitive applications and for tracing potential issues back to their source.

```
{
  "datasetId": "sha256:71f396d9...",
  "name": "malaria-cell-images",
  "version": "2.1.0",
  "creator": {
    "id": "did:key:z6Mkfrr7...",
    "name": "Nanoro Malaria Research Center"
  },
  "sources": [
    {
      "id": "sha256:8a74d209...",
      "name": "original-cell-images",
      "uri": "dataset://nanoro/malaria/cell-images@1.0.0"
    }
  ],
  "transformations": [
    {
      "type": "preprocessing",
      "description": "Image normalization and
      ↪ augmentation",
      "tool": "OpenCV 4.5.3",
      "parameters": {
        "normalization": "min-max",
        "augmentations": ["rotation", "flip"]
      },
      "hash": "sha256:b2d8a2e9..."
    }
  ],
  "signatures": [
    {
      "keyId": "did:key:z6Mkfrr7...",
      "signature": "MEUCIQDKVt..."
    }
  ]
}
```

Listing 2. Example Dataset Provenance

The provenance format is designed to be both human-readable for manual review and machine-parsable for automated verification. Each dataset is assigned a unique identifier based on its cryptographic hash, enabling precise tracking throughout the AI development lifecycle. Transformations applied to the data are documented with their parameters and tools, creating a reproducible record of data processing. The entire provenance record is cryptographically signed to prevent tampering, ensuring that the history of data cannot be falsified. This structured approach enables detailed tracking of data lineage while supporting efficient verification of dataset integrity and authenticity throughout the AI development process.

C. Model Provenance Implementation

For AI models, AI4DVault extends the SLSA provenance framework with ML-specific attributes that capture the unique aspects of model training and evaluation. Our implementation

builds on established software supply chain security practices while addressing the specific requirements of machine learning artifacts. This approach maintains compatibility with existing supply chain security tools while adding the additional context needed to secure AI systems.

The model provenance format captures critical information about the model's origins and training process. It records the model architecture, hyperparameters, and training environment, enabling assessment of the model's technical foundations. Training and validation datasets are referenced by their cryptographic identifiers, establishing verifiable links to the data provenance records. Evaluation metrics provide context for understanding model performance and suitability for specific applications. The complete training environment is documented, including framework versions and hardware configurations, creating a reproducible record of the model's creation. This comprehensive provenance record enables thorough verification of a model's origins and characteristics, enhancing security throughout the AI supply chain.

D. Artifact Verification Protocol

AI4DVault implements a multi-stage verification protocol that ensures artifacts meet security requirements before they are deployed in sensitive applications. The verification process begins with basic integrity verification, validating that artifact contents match their claimed cryptographic hash to detect any tampering during storage or transfer. The system then performs signature verification, checking that signatures were created by the claimed identity using the Sigstore-based transparent log. This step confirms the artifact's authentic origin without requiring complex key management by consumers.

Once basic integrity is established, the system conducts provenance validation to verify that the artifact's provenance record is complete and properly signed. This step ensures the artifact's full history is transparent and verifiable. Format validation follows, ensuring the artifact uses secure serialization formats and follows best practices to prevent code execution vulnerabilities like those seen in recent pickle-based attacks. Finally, policy evaluation confirms the artifact meets organizational or community security policies, allowing enforcement of domain-specific security requirements.

This layered approach ensures comprehensive verification while allowing organizations to enforce their specific security requirements. The verification results are presented to users in clear, actionable formats that highlight any security concerns and provide guidance for remediation. For automated workflows, the verification status is returned in machine-readable formats that can be integrated into CI/CD pipelines and deployment systems. This comprehensive verification protocol enables organizations to make informed decisions about which artifacts to trust and deploy.

E. Model Fingerprinting and Verification Protocol

Model fingerprinting extends beyond simple weight hashing to capture behavioral characteristics. Our approach implements three complementary fingerprinting methods:

1) **Structural Fingerprinting**: Cryptographic hashing of model architecture, layer configurations, and parameter distributions
2) **Behavioral Fingerprinting**: Statistical analysis of model outputs on standardized test inputs, creating reproducible behavioral signatures [10]
3) **Training Fingerprinting**: Cryptographic attestation of training process parameters, including optimizer states and convergence patterns

```
# Example model fingerprint generation
def generate_model_fingerprint(model,
    training_config):
    structural_hash = hash_model_structure(model)
    behavioral_sig = generate_behavioral_signature(
    model, standard_inputs)
    training_hash = hash_training_config(
    training_config)

    return {
        "composite_fingerprint": combine_hashes(
    structural_hash,

    behavioral_sig,

    training_hash),
        "verification_proofs": generate_zk_proofs(
    model, training_config),
        "transparency_log_entry": submit_to_rekor(
    fingerprint_data)
    }
```

Listing 3. Model Fingerprint Generation

The verification protocol implements a multi-stage approach: (1) Basic integrity verification through hash comparison, (2) Behavioral verification using statistical tests on model outputs, (3) Provenance verification through cryptographic proof validation, and (4) Policy compliance checking against organizational security requirements.

F. Cryptographic Mechanisms

AI4DVault employs a layered cryptographic architecture building on established frameworks:

Signing Infrastructure: We leverage Sigstore's keyless signing approach, eliminating long-term key management burden while providing cryptographic guarantees. All artifacts are signed using ECDSA with ephemeral keys, with signatures stored in Rekor's append-only transparency log.

Zero-Knowledge Proofs: For high-security deployments, we implement ZKML verification using the EZKL framework [11], enabling verification of model properties without revealing model weights. Current implementation supports models up to 18M parameters with proof generation time under 10 minutes for typical transformer architectures.

Content-Addressable Storage: Artifact integrity leverages IPFS's Merkle DAG structure [12], providing cryptographic verification through hash chains. The system implements deduplication reducing storage requirements by up to 60% for derivative models sharing common components.

Threshold Cryptography: For community governance, we implement threshold signature schemes requiring k-of-n signa-

tures for critical operations, enabling distributed trust without single points of failure [13].

G. Dataset Provenance Capture Implementation

AI4DVault will implement comprehensive dataset provenance using a novel combination of cryptographic hashing and metadata tracking. Each dataset undergoes multi-level fingerprinting:

```
{
  "dataset_fingerprint": {
    "content_hash": "sha256:a1b2c3d4...",
    "sample_hashes": ["sha256:e5f6g7h8...",
    ↪ "sha256:i9j0k1l2..."],
    "statistical_fingerprint": {
      "mean": 0.342, "std": 1.254,
    ↪ "skewness": -0.123,
      "feature_correlations": {"f1_f2": 0
    ↪ .789, "f1_f3": -0.234}
    },
    "transformation_chain": [
      {"operation": "normalization",
    ↪ "params": {"method": "min-max"}},
      {"operation": "augmentation", "params":
    ↪ {"rotation": 15, "flip": true}}
    ]
  }
}
```

Listing 4. Envisioned Dataset Fingerprint Structure

The system captures provenance at three levels: (1) File-level integrity using content-addressable storage, (2) Sample-level verification through statistical sampling of individual data points, and (3) Semantic-level tracking of transformations and their parameters.

For sensitive datasets, we implement differential privacy mechanisms during provenance collection, adding calibrated noise to statistical fingerprints while preserving verification capabilities [14]. This approach enables provenance tracking without compromising privacy-sensitive information.

H. Security Scanning Mechanism

The vulnerability scanning system implements multiple detection strategies to identify potential security issues before they reach production. Format-based detection identifies known unsafe serialization formats such as pickle, which have been exploited in recent supply chain attacks. This detection is particularly important in the AI4D context, where awareness of these vulnerabilities may be limited. The scanner performs static analysis of model structures, examining for suspicious components or architectural patterns that could indicate security issues.

For higher-risk scenarios, optional dynamic analysis can be performed in a sandboxed execution environment to detect runtime behavior that might indicate malicious code. The scanner performs thorough metadata validation to verify that artifact metadata is complete and consistent, detecting potential tampering or misrepresentation. Dependency analysis checks libraries and frameworks against vulnerability databases to identify known security issues in the underlying components.

Scanning results are categorized by severity and linked to specific remediation guidance, helping developers address identified issues efficiently. The scanning system is designed to be extensible, allowing new detection methods to be added as new threats emerge. This forward-looking approach ensures that the security scanning capability can evolve alongside the threat landscape, providing ongoing protection for AI4D artifacts.

I. Security Analysis and Threat Mitigation

We provide a formal analysis of AI4DVault's security properties against the threat model established in Section IV-D.

Threat T1: Data Poisoning Attacks AI4DVault addresses data poisoning through dataset provenance tracking with cryptographic attestation of sources and transformations. Any modification to training data results in detectable changes to provenance fingerprints, though the system cannot detect adversarial examples crafted to appear legitimate.

Threat T2: Model Tampering The system implements content-addressable storage with cryptographic hashing and behavioral fingerprinting. Any modification to model weights changes both the artifact identifier and behavioral signature, making tampering detectable. However, sophisticated attacks might preserve behavioral signatures on test inputs.

Threat T3: Malicious Model Publication Community attestation, vulnerability scanning, and reputation systems provide multi-layered protection. Models require multiple independent verifications before reaching production status, though social engineering attacks against community verifiers remain possible.

Threat T4: Supply Chain Compromise End-to-end cryptographic verification and transparent audit logs ensure that compromise of individual components does not compromise artifact integrity. Coordinated attacks against multiple infrastructure components could potentially succeed.

Security Assumptions The analysis assumes cryptographic primitives (SHA-256, ECDSA, zkSNARKs) remain secure, at least one honest participant exists in community verification, hardware security modules for critical signing operations are not compromised, and network communication uses authenticated channels (TLS 1.3+).

J. Deployment Considerations

AI4DVault supports multiple deployment models to accommodate varying resource constraints and security requirements across diverse AI4D contexts. A centralized deployment model provides a single registry instance serving an entire community or organization, offering efficiency and simplified management. For larger or more distributed environments, a federated deployment enables multiple registry instances that share artifact metadata while maintaining local storage, reducing bandwidth requirements while preserving security guarantees.

To support environments with limited connectivity, the system includes an offline mode that enables periodic synchronization of critical artifacts and metadata. This capability

is essential for AI4D contexts where internet connectivity may be intermittent or expensive. For the most resource-constrained environments, a minimal deployment configuration optimizes the system for low-resource operation while maintaining core security features.

These flexible deployment options allow AI4DVault to be effectively implemented across the diverse environments common in AI4D contexts. The deployment architecture is designed to scale from small, single-team implementations to large, multi-organizational registries, all while maintaining the security guarantees needed to protect AI supply chains. This adaptability is critical for the diverse conditions encountered in development contexts, where infrastructure quality and availability can vary widely.

VI. Comparison with Existing Secure Model Registries

Current secure model registry solutions demonstrate varying security maturity and architectural approaches, highlighting the need for AI4DVault's comprehensive security model.

Enterprise Cloud Platforms: AWS SageMaker Model Registry provides comprehensive cross-account model sharing with mandatory KMS encryption and sophisticated approval workflows [15]. However, the platform lacks provenance transparency and requires vendor lock-in. Azure ML Registry excels in enterprise identity integration [16] but provides limited community verification capabilities. Google Model Garden emphasizes organizational policy controls [17] but lacks fine-grained provenance tracking.

Community Platforms: Hugging Face Hub implements comprehensive malware scanning (ClamAV, Picklescan) and secret detection (TruffleHog) [18], but the platform's 2024 security breach involving compromised Spaces secrets [19] demonstrates ongoing vulnerabilities in community-driven platforms. The platform lacks cryptographic signing integration and comprehensive provenance tracking.

Open-Source Solutions: Projects like Harbor and Red Hat Quay provide strong container registry security [20] but lack AI-specific features like model verification and provenance tracking. The Sigstore model-transparency project [21] represents the closest existing solution, providing cryptographic signing for ML models but lacking comprehensive registry capabilities and AI4D-specific features.

AI4DVault's Novel Contributions:

1) **AI4D-Specific Design**: Unlike enterprise platforms designed for well-resourced environments, AI4DVault addresses connectivity constraints, limited computational resources, and diverse organizational structures common in development contexts.
2) **Community-Driven Security**: While existing platforms rely on centralized trust, AI4DVault implements community attestation and reputation systems enabling distributed verification.
3) **Comprehensive Provenance**: Extending beyond basic model tracking, AI4DVault provides end-to-end provenance from data collection through deployment, addressing AI-specific supply chain vulnerabilities.
4) **Offline Capabilities**: Unlike cloud-native solutions requiring constant connectivity, AI4DVault supports offline verification and periodic synchronization for resource-constrained environments.

VII. Future Work

AI4DVault represents a comprehensive architectural vision for securing AI supply chains in development contexts. It is important to emphasize that while the architecture and design principles presented in this paper are well-defined, a complete implementation of AI4DVault remains a work in progress. Our current prototype demonstrates core functionality, but significant development efforts are still needed to realize the full vision. This section outlines key areas for future work as we move from architectural design toward a production-ready system.

A. Implementation Roadmap

Our immediate priority is to complete the implementation of the core components described in this paper. The initial focus will be on establishing the fundamental infrastructure for secure artifact storage, basic provenance tracking, and cryptographic verification. This foundation will enable early adopters to begin integrating security practices into their workflows while more advanced features are developed. We have published an open-source roadmap detailing implementation milestones and inviting community contributions to accelerate development.

Once the core functionality is stable, we will proceed with implementing the more sophisticated features such as advanced provenance verification, policy enforcement, and federated deployments. Throughout this process, we will maintain a strong emphasis on security, usability, and performance testing to ensure the system meets the needs of diverse AI4D environments. Regular security audits and community feedback will guide our development priorities to address the most pressing needs first.

1) Concrete Implementation Steps: We have developed a phased implementation approach for AI4DVault based on established best practices from similar secure registry projects. The implementation follows a three-phase roadmap designed to provide immediate value while building toward the full architectural vision.

Phase 1 (Months 1-8): Core Infrastructure

- Implement content-addressable storage using IPFS-based architecture, building on proven implementations that achieve significant latency reduction for small files [22]
- Deploy basic cryptographic signing using Sigstore's keyless infrastructure, leveraging the existing model-transparency project [21]
- Establish fundamental CLI interface modeled after npm/-cargo package managers
- Implement basic artifact verification protocols for integrity checking

Phase 2 (Months 9-18): Enhanced Security Features

- Deploy comprehensive provenance tracking extending SLSA framework with ML-specific attributes [23]
- Implement vulnerability scanning using ClamAV and specialized pickle file detection [24]
- Establish federated deployment capabilities for multi-organizational use
- Integrate with existing ML workflows (MLflow, DVC, Kubeflow) [25]

Phase 3 (Months 19-24): Advanced Capabilities

- Deploy zero-knowledge proof verification for models up to 18M parameters using EZKL framework [26]
- Implement policy enforcement and governance frameworks
- Establish community attestation and reputation systems
- Deploy production-grade monitoring and incident response capabilities

2) Timeline and Milestones: Current development status includes a functional prototype demonstrating core storage and basic verification capabilities. Critical milestones include:

- Q3 2025: Alpha release with basic signing and verification
- Q1 2026: Beta release with full provenance tracking
- Q3 2026: Production release with advanced security features

3) Potential Collaborators: We are actively pursuing collaboration with several key stakeholders:

- Technical partnerships with Sigstore project maintainers and SLSA working group
- Deployment partnerships with AI4D organizations including AI4D Africa (https://www.ai4d.ai/) and IDRC (https://idrc-crdi.ca/)

B. Enhanced Provenance Verification

As implementation progresses, we plan to explore more sophisticated provenance verification techniques to address complex verification scenarios. We will develop methods for verifiable inference tracking, enabling organizations to trace not just how models were trained but how they are being used in production. This capability will be particularly valuable for sensitive applications in healthcare and financial inclusion where documenting model usage is critical for regulatory compliance.

We also aim to improve dataset provenance verification by developing techniques to cryptographically verify the presence or absence of specific data points within training datasets. This capability could help address copyright and licensing concerns by providing verifiable evidence that certain protected content was not used during model training. While challenging from both technical and computational perspectives, such verification would significantly strengthen the legal and ethical foundations of AI4D initiatives.

C. Federated Registry Networks

Building on our current federation capabilities, we envision developing a more comprehensive network protocol for AI4DVault instances to collaborate across organizational boundaries. This approach would enable regional registry networks that share security information and artifact metadata while respecting data sovereignty requirements. By establishing trust relationships between registries, we can enable secure cross-organization collaboration without requiring centralized infrastructure or governance.

The federation protocol will support various trust models, from hierarchical arrangements suitable for governmental or regulatory contexts to peer-to-peer relationships appropriate for collaborative research partnerships. This flexibility will allow AI4DVault to adapt to the diverse organizational structures present in the AI4D ecosystem, from grassroots community initiatives to large international development programs.

D. Resource-Optimized Security

A persistent challenge in developing regions is the limited availability of computational resources and connectivity. Future work will focus on developing even more resource-efficient security mechanisms that provide strong guarantees with minimal overhead. We plan to explore progressive verification approaches that adapt to available resources, providing basic security guarantees even in severely constrained environments while enabling more comprehensive verification when resources permit.

Research into compressed provenance representations could significantly reduce the storage and bandwidth requirements for security metadata. Similarly, developing verification techniques optimized for low-power devices would extend secure AI practices to the edge computing environments increasingly important in development contexts. These optimizations will be crucial for ensuring that security does not become a privilege limited to well-resourced organizations.

E. Integration with AI Governance Frameworks

As regulatory frameworks for AI continue to develop globally, AI4DVault will need to evolve to support compliance with emerging governance requirements. Future work will focus on aligning our provenance and verification mechanisms with international standards and regulatory frameworks specific to AI systems. We plan to develop compliance reporting modules that automatically generate documentation required by various regulatory regimes based on the provenance data already captured by the system.

Additionally, we aim to integrate with broader AI governance tools, including fairness assessment frameworks, privacy impact analysis systems, and human rights due diligence processes. This integration will allow organizations to leverage the security infrastructure provided by AI4DVault to address a wider range of governance concerns, creating a more comprehensive approach to responsible AI development.

F. Community-Driven Security

The open nature of many AI4D initiatives creates opportunities for community-driven security approaches that complement the technical mechanisms provided by AI4DVault. We plan to develop collaborative security features that enable community vetting and attestation of artifacts, allowing organizations to leverage collective expertise and oversight. This approach could include reputation systems for artifact publishers, community vulnerability reporting mechanisms, and collaborative security policy development.

Research into effective incentive structures for community security contributions could significantly enhance the sustainability of these approaches. By recognizing and rewarding security-enhancing behaviors, we can foster a community that actively contributes to securing the shared AI commons upon which many development initiatives depend. This community-centered approach acknowledges that technical solutions alone cannot address all security challenges, particularly in contexts where technical expertise may be limited.

VIII. Conclusion

The increasing adoption of artificial intelligence in development contexts brings both tremendous opportunities and significant security challenges. As AI systems become embedded in critical infrastructure and essential services across the Global South, securing their supply chains becomes an ethical imperative. AI4DVault addresses these challenges by providing a comprehensive architecture for securing AI artifacts throughout their lifecycle.

Our approach adapts established software supply chain security principles to the unique requirements of AI systems, extending concepts like provenance tracking and cryptographic verification to encompass datasets, models, and training processes. By building on proven techniques while addressing AI-specific concerns, we offer a practical path toward more secure AI development in resource-constrained environments.

The architecture balances security requirements with the practical realities of AI4D contexts, where limited resources, intermittent connectivity, and diverse organizational structures necessitate flexible, adaptable solutions. By supporting multiple deployment models and integrating seamlessly into existing workflows, AI4DVault makes security accessible rather than burdensome.

Beyond securing the AI supply chain, the provenance tracking mechanisms we've developed can address broader concerns including copyright compliance, ethical data usage, and regulatory adherence. This expandability ensures that investments in security infrastructure will continue to yield benefits as the governance landscape evolves.

As we move from architectural design to implementation, we invite collaboration from researchers, developers, and organizations across the AI4D ecosystem. Only through collective effort can we establish the secure foundations needed for responsible AI innovation in development contexts. AI4DVault represents a starting point for building increasingly sophisticated approaches to securing our shared AI future and ensuring that the benefits of AI can be safely realized in diverse development settings.

References

[1] S. Mann and M. Hilbert, "Ai4d: artificial intelligence for development," *SSRN*, 2018.

[2] I. Hepworth, K. Olive, K. Dasgupta, M. Le, M. Lodato, M. Maruseac, S. Meiklejohn, S. Chaudhuri, and T. Minkus, "Securing the ai software supply chain," tech. rep., Technical report, Google, 2024. URL: https://research. googl e/pubs/securing ..., 2024.

[3] A. Wood and M. Walker, "Confused learning: Supply chain attacks through machine learning models," in *Black Hat Asia*, Black Hat, 2024.

[4] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg, "Badnets: Evaluating backdooring attacks on deep neural networks," *IEEE Access*, vol. 7, pp. 47230–47244, 2019.

[5] SLSA Working Group, "SLSA: Supply chain levels for software artifacts." https://slsa.dev, 2022. Accessed: 2024-03-01.

[6] Z. Newman, J. S. Meyers, and S. Torres-Arias, "Sigstore: Software signing for everybody," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2353–2367, 2022.

[7] M. Lucas, *PGP & GPG: Email for the practical paranoid*. No Starch Press, 2006.

[8] M. Pilgrim, "Serializing python objects," in *Dive Into Python 3*, pp. 205–223, Springer, 2009.

[9] X. Zhao, S. Gunn, M. Christ, J. Fairoze, A. Fabrega, N. Carlini, S. Garg, S. Hong, M. Nasr, F. Tramer, S. Jha, L. Li, Y.-X. Wang, and D. Song, "Sok: Watermarking for ai-generated content," 2025.

[10] J. Fairoze, S. Garg, M. Wang, and S. Jha, "Publicly-detectable watermarking for language models," *arXiv preprint arXiv:2310.18491*, 2024.

[11] EZKL Team, "Ezkl: Zero-knowledge machine learning." url=https://ezkl.xyz/, 2024.

[12] Filebase, "Leveraging ipfs for reliable and efficient ai applications." Technical Blog, 2024.

[13] W. Chen *et al.*, "Mpcfl: Towards multi-party computation for secure federated learning aggregation." ResearchGate, 2024.

[14] S. Schelter *et al.*, "Supporting better insights of data science pipelines with fine-grained provenance," 2024.

[15] Amazon Web Services, "Sagemaker model registry now supports model lineage to improve model governance." AWS News, 2024.

[16] Microsoft, "Manage models registry with mlflow - azure machine learning." Microsoft Learn, 2024.

[17] Google Cloud, "Overview of model garden." Google Cloud Documentation, 2024.

[18] Hugging Face, "Hugging face partners with trufflehog to scan for secrets." Company Blog, 2024.

[19] K. Wiggers, "Hugging face says it detected 'unauthorized access' to its ai model hosting platform," 2024.

[20] Daily.dev, "Top 9 container registries 2024: How to choose." Technical Blog, 2024.

[21] S. Community, "Sigstore model-transparency: Supply chain security for ml." GitHub, 2024.

[22] J. Santos, T. Wauters, B. Volckaert, and F. De Turck, "Understanding i/o performance of ipfs storage: a client's perspective." ResearchGate, 2024.

[23] I. Labs, "Atlas: A framework for ml lifecycle provenance & transparency," 2024.

[24] Hugging Face, "Hugging face security documentation." Web Documentation, 2024.

[25] M. Fowler *et al.*, "Continuous delivery for machine learning." Martin Fowler's Blog, 2024.

[26] Z. Inc., "Zkml: An optimizing system for ml inference in zero-knowledge proofs." Medium, 2024.