Signing ML Artifacts: Building towards tamper-proof ML metadata records

Workstream 1: Software Supply Chain Security for AI Systems

Version 244c6a7fca256664b3ab680ca6a53e19c0f65f98

29 September 2025

Contents

Signing ML Artifacts: Building towards tamper-proof ML metadata records

OASIS Open Project : Coalition for Secure AI (CoSAI) Workstream 1: Software Supply Chain Security for AI Systems

Approved by the CoSAI Project Governing Board on 29 September 2025

1. Introduction

The rapid advancement of machine learning (ML) has brought unprecedented capabilities, but it has also introduced complex challenges in ensuring transparency, accountability, and integrity throughout the model development lifecycle. As ML models are supporting a broader range of use cases, increasingly influencing critical decisions across industries, the need for robust mechanisms to verify their provenance and authenticity has become paramount. This paper introduces model signing as a foundational concept to address these challenges, providing a framework to establish trust between model producers and consumers.

Traditional software supply chains have long faced risks such as dependency poisoning, infrastructure compromise, and version integrity attacks. However, the emergence of AI supply chains introduces unique vulnerabilities that demand novel solutions, many of these being covered in our earlier landscape paper, Establish Risks and Controls for the AI Supply Chain.

The consequences of these AI-specific threats are particularly severe due to three key differences from traditional software.

1. ***Expanded Scope of Impact***: compromised models can lead to biased or manipulated decisions at scale, often affecting multiple system outcomes and workflows at once, rather than single isolated functions.

2. ***Monitoring and Detection Complexity***: detection becomes more challenging as failures in AI systems are often subtle, requiring continuous monitoring of inherently non-deterministic processes, often across multiple interactions (session) rather than simple discrete observations

3. ***Increased Agency***: AI models are becoming more proficient at tasks coupled with Agentic AI systems that have the agency to perform in some cases autonomous actions increasing the repercussions as these systems are integrated into applications that grant additional agency.

ML models are inherently opaque, with their training data, pipelines, and dependencies often not easily inspectable by consumers. This opacity makes it difficult to detect compromises or ensure compliance. To increase transparency into these processes, we strongly recommend adopting the claimant model. This means the producer of the ML artifact makes one or more tamper-proof claims about the artifact in the form of *signed attestations*. The signature of the attestation ties the identity of the author to the contents of the attestation. Consumers of ML artifacts start by deciding if they want to trust the identity of these claims, before validating the signature on the claim. If the signature is valid, then consumers can believe the information contained in the attestation. If it turns out that the author has made a false claim, the consumer can reduce further trust on the identity of that author. By having the consumer decide what policies of trust to implement and enforce, we make it easier to react to supply chain compromises, without requiring strong formal systems.

The claims we are considering in this paper are intended to provide a verifiable record of a model's origin and transformations throughout its lifecycle, this effectively eliminates several avenues for supply chain compromise. There are three main categories of claims that can be made:

1. ***Integrity* claims**: They have the form "model X has been generated with this digest, as attested by identity A", and ensure that the subject artifact cannot be modified without invalidating the claim.

2. ***Provenance* claims**: They have the form "model X has been generated by following process P with inputs I, as attested by identity A". These claims ensure visibility between an artifact and what sources and dependencies were needed to build it.

3. ***Generic property* claims**: They have the form "model X has property P, as attested by identity A". These claims can, for example, attest in a tamper-proof way that a model has a specific test score when evaluating it against a benchmark dataset.

The first two sets of claims are claims usually found in software supply chains. Given the inherent complexities of ML systems, we need to also allow claims of the third type, to help in understanding the behavior of ML systems and reacting to incidents, for example, if a model is found to be biased, auditors can trace its lineage through these claims (as available), to identify compromised data sources of pipeline steps.

A common pattern across all these claims is the "as attested by identity A" part. This is tying the claim with the identity of the claimant. Consumers and organizations can enforce policies stating that only attestations produced by certain identities are trusted. Similarly, organizations can enforce other policies (e.g., rejecting models trained on untrusted data) to meet regulatory requirements through verifiable evidence.

To perform this tying, and to make the attestations tamper-proof, cryptographic signatures are needed. To hide their involvement, attackers must compromise multiple parties (e.g., both the infrastructure that generates the artifact and the claim, and the PKI infrastructure that is used to sign the attestation). This significantly increases the complexity of coordinated attacks.

For the remainder of this paper we will focus on model signing, as a way to generate signatures for attestations about models. However, the same approaches can be applied to the other ML artifacts (datasets, agents, etc.), once the software components are in place.

Model signing fosters informed decision-making, by enabling organizations to filter which models can run on their infrastructure, based on tamper-proof predicates, tied to trusted identities. Thus, it represents a crucial step toward building trust in AI systems, ensuring that the benefits of ML are realized without compromising security or ethical standards. As AI systems become increasingly integrated into critical infrastructure, the need for trust and accountability in the AI supply chain has never been greater and organizations should adopt this workflow as a key improvement. Model signing provides a practical framework to address these challenges, offering a balance between transparency and security. By adopting model signing, organizations can mitigate risks, ensure compliance, and build confidence in the AI technologies shaping our future. The OpenSSF model-signing library with its OMS model signing specification offers a practical implementation for generating model signatures while also aiming to offer support for signed provenance attestations in future releases.

2. Personas and Stakeholders

From a model signing perspective, we need to consider a specific set of actors and personas within the model development and distribution process. The following figure introduces the concepts of model producers and consumers as well as areas where signatures should be produced and verified.

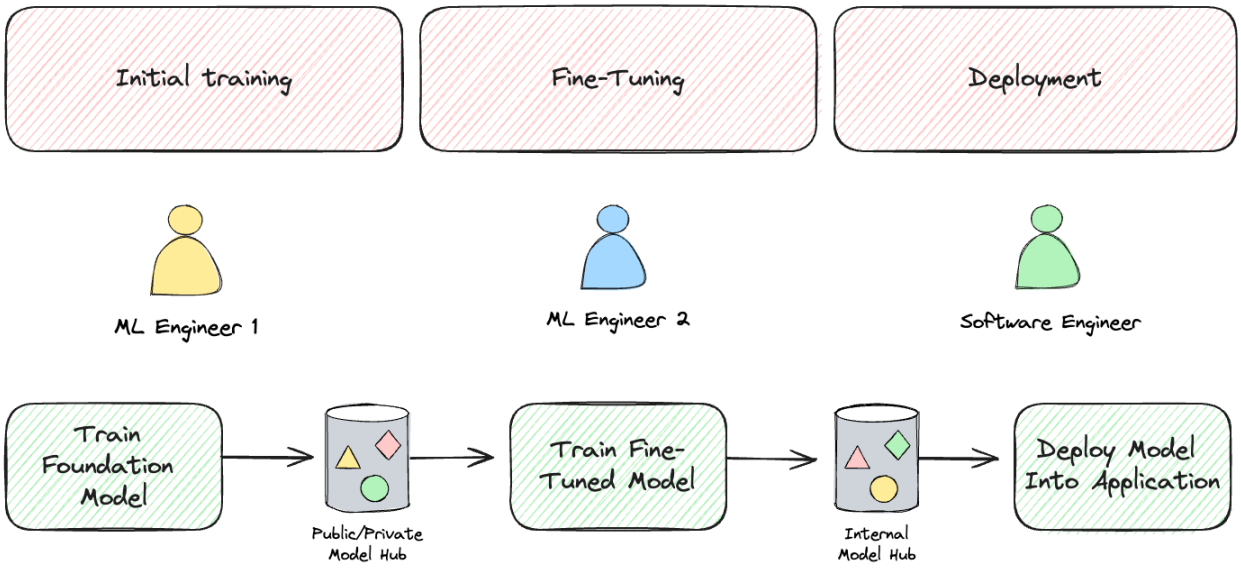

Figure 1: *signing lifecycle*

The signature generation and verification process is centered around the model artifacts that are produced and transformed until they are integrated into a full AI system, see section Defining the AI Supply Chain. Within the supply chain of the model artifacts, you might have multiple model producers and consumers. For steps that involve transformation and creation of derived models, the same entities might be both consumers and producers at the same time.

2.1 Model Producers

Model producers are the primary creators and distributors of machine learning (ML) artifacts, encompassing a diverse group of professionals and organizations. This persona includes data scientists & ML engineers, who design and train models; MLOps engineers, who provide infrastructure to support building, delivery and surveillance of these models; and organizations that oversee the entire lifecycle of ML development. The artifacts they produce drive the ML ecosystem, ranging from model weights and architectures to metadata and training pipelines. Similar to CI/CD pipelines for traditional software, these are the building blocks of AI systems, and their integrity, authenticity, and provenance are paramount to establishing trust with downstream consumers.

For model producers, signing ML artifacts offers several tangible benefits.

1. ***Enhances Trust***: Signing ML artifacts provides verifiable proof of the artifact's origin and integrity, enabling model consumers (customers) to verify integrity on receipt irrespective of the path it was sourced from. If a model is intentionally modified (e.g., via fine-tuning, distillation, or other effort), then a new signature may be attached, extending the lineage to allow additional integrity claims to be validated.

2. ***Fosters Accountability***: Producers that invest in better security practices, generating safe and reliable models, can be clearly identified and will be trusted more, having their models get a larger market share. This is particularly important in regulated industries, where compliance with standards and regulations is mandatory – with automatic verification of claims from trusted entities, adding a new model into these pipelines is a fast process.

3. ***Intellectual Property (IP) Protection***: Enables producers to assert ownership over their models and allow detection of unauthorized distribution of models – an entity A will not be

able to release a model and claim that it belongs to producer B if all models produced by B are already signed.

The absence of model signing exposes model producers to significant risks. Misuse of unsigned models can lead to unintended applications, while tampering can compromise model performance or introduce malicious behavior. Such incidents can result in reputational harm, loss of competitive advantage over time, eroding trust among consumers and stakeholders. Signing models gives producers the tools to enforce model integrity and a fundamental element for managing lineage. This makes them immune to claims of negligence or liability in case of failures, as well as enabling faster incident response avenues.

Model signing provides producers with specific guarantees, including authenticity (proof of origin), integrity (assurance that the artifact has not been altered), and provenance (traceability of the artifact's lifecycle). While model signing does not inherently guarantee model performance, fairness, or robustness – as these properties depend on the quality of the training process and data – it does allow adding these claims to a tamper proof model card or manifest that is rigorously bound to the model provided, further increasing trust in the model and the model producer. Producers must complement signing with rigorous testing, validation, and documentation to address these aspects comprehensively.

2.2 Model Consumers

Once an ML/AI model has been developed, tested and published then it may be used by a Model Consumer. Model Consumers integrate the models they consume into AI systems, they use them within their own toolchain to perform AI-supported tasks, or they use them to create derived models. In the latter case, a Model Consumer also becomes a Model Producer and must be attentive to maintaining appropriate provenance claims in that role.

Model Consumers take a few different forms, typically they fall into:

- Data Scientists, Researchers, AI/ML Engineers who integrate and evaluate the performance of the model within the AI system
- MLOps engineers, who operationalize and deploy these models
- AI governance teams that oversee various aspects like ethics, transparency, and security of the model
- Other parts of the organization that oversee the entire lifecycle

Model Consumers want to reduce their supply chain risks, therefore they need ways to verify that the artifacts they consume are really what the Model Producer intended to release. The verification of a model signature provides the guarantee that nobody tampered with the artifacts between the time a Model Producer assembled and signed the model artifacts and the time a Model Consumer verifies the signatures. That is, for integrity claims, the Model Consumer can now be assured that the model is the one that the Producer intended to release.

On the other hand, in case of provenance claims, or for attestations that also include details about model cards, model performance, etc., Model Consumers get additional trusted information: they know that everything that is included in the signature can be attributed to the Producer, being evaluated at the time the signature was generated. For example, Model Consumers can identify if a model has been trained on a specific dataset, or they can be assured that a model's performance on some test dataset is above a threshold, if these informations are present in the attestation.

In short, model signing provides a guarantee that nobody modified the consumed artifacts altering the intended behavior of the model from the provider. It can also offer a trusted channel for the Model Provider to deliver additional claims: e.g. in the scope of responsible AI, data confiden-

tiality, safety, security, bias, etc. It is also possible for any other provider to add new claims as to model capabilities, as separate attestations.

It is still the responsibility of the Model Consumer to ensure that the AI system, that integrates the model, adheres to the required AI governance standards. However model signing provides a solid fundament to perform root cause analysis and is a key element for AI explainability and tracing provenance.

Furthermore, model signing delegates the role of Model Distributor (someone who receives, stores, and redistributes models, like model repositories such as NVIDIA NGC or HuggingFace) to an entity that may strengthen the supply chain by independently validating the provided claims, while also being able to generate new ones. In any case, a major supply chain risk where a malicious / compromised Model Distributor could alter the model is completely eliminated. That is, Model Distributors cannot harm the trust relationship between the Model Producer and the Model Consumer, but they can provide additional information about the models..

3. Foundational Components of Model Signing

Model signing and verification can be leveraged at various phases of the model development life-cycle. A model producer must generate signatures for each model artifact produced, whether the artifact was generated from scratch or derived from a model through either retraining, fine tuning, distillation, quantization or any other technique altering a previous model artifact. The signature should be created as early as possible to allow the authentication of the entities that participated in its generation (actors, systems on which it was generated...). There is a difference between a claim made about the model at the end of the training process and the same claim made when the model gets published – a malicious or compromised insider could have tampered with the model in between these steps. However, having an attestation at the publishing point, in addition to the attestation from the training process is also worthwhile, establishing a *trusted publishing* process. Furthermore, attestations can provide verifiable proof about the state, authenticity, or integrity of a model or system.

Model consumers must verify model attestations when consuming a model, at each phase of the model consumption lifecycle, as close as possible to the use of the artifact, either locally, in a pipeline, or as a hosted service. The consumers must first authenticate the identity of the signature against a deployment policy in their organization, then verify the integrity of the model. If the signed attestation contains additional information, the consumer must validate these against the same deployment policy. The deployment policy might also include constraints that are not certified in the model signature itself (e.g., there might be a policy stating that models in pickle format are not allowed).

If the model consumer is also a model producer, the information that has been validated above can be forwarded to the attestations that are generated for the newly produced artifacts.

There are several factors and technical considerations that influence the choice of appropriate signing formats and when signatures should be created and consumed. Some tension may exist amongst these elements including but not limited to

- Security robustness
- Development/integration effort
- Deployment and verification speed
- Signing complexity and ease of use

As mentioned before, a signature should be generated ideally as close as possible to the creation

of the model artifacts, but at the latest at the moment of storing the artifacts in a model hub. Similarly, while a signature should be verified immediately after downloading the model artifacts from a model hub, ideally the verification would happen also immediately before its consumption, e.g. just before loading the model from disk into memory.

Model Registries/Hubs should perform signature verifications between the time an artifact is uploaded (along with its signature) and the time a consumer triggers the download. While this adds limited value to the model consumer, which should verify the signature anyways, it still increases the security posture for the hub provider's internal infrastructure through integrity verification. Suspicious models, invalid or incomplete signatures, additional components added to but not captured in the manifest could be rejected or other policies enforced by the hub provider to ensure the trust and reliability of its offerings are maintained. Furthermore, the model hub itself could act as a redistributor of the model (e.g., adapting the model to a new format) as well as generating new claims about the model or validating existing claims.

The exact choice of the appropriate locations and time to produce and verify signatures depends on the overhead, technical limitations, impact and threat analysis related to the specific workflows used by the involved personas.

3.1 What is different for model signing?

Outside of ML, there are a plethora of solutions for code and artifact signing. So, a natural question arises about what is different in ML that a different solution is needed.

Models are large, complex assemblies. A model may consist of several files and can consist of model weights, configuration files, tokenizers, and training logs. Signing these artifacts separately could cause integrity drift, as well as complicating the verification story for consumers – they will need to retrieve multiple artifacts, multiple signatures, and then execute multiple verification steps. Thus, a signed manifest over the entire collection of model files should be used instead.

The signed manifest represents a cryptographically signed collection of hashes for each artifact in the model. Small models could use a single hash across the entire set of files, whereas moderate and large models will need to compute hashes for each file individually. This is an optimization process, allowing for hashing in parallel. It also enables verifiers to verify the integrity of only the components of the models that they are using – this is particularly important for the case when the model hub serves the same model as a collection of files for different model architectures (e.g., Safetensors, pickles, etc.), but the user only uses one of them. Finally, for extremely large models, we can add a new optimization layer by computing hashes for portions of large files.

In any case, once the collection of hashes is computed, the manifest should be transformed to a canonical form and then signed. As part of the canonicalization process, a root hash could also be computed to allow referring to the model by one single digest, across various supply chain documents (e.g., SLSA attestations).

On the consumer side, verification proceeds by validating the signature and extracting the signed manifest into a collection of expected hashes. Then, the verifier should re-compute the hashes of the model artifacts as present on disk and compare each one with the equivalent from the signed manifest. The model signature is valid only if all components match.

Before the signature can be written to disk, we need to define how we represent the canonicalized manifest. Given that scalability and performance are critical for ML environments, where high throughput and training performance are stringent requirements, we need to pick a format that permits generating the manifest in a robust way. For the model–signing project, the OMS standard proposes encoding the manifest as an in-toto statement that ties every model artifact with its corresponding hash. An alternative version would be to use the SPDX (Software Package Data Exchange) format, given that SPDX is an internationally recognized standard (ISO/IEC 5962:2021)

for representing Software Bills Of Materials (SBOMs) and is widely adopted across various industries. Alternatively, the manifest could be represented as a C2PA manifest (see also Project Atlas), or, mapped to the OCI Image specification for Docker images. However, all of these alternate formats provide more functionality than what is needed for just the model signature.

Once the manifest is serialized, we need a way to also pair it with its signature. One option would be to extend the model format to contain the signature as part of the model itself, but this would require adapting an ever growing number of model formats and not be scalable. Instead, having a detached signature that applies the same way regardless of the model format is a better approach. A lightweight solution like DSSE (Dead Simple Signing Envelope) is a good fit here, due to its simplicity and ease of implementation – which is why this is the preferred approach for the model-signing project. However, for more complex workflows that require robust security and interoperability, formats like PKCS#7 or COSE are more suitable. These standards offer comprehensive features but may be more complex to implement.

At this point, the signed manifest is available, but there is still one open question left: how do we sign it. Traditionally, signing keys and certificates have been used for generating digital signatures, but these run the risk of a leakage allowing attackers to sign malicious artifacts (as in the SolarWinds incident). An emerging standard is to use short-lived signing certificates, like those proposed by Sigstore. In order to be PKI-agnostic, supporting all these approaches, the model-signing project has decided to adopt the Sigstore bundle format for the signature: attach the signature verification material to the signed DSSE envelope. However, compared to a pure Sigstore signature, in the OMS signature the verification material can also be generated from traditional PKI approaches.

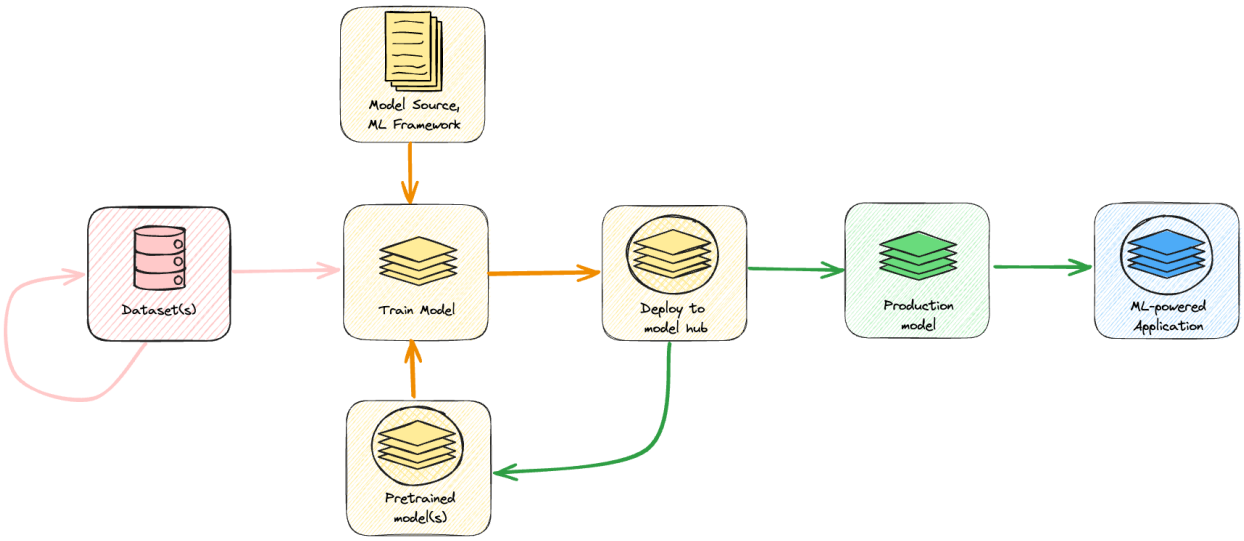

Figure 2: *sigstore bundle*

There are other industry standards for generating signatures, depending on security requirements, preferred security frameworks, DevSecOps practices, etc. As an example, the Notary Project and in-toto can both sign components of the supply chain, ensuring integrity and authenticity of artifacts, although these approaches don't scale for large models. Furthermore, Notary requires integration with Docker and other tools, while most ML developers prefer to serve the models as self-contained entities. Alternatively, COSE could be used, although it is complex to implement and manage, especially for large-scale workflows, or non-CBOR systems. PKCS#7 is another alternative, but it produces large signatures and performance overheads. If you are already using Docker or Kubernetes, Notary and Harbor can integrate seamlessly with

your existing tools. For those heavily invested in AWS, AWS Signer offers a managed service that integrates well with other AWS services, providing a streamlined signing experience. However, it is tied to the AWS ecosystem, which may limit flexibility and potentially incur higher costs associated with managed services.

Given all these constraints and shortcomings of the existing approaches, we recommend the OMS format, as implemented by the model–signing project and described in this section. This is already adopted by industry partners, as we will show in a case study further down in this paper. This format also has the advantage of being extensible: new predicates could be added to the in-toto statement to record additional properties. For example, if the model producer wants to provide a model card with the signature, this could be represented as a new field in the statement.

4. Maturity Levels and Adoption

The implementation of model signing in AI/ML systems naturally evolves through progressive stages of sophistication (i.e., maturity levels), each addressing increasingly complex trust and provenance requirements. This evolution reflects the practical reality that organizations must balance immediate security needs with longer-term vision, technical capabilities with resource constraints, and current industry standards with future regulatory requirements.

Maturity Level	Description
1: Basic Artifact Integrity	Models treated as opaque binary artifacts requiring cryptographic protection. The model signing process creates a cryptographic binding of the model's content to the identity of its creator establishing both authenticity and integrity guarantees.
2: Signature Chaining and Lineage	Models treated as components within complex, interconnected development ecosystems. Signature chaining extends the cryptographic guarantees of basic digital signing by establishing verifiable relationships between models and their dependencies. Provides a comprehensive trail of trust that enables sophisticated provenance tracking and enhanced supply chain security measures.
3: Structured Attestations, Model Provenance, and Policy Integration	A comprehensive provenance system that captures rich, structured information about model development processes, properties, and compliance status. Represents the convergence of cryptographic signing with AI governance, enabling organizations to implement automated policy controls evaluations at different phases of the model development lifecycle

Understanding these maturity levels provides a foundation for organizations to make informed decisions about their model signing implementations while establishing a roadmap for enhanced security and governance capabilities.

The evolution of model signing capabilities must be aligned with broader organizational strategies for AI governance and risk management. Model signing should not be viewed as an isolated

security measure but as an integral component of comprehensive AI governance frameworks that include model validation, bias testing, performance monitoring, and regulatory compliance. This alignment ensures that investments in model signing capabilities at each maturity level deliver tangible security and compliance benefits while building the foundation for more advanced capabilities while supporting broader organizational objectives for responsible AI development and deployment.

5. Future directions and standardization

The continued evolution of model signing technology and practices will be shaped by advancing cryptographic capabilities, emerging regulatory requirements, and the growing sophistication of AI/ML systems. Organizations implementing model signing systems today must design for adaptability and extensibility, ensuring that their investments remain valuable as the technology landscape evolves.

5.1 Story of NVIDIA

As AI systems become more agentic, able to take actions and interact with external environments, the risk of model compromise becomes a direct security concern. A single tampered model can be leveraged to steal data, misuse system access, or trigger cascading failures across critical infrastructure. Trust in model integrity cannot be assumed; it must be verified.

To address this challenge, NVIDIA partnered with industry leaders in the OpenSSF AI/ML Security Working Group to define the OpenSSF Model Signing (OMS) standard. The specification supports both traditional PKI and keyless signing with Sigstore, allowing publishers to easily adopt with minimal changes or disruption to existing workflows. Since March 2025, every NVIDIA model published in the NGC Catalog has shipped with an OMS signature, making NGC the first major model hub to deliver verifiable model trust at scale.

Integrating OMS into the publishing pipeline was seamless and created immediate security benefits. Just as important, it reinforces stronger release management practices, from provenance claims to automated checks. Signing establishes a higher bar for the AI supply chain, and NVIDIA encourages all model publishers to adopt OMS to make verified trust the default.

5.2 Further Standardization

Standardization efforts across the industry will play a critical role in the widespread adoption of model signing practices. The effectiveness of all model signing approaches, from basic artifact integrity to comprehensive provenance systems, depends significantly on the adoption of standardized metadata schemas and verification protocols across the AI supply chain. Rather than relying on individual organizational approaches, industry-wide standards and frameworks are essential for enabling interoperability and automated validation capabilities. Emerging standards such as OMS and SPDX for AI/ML artifacts and extensions to existing supply chain security frameworks like in-toto provide the foundation for consistent implementation across diverse organizations and toolchains. These standards enable the development of automated validation tools that can verify signatures and attestations across organizational boundaries, supporting the collaborative nature of modern AI development. Organizations should actively participate in standardization efforts, contributing their expertise and requirements to ensure that emerging standards meet practical needs while maintaining appropriate security guarantees.

The practical reality of mixed-maturity environments presents significant interoperability challenges that must be addressed through thoughtful design and implementation strategies. Organizations frequently find themselves operating in heterogeneous ecosystems where supply chain partners, model repositories, and deployment environments operate at different maturity levels. A comprehensive provenance system attempting to consume models from partners using basic signing must gracefully degrade verification capabilities while maintaining appropriate security

guarantees. Similarly, organizations implementing signature chaining must account for scenarios where upstream dependencies lack the detailed attestations required for complete lineage tracking, requiring fallback mechanisms that preserve partial verification capabilities.

Compatibility challenges with existing AI/ML toolchains compound these interoperability concerns, as model signing implementations must integrate with diverse frameworks, registries, and deployment systems that may have varying levels of signing support. Organizations must develop bridging mechanisms that enable signature verification across different tool ecosystems while avoiding vendor lock-in or excessive complexity. This often requires implementing abstraction layers that can translate between different signature formats and metadata schemas, ensuring that core security guarantees are preserved regardless of the specific tools used throughout the model lifecycle. The successful navigation of these interoperability challenges requires careful architectural planning that anticipates the diverse and evolving nature of AI/ML toolchains while maintaining backward compatibility and migration paths as standards mature.

The integration of model signing with emerging technologies such as confidential computing, federated learning, and edge deployment will create new challenges and opportunities. These technologies introduce additional trust and verification requirements that must be addressed through advanced signing and attestation mechanisms. The development of signing approaches that can operate in these constrained environments while maintaining security guarantees represents an important area for future research and development.

The convergence of model signing with regulatory compliance requirements will drive continued innovation in attestation and verification technologies. As AI regulations become more specific and demanding, organizations will require increasingly sophisticated capabilities for demonstrating compliance through cryptographic proof. The development of automated compliance systems that can generate regulatory reports directly from signing and attestation data represents a significant opportunity for reducing compliance costs while improving accuracy and completeness.

6. Contributors and Acknowledgements

Workstream Leads

- Matt Maloney, Cohere (mattmaloney@cohere.com)
- Andre Elizondo, Wiz (andre.elizondo@wiz.io)
- Jay White, Microsoft (jaywhite@microsoft.com)

Editors

- Mihai Maruseac, Google (mihaimaruseac@google.com)
- John Stone, Google (thestone@google.com)
- Daniel Rohrer, NVIDIA (drohrer@nvidia.com)
- Danilo Tommasina, Thomson Reuters (danilo.tommasina@thomsonreuters.com)
- Yassine Ilmi, Thomson Reuters (yassine.ilmi@thomsonreuters.com)

Contributors

- Adam Tuaima, TrendMicro (adam_tuaima@trendmicro.com)

- Arber Salihi, Thomson Reuters (Arber.Salihi@thomsonreuters.com)
- Barak Sharoni, Wiz (barak.sharoni@wiz.io)
- Eoin Wickens, HiddenLayer (eoin@hiddenlayer.com)
- Jigisha Mavani, IBM (jigisha.mavani@ibm.com)
- Judith Furlong, Dell Technologies (Judith.Furlong@dell.com)
- Stefan Berger, IBM (stefanb@us.ibm.com)
- Martin Sablotny, NVIDIA (msablotny@nvidia.com)

Technical Steering Committee Co-Chairs

- Akila Srinivasan, Anthropic (akila@anthropic.com)
- J.R. Rao, IBM (jrrao@us.ibm.com)

7. Appendix

7.1 CoSAI Focus

CoSAI is an OASIS Open Project, bringing together an open ecosystem of AI and security experts from industry-leading organizations. The project is dedicated to sharing best practices for secure AI deployment and collaborating on AI security research and product development. The scope of CoSAI is specifically focused on the secure building, integration, deployment, and operation of AI systems, with an emphasis on mitigating security risks unique to AI technologies. Other aspects of Trustworthy AI are deemed important but beyond the scope of the project including, ethics, fairness, explainability, bias detection, safety, consumer privacy, misinformation, hallucinations, deep fakes, or content safety concerns like hateful or abusive content, malware, or phishing generation. By concentrating on developing robust measures, best practices, and guidelines to safeguard AI systems against unauthorized access, tampering, or misuse, CoSAI aims to contribute to the responsible development and deployment of resilient, secure AI technologies.

7.2 Guidelines on usage of more advanced AI systems (e.g. large language models (LLMs), multi-modal language models. etc) for drafting documents for OASIS CoSAI:

tl;dr: CoSAI contributions are actions performed by humans, who are responsible for the content of those contributions, based on their signed OASIS iCLA (and eCLA, if applicable). [Each contributor must confirm whether they are entitled to donate that material under the applicable open source license; OASIS and the CoSAI Project do not separately confirm that.] Each contributor is responsible for ensuring that all contributions comply with these AI use guidelines, including disclosure of any use of AI in contributions.

- Selection of AI systems: CoSAI recommends the use of reputable AI systems (lowering the risk of inadvertently incorporating infringing material).
- Model constraints: Currently, CoSAI or OASIS are not required to have a contract or financial agreement for using AI systems from specific vendors. However, CoSAI editors should consider employing varying tools to avoid potential fairness concerns among vendors.
- IP infringement: It is the responsibility of the individual who subscribes/prompts and

receives a response from an AI system to confirm they have the right to repost and donate the content to OASIS under our rules.

- Transparency: CoSAI's goal will be to maintain transparency throughout the process by documenting substantial use of AI systems whenever possible (e.g., the prompts and the AI system used), and to ensure that all content, regardless of production by human or AI systems, was reviewed and edited by human experts. This helps build trust in the standards development process and ensures accountability.

- Human-edited content and quality control: CoSAI mandates human-reviewed or -edited results for any final outputs. A robust quality control process should be in place, involving careful review of the generated content for accuracy, relevance, and alignment with CoSAI's goals and principles. Human experts should scrutinize the output of AI systems to identify any errors, inconsistencies, or potential biases.

- Iterative refinement: The use of AI systems in drafting standards should be seen as an iterative process, with the generated content serving as a starting point for further refinement and improvement by human experts. Multiple rounds of review and editing may be necessary to ensure the final standards meet the required quality and reliability thresholds.

7.3 Copyright Notice