Rethinking Certification for Trustworthy Machine-Learning-Based Applications

Marco Anisetti, *Università degli Studi di Milano, 20133, Milan, Italy*

Claudio A. Ardagna, *Università degli Studi di Milano, 20133, Milan, Italy*

Nicola Bena, *Università degli Studi di Milano, 20133, Milan, Italy*

Ernesto Damiani, *Università degli Studi di Milano, 20133, Milan, Italy*

Machine learning (ML) is increasingly used to implement advanced applications with nondeterministic behavior, which operate on the cloud–edge continuum. The pervasive adoption of ML is urgently calling for assurance solutions to assess applications' nonfunctional properties (e.g., fairness, robustness, and privacy) with the aim of improving their trustworthiness. Certification has been clearly identified by policy makers, regulators, and industrial stakeholders as the preferred assurance technique to address this pressing need. Unfortunately, existing certification schemes are not immediately applicable to nondeterministic applications built on ML models. This article analyzes the challenges and deficiencies of current certification schemes, discusses open research issues, and proposes a first certification scheme for ML-based applications.

Modern applications consist of elastic server-side processes running in the cloud, implemented as micro- and nanoservices developed with cloud-native technologies and orchestrated at runtime. The availability of new orchestration platforms and programming frameworks is making it possible to execute these applications at line speed.[14] In the fullness of time, an intelligent cloud continuum will support autonomous, self-configuring applications that will request the activation of its services wherever they are needed, including innovative Internet of Things devices.[1] While this new paradigm promises many advantages in terms of availability; elasticity; and, ultimately, quality of service, its complexity is much higher than that of its predecessors. This complexity leap is affecting the governance, risk, and compliance landscape and even the procedures to guarantee the security and safety of citizens.

To understand why, let us discuss where we stand right now in terms of available solutions to assess and verify applications behavior. . . .

Digital Object Identifier 10.1109/MIC.2023.3322327
Date of publication 6 October 2023; date of current version 6 November 2023.

Traditionally, assurance techniques have been used to assess and verify the trustworthiness of a system or application.[6] Over the years, certification has become the most popular assurance technique, providing a way for a trusted authority to assert that a system or application supports a given nonfunctional property (or set of nonfunctional properties) according to some evidence on its operation. Test-based certification schemes have been applied to software systems since the 1980s, with the release of the *Orange Book*.[a] Test-based schemes underwent a crisis in the middle of the past decade, when it became clear that certifying service-based applications, even with the low degree of autonomy available at the time, required monitoring and runtime reverification, as different services were recruited at each execution. In 2012, on the crest of the service-oriented computing wave, the seminal ASSERT4SOA project[b] led the way toward a new generation of dynamic certification techniques with the following slogan: *You live in a certified house, you drive a certified car, why would you use an uncertified service?* At that time, certification enabled users to select and compose applications on the basis of their certified properties.

[a]U.S. Department of Defense. "Department of Defense Trusted Computer System Evaluation Criteria". Dec. 1985.
[b]https://cordis.europa.eu/project/id/257351/results

Today, a second crisis of certification schemes is looming, as the massive adoption of machine learning (ML) is radically transforming applications behavior.[9]

Opaque ML models pose new concerns on how to test and certify the trustworthiness, safety, and reliability of the applications. The challenges we faced in 2012 are back today, like the most classic of Groundhog Days. A new slogan is, then, emerging: *You use certified services, you hire certified professionals, why would you use an application driven by uncertified ML?*

This question sheds some light on the motivations of the increasing push toward the definition of sound techniques for the nonfunctional certification of ML-based applications.[7,13] What needs to be certified is the *behavior of the application driven by ML models*, rather than some theoretical notion of the ML models. For instance, fairness means there is no discrimination among users accessing an application; privacy and security mean that the application safely processes user data; and robustness means that the application will keep operating reliably while under attack. Nevertheless, these properties depend on the ML models driving the application and on the process/data used to train them. Training data can be partial or inaccurate (affecting fairness), poisoned (affecting robustness and security), and sensitive (affecting privacy and explainability).

This article outlines the key elements of a sound certification scheme for ML-based applications. To this aim, we start with current certification schemes and analyze their limitations. We then give a first definition of a certification scheme for ML-based applications, based on the simultaneous verification of three factors: 1) the data used for training, 2) the training process, and 3) the ML model. We finally apply the scheme in a real-world example of its application.

TRADITIONAL CERTIFICATION SCHEMES

Traditional certification schemes evolved into dynamic schemes suitable for deterministic composite applications, where services are orchestrated at runtime according to their nonfunctional properties.[2] Application properties are inferred from the properties of the applications components and continuously verified across service changes.[5,10]

Figure 1 shows the typical process of a certification scheme. The process starts with a certification authority (CA) defining the certification model, that is, a specification of the activities to be executed to certify an application. Formally, the certification model is a tuple $\mathrm{CM} = \langle p, \mathrm{ToC}, \mathcal{E} \rangle$, where p is the *nonfunctional property* to be certified (e.g., confidentiality); ToC is the *target* of certification, that is, the application to be certified (e.g., a cloud-based application); and $\mathcal{E}$ is the *evidence collection model* (e.g., a set of test cases). An accredited lab, delegated by the CA, executes $\mathcal{E}$ to collect the *evidence* that may support the awarding of a *certificate* proving p on ToC.[2,6]

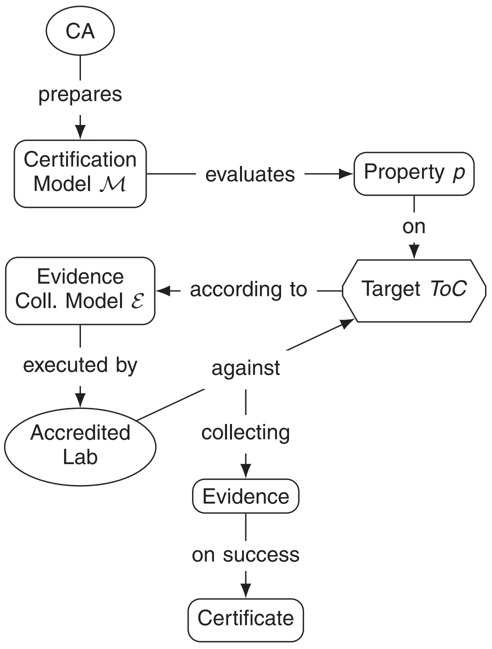

FIGURE 1. Traditional certification process. CA: certification authority.

Recently, certification schemes broaden the definition of target of certification[2] toward multifactor certification. Multifactor certification is a natural evolution of certification schemes to accommodate the peculiarities of modern applications. The nonfunctional posture of an application, in fact, depends on its software artifacts as well as on the processes that brought it to operation (e.g., development and deployment processes).

However, even dynamic, multifactor certification schemes struggle with the latest evolution of modern applications toward ML.

CHALLENGES IN ML-BASED APPLICATION CERTIFICATION

Today, the certification of ML-based applications ("ML certification" in the following) is more an art than a science,[8] resulting in ad hoc solutions tailored to specific properties (e.g., explainability, fairness, and robustness[4,17,19]). Research is at a standstill: no certification scheme for ML is available, despite the increasing push coming from society[7]; at the same time, none of the existing system certification schemes[6] can be adapted to the certification of ML. We argue that this standstill is due to four unsolved challenges, which we designate here as C1—C4.

To exemplify the challenges, we introduce a reference scenario that considers a malware detection application ("malware detector" in short). It is based on a deep learning model trained on real data collected from the field as well as synthetic data generated according to a generative adversarial network.[3] Data are performance metrics retrieved from the underlying system. To operate in an adversarial environment, the malware detector must be certified for property *robustness against inference-time attacks*, that is, its ability to correctly operate in the presence of adversaries interfering with the malware detection process by perturbing collected data.

C1: Target Definition

A target of certification ToC is commonly defined as a list of components (i.e., endpoints, services, and functions) with clear and unambiguous (i.e., deterministic) behavior. This approach does not suit applications whose components are not deterministic.[19] Current definitions of ToC are inapplicable to ML-based applications and must evolve to accomplish the uncertainty introduced by ML models. In our reference scenario, the behavior of the deep learner must be certified in terms of the robustness of the dataset and process used for training as well as the characteristics of the learned model.

C2: Property Definition

The literature is rich for well-formalized nonfunctional properties (e.g., k anonymity for privacy, confidentiality, integrity, and availability for security), where property definition is decoupled from property verification. The latter is, in fact, left to the evidence collection model. ML certification, instead, lacks commonly accepted and rigorous definitions of ML properties (e.g., explainability, fairness, and robustness), where property verification must be included in the property definition. Property verification, in fact, substantially characterizes the property itself and defines the means driving evidence collection.[4,17] For instance, in our reference scenario, property ML robustness must specify how adversarial samples are crafted for the runtime verification of adversarial attacks.

C3: Certification Process

The certification process relies on evidence collection models executing test cases or monitoring rules to collect the evidence at the basis of a certificate award.[6] Traditional evidence collection statically assesses application interfaces, which might be insufficient to certify the behavior of an ML-based application. An evidence collection model for ML certification must consider three factors, namely, the (training) data, (training) process, and ML model itself. In particular, the data factor is novel and must consider how a model is learned (i.e., developed), including the specific characteristics of the training set. The latter is completely neglected in traditional systems certification and would compare to the evaluation of the application developer (e.g., her experience and skills).

C4: ML Pipelines

The structure of an ML-based application is recursive: each of its components can implement an ML pipeline orchestrating other components implementing an aspect of ML, from data ingestion to data processing. Certification schemes must support such a structure. In our reference scenario, the certification of property robustness should consider the robustness of the ML model against inference-time attacks as well as the integrity of retrieved performance metrics. In some cases, this can be achieved by customizing existing solutions for certificate composition.[5] However, this challenge remains an open issue that we leave for our future work.

ML-BASED APPLICATION CERTIFICATION

To address challenges C1–C3, we need to reshape traditional certification schemes according to three main aspects: *1) the multifactor certification* of ML-based applications behavior (challenge C1), *2) an ML-specific nonfunctional properties* (e.g., fairness, explainability, and robustness) definition (challenge C2), and *3) ML-specific evidence collection models* supporting nonfunctional properties verification at point 2 (challenge C3).

Multifactor Certification of ML-Based Application Behavior

Multifactor certification schemes[2] are the natural choice for certifying ML-based applications. Three factors f should be considered as follows:

- The *data factor* (f_d) includes information on the dataset used for model training/validation (e.g., characteristics of the samples).
- The *process factor* (f_p) includes information on the training process (e.g., the adoption of boosting and transfer learning).
- The *model factor* (f_m) includes information on the behavior of the ML model in operation.

Our *three-factor* certification model CM is a set of three independent certification models $\{\mathrm{CM}^{f_d}, \mathrm{CM}^{f_p}, \mathrm{CM}^{f_m}\}$, where each CM^* is a tuple of the form $\langle p^*, \mathrm{ToC}^*, \mathcal{E}^*\rangle$. Each certification model implements a certification process as follows.

Certification model $\mathrm{CM}^{f_d} = \langle p^{f_d}, \mathrm{ToC}^{f_d}, \mathcal{E}^{f_d}\rangle$ (*data factor*) evaluates the dataset used for training and its impact on the ML model. For instance, a poisoned training set negatively impacts the property robustness of the malware detector since it reduces its ability to distinguish between benign and malign samples. CM^{f_d} includes a property p^{f_d} specific for data, a target ToC^{f_d} modeling the dataset used for training/validation, and an evidence collection model $\mathcal{E}^{f_d}$ specifying the procedure for collecting evidence on the ToC, including evidence on dataset balancing and feature extraction.

Certification model $\mathrm{CM}^{f_p} = \langle p^{f_p}, \mathrm{ToC}^{f_p}, \mathcal{E}^{f_p}\rangle$ (*process factor*) expresses how the training process is implemented and its impact on the ML model. For instance, a sanitization technique for fixing poisoned samples in the dataset positively impacts the property robustness of the malware detector, decreasing the shift in the learned classification boundaries. CM^{f_p} includes a property p^{f_p} specific for the training process; a target ToC^{f_p} modeling the training process; and an evidence collection model $\mathcal{E}^{f_p}$, including the procedure for collecting evidence on the design and execution of the training process, such as the inspection of checkpoints generated during training.

Certification model $\mathrm{CM}^{f_m} = \langle p^{f_m}, \mathrm{ToC}^{f_m}, \mathcal{E}^{f_m}\rangle$ (*model factor*) evaluates the ML model in operation. It is a crucial target of certification, and its verification is strongly intertwined with the property to be verified. For instance, property robustness of the malware detector can be actively tested, assessing its ability to spot an adversary trying to inject adversarial samples to alter the ML model predictions. As another example, property privacy can be compromised by attacks that infer the presence of a sample in the training set. This is done by inspecting the ML model predictions.[18] However, if the application returns the predicted label only, the attack fails, and privacy is preserved despite the lack of a specific protection. CM^{f_m} includes a property p^{f_m} specific for the ML model; a target ToC^{f_m} describing the ML model (e.g., its architecture and parameters); and an evidence collection model $\mathcal{E}^{f_m}$, including the procedure for collecting evidence on the behavior of the ML-based application, such as functions exercising the ML model.

ML-Specific Nonfunctional Properties

ML certification requires the definition of ML-specific nonfunctional properties (e.g., fairness, explainability, robustness, and safety) and the redesign of traditional nonfunctional properties (e.g., confidentiality, integrity, and availability). In general, these properties are the union of factor-specific properties p^{f_d}, p^{f_p}, and p^{f_m} and must specify the verification means driving evidence collection (see challenge C2). We note that, depending on the property, some factors might not be relevant, and verification means might be neglected.

Let us consider property robustness in our reference scenario. It is the union of the property robustness of the training set (p^{f_d}), the training process (p^{f_p}), and the ML model (p^{f_m}). For example, the property robustness of the training set (p^{f_d}) is defined as the absence of targeted poisoning in the training set; its definition includes the function used to detect poisoned points.

Let us, then, consider property integrity. It traditionally proves the integrity of the application and its artifacts. The property integrity of ML, instead, proves the integrity of the ML model behavior according to the three factors. Property integrity in the data factor is the integrity of training data (e.g., verifying that the dataset cannot be altered). Property integrity in the process factor is the integrity of the training process (e.g., a training process that includes an adversarial training specification). Property integrity in the model factor is the integrity of the generated ML model (e.g., verifying that the packaged ML model is tamperproof).

ML-Specific Evidence Collection Models

Evidence collection models are factor specific and describe how to collect evidence according to the three factors. All evidence and metadata collected during the certification of each factor must be stored in certificates to ensure reproducibility and trustworthiness.

Let us consider property robustness in our reference scenario. $\mathcal{E}^{f_p}$ and $\mathcal{E}^{f_m}$ must be adapted to collect data coming from the training process and the ML model coping with their opaqueness. For instance, $\mathcal{E}^{f_p}$ verifies the robustness of the training process by ensuring the usage of strengthening techniques. $\mathcal{E}^{f_m}$ crafts and sends adversarial samples to verify robustness.

In addition, let us consider an evidence collection process $\mathcal{E}^{f_m}$ for property privacy. This property can be verified in terms of the (in)ability to reverse engineer training data while operating the ML model.[18,20]

CERTIFICATION IN ACTION

We demonstrate our ML certification scheme in our reference scenario, which considers a malware detection application (see Anisetti et al.[3] for more details on the malware detector). We recall that the property of

TABLE 1. Summary of the certification scheme applied to our scenario.

f	*p*	ToC	$\mathcal{E}$	Outcome
f_d	Robustness against inference-time attacks (Absence of adversarial poisoning in the training set)	Training set	› •Check the presence of poisoned samples › •*Evidence*: poisoned samples $= \emptyset$	✓
f_p	Robustness against inference-time attacks(Usage of randomized smoothing and adversarial training)	Training process	› •Check the training process and checkpoints › •*Evidence*: training process does not include *randomized smoothing* and *adversarial training*	✗
f_m	Robustness against inference-time attacks(Ineffectiveness of adversarial attacks)	Malware detector (deep learning model)	› •Craft and send adversarial samples to the machine learning model › •*Evidence*: recall $= 0$	✗

interest is *robustness against inference-time attacks*, that is, the ability of the malware detector to behave correctly in the presence of an adversarial perturbation aimed to hide malware activities. Table 1 summarizes the three factors and their outcomes.

Data Factor

The data factor defines a certification model $\mathrm{CM}^{f_d} = \langle p^{f_d}, \mathrm{ToC}^{f_d}, \mathcal{E}^{f_d} \rangle$ as follows.

- Property robustness p^{f_d} is defined as the *absence of poisoned samples from the training set*. These are samples injected in the training set to alter the classification boundaries learned by the model, thus masking malware activities at inference time.[16]
- Target ToC^{f_d} is the training set.
- Evidence collection model $\mathcal{E}^{f_d}$ adapts the poisoning removal technique of Peri et al.[15] to flag possibly poisoned samples.

In our scenario, no poisoned samples are retrieved, and the assessment for the data factor f_d is positive (✓).

Process Factor

The process factor defines a certification model $\mathrm{CM}^{f_p} = \langle p^{f_p}, \mathrm{ToC}^{f_p}, \mathcal{E}^{f_p} \rangle$ as follows.

- Property robustness p^{f_p} is defined as the *usage of two strengthening techniques during the training process*: *randomized smoothing* and *adversarial training*. These techniques reduce the presence of poisoned samples and the impact of adversarial samples, respectively.
- Target ToC^{f_p} is the training process. It does not include any strengthening techniques.
- Evidence collection model $\mathcal{E}^{f_p}$ inspects the training code as well as the checkpoints created during training to retrieve evidence on the usage of *randomized smoothing* and *adversarial training*.

In our scenario, the process inspection fails due to the lack of both techniques, and the assessment for process factor f_p is negative (✗).

Model Factor

The model factor defines a certification model $\mathrm{CM}^{f_m} = \langle p^{f_m}, \mathrm{ToC}^{f_m}, \mathcal{E}^{f_m} \rangle$ as follows.

- Property robustness p^{f_m} is defined as the effectiveness of adversarial attacks against the ML model. It specifies how samples of adversarial attacks are crafted and sent to the model.
- Target ToC^{f_m} is the malware detector.
- Evidence collection model $\mathcal{E}^{f_m}$ exercises the malware detector, sending adversarial (malware) samples and verifying whether the recall retrieved on such samples is larger than 0.95.

In our scenario, recall on adversarial samples is zero, meaning that they are all misclassified as benign. The assessment for model factor f_m is negative (✗).

To conclude, the malware detector lacks the proper robustness to operate in adversarial settings. Although it might appear that f_m is the only relevant factor and that traditional certification is sufficient, the certification of the three factors allows final users to completely understand how the data, the training process, and the model in operation jointly contribute or not to the requested nonfunctional property. For instance, failure in the process factor motivates failure in the model factor. A certificate can be awarded for each factor independently in the case of a positive outcome (✓), or a single certificate can be released by merging the three assessments according to predefined rules.

CONCLUSION

This article takes a first step toward the certification of ML-based applications. The road ahead is still long and difficult. Our evidence collection techniques based on the analysis of the training set, training process, and ML model predictions (Table 1) must be complemented by other approaches, such as *abstract interpretation*[11] or inspection of a surrogate white-box model.[12] The lifecycle of ML models and their certificates must be carefully managed. ML models need, in fact, to be continuously adapted according to evolving system behavior, novel requirements, and drift in the incoming data, to name but a few. This requires certification to follow the changes of ML models along their lifecycle. However, certification should not be limited to tracking changes; it should drive ML model evolution so that ML models can evolve while supporting the desired properties in all factors.

To conclude on an optimistic note, we are confident that certification will eventually succeed in supporting trust in ML-based application behavior. It is a good omen that ongoing regulatory discussions[7,13] agree that certification should target all artifacts involved in ML training and operations.

ACKNOWLEDGMENTS

The work was partially supported by the following projects: 1) the Multilayered Urban Sustainability Action project, funded by the European Union (EU)–NextGenerationEU under the National Recovery and Resilience Plan (NRRP) Mission 4 Component 2 Investment Line 1.5: Strengthening of research structures and creation of R&D “innovation ecosystems,” made up of “territorial leaders in R&D” (CUP G43C22001370007, Code ECS00000037), and 2) SEcurity and RIghts In the CyberSpace (PE00000014) under the NRRP Ministero dell’Università e della Ricerca program funded by the EU–NextGenerationEU.

REFERENCES

1. Y. Alexeev et al., “Quantum computer systems for scientific discovery,” *PRX Quantum*, vol. 2, no. 1, 2021, Art. no. 017001, doi: 10.1103/PRXQuantum.2.017001.
2. M. Anisetti, C. A. Ardagna, and N. Bena, “Multi-dimensional certification of modern distributed systems,” *IEEE Trans. Services Comput.*, vol. 16, no. 3, pp. 1999–2012, May/Jun. 2023, doi: 10.1109/TSC.2022.3195071.
3. M. Anisetti, C. A. Ardagna, N. Bena, V. Giandomenico, and G. Gianini, “Lightweight behavior-based malware detection,” in *Proc. MEDES*, Heraklion, Greece, May 2023.
4. M. Anisetti, C. A. Ardagna, E. Damiani, and P. G. Panero, “A methodology for non-functional property evaluation of machine learning models,” in *Proc. MEDES*, Abu Dhabi, UAE, Nov. 2020, pp. 38–45, doi: 10.1145/3415958.3433101.
5. M. Anisetti, C. A. Ardagna, E. Damiani, and G. Polegri, “Test-based security certification of composite services,” *ACM Trans. Web*, vol. 13, no. 1, 2019, Art. no. 3, doi: 10.1145/3267468.
6. C. A. Ardagna, R. Asal, E. Damiani, and Q. H. Vu, “From security to assurance in the cloud: A survey,” *ACM Comput. Surv.*, vol. 48, no. 1, 2015, Art. no. 2, doi: 10.1145/2767005.
7. “Proposal for a regulation of the European parliament and of the council laying down harmonised rules on artificial intelligence (artificial intelligence act) and amending certain union legislative acts,” European Commission, Brussels, Belgium, 2021. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52021PC0206
8. E. Damiani and C. A. Ardagna, “Certified machine-learning models,” in *Proc. Int. Conf. Current Trends Theory Pract. Inform.*, Limassol, Cyprus, Jan. 2020, pp. 3–15, doi: 10.1007/978-3-030-38919-2_1.
9. T. L. Duc, R. G. Leiva, P. Casari, and P.-O. Östberg, “Machine learning methods for reliable resource provisioning in edge-cloud computing: A survey,” *ACM Comput. Surv.*, vol. 52, no. 5, 2019, Art. no. 94, doi: 10.1145/3341145.
10. R. Faqeh et al., “Towards dynamic dependable systems through evidence-based continuous certification,” in *Proc. Int. Symp. Leveraging Appl. Formal Methods*, Rhodes, Greece, Oct. 2020, pp. 416–439, doi: 10.1007/978-3-030-61470-6_25.
11. T. Gehr, M. Mirman, D. Drachsler-Cohen, P. Tsankov, S. Chaudhuri, and M. Vechev, “AI2: Safety and robustness certification of neural networks with abstract interpretation,” in *Proc. IEEE Symp. Secur. Privacy*, San Francisco, CA, USA, May 2018, pp. 3–18, doi: 10.1109/SP.2018.00058.
12. R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi, “A survey of methods for explaining black box models,” *ACM Comput. Surv.*, vol. 51, no. 5, 2018, Art. no. 93, doi: 10.1145/3236009.

13. High-Level Expert Group on Artificial Intelligence, “Assessment list for trustworthy artificial intelligence (ALTAI) for self-assessment,” European Commission, Brussels, Belgium, 2020. [Online]. Available: https://digital-strategy.ec.europa.eu/en/library/assessment-list-trustworthy-artificial-intelligence-altai-self-assessment
14. M.-J. Montpetit and N. Crespi, “Computing in the network: The core-edge continuum in 6G network,” in *Shaping Future 6G Networks: Needs, Impacts, and Technologies*, E. Bertin, N. Crespi, and T. Magedanz, Eds. Hoboken, NJ, USA: Wiley, 2021, pp. 133–166.
15. N. Peri et al., “Deep k-NN defense against clean-label data poisoning attacks,” in *Proc. Eur. Conf. Comput. Vision*, Glasgow, U.K., Aug. 2020, pp. 55–70, doi: 10.1007/978-3-030-66415-2_4.
16. A. Shafahi et al., “Poison frogs! Targeted clean-label poisoning attacks on neural networks,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, Montréal, PQ, Canada, Dec. 2018, pp. 6106–6116, doi: 10.5555/3327345.3327509.
17. S. Sharma, J. Henderson, and J. Ghosh, “CERTIFAI: A common framework to provide explanations and analyse the fairness and robustness of black-box models,” in *Proc. AAAI/ACM Conf. AI, Ethics, Soc.*, New York, NY, USA, Feb. 2020, pp. 166–172, doi: 10.1145/3375627.3375812.
18. R. Shokri, M. Stronati, C. Song, and V. Shmatikov, “Membership inference attacks against machine learning models,” in *Proc. IEEE Symp. Secur. Privacy*, San Jose, CA, USA, May 2017, pp. 3–18, doi: 10.1109/SP.2017.41.
19. G. Vidot, C. Gabreau, I. Ober, and I. Ober, “Certification of embedded systems based on Machine Learning: A survey,” 2021, *arXiv:2106.07221*.
20. X. Wu, M. Fredrikson, S. Jha, and J. F. Naughton, “A methodology for formalizing model-inversion attacks,” in *Proc. IEEE 29th Comput. Secur. Found. Symp.*, Lisbon, Portugal, 2016, pp. 355–370, doi: 10.1109/CSF.2016.32.

MARCO ANISETTI is a full professor at the Università degli Studi di Milano, 20133, Milan, Italy, and cofounder of Moon Cloud srl. His research interests include computational intelligence and its application to the design and evaluation of complex systems. Anisetti received his Ph.D. degree in computer science from Università degli Studi di Milano. Contact him at marco.anisetti@unimi.it.

CLAUDIO A. ARDAGNA is a full professor at the Università degli Studi di Milano, 20133, Milan, Italy; the director of the Consorzio Interuniversitario Nazionale per l’Informatica National Lab on Data Science; and cofounder of Moon Cloud srl. His research interests include cloud–edge security and assurance as well as data science. Ardagna received his Ph.D. degree in computer science from Università degli Studi di Milano. Contact him at claudio.ardagna@unimi.it.

NICOLA BENA is a Ph.D. student at the Università degli Studi di Milano, 20133, Milan, Italy. His research interests include the security of modern distributed systems with particular reference to certification, assurance, and risk management techniques. Bena received his M.Sc. degree in cybersecurity from Università degli Studi di Milano. Contact him at nicola.bena@unimi.it.

ERNESTO DAMIANI is a full professor at the Università degli Studi di Milano, 20133, Milan, Italy, and founding director of the Center for Cyber-Physical Systems, Khalifa University, United Arab Emirates. His research interests include cybersecurity, big data, and cloud/edge processing. Damiani received his honorary doctorate degree for “his contribution to Big Data teaching and research” from Institute National des Sciences Appliquées de Lyon, France. Contact him at ernesto.damiani@ku.ac.ae.

Open Access funding provided by ‘Università degli Studi di Milano’ within the CRUI CARE Agreement