

**Ph.D. in Information Technology
Thesis Defense**

**October 20th, 2026
at 3:00 pm
Room 2B – building 20A**

Gabriele DIGREGORIO – XXXVIII Cycle

Titolo: A Systems Perspective on Machine Learning Security: from Malware-Detection Pipelines to Model Artifacts

Supervisor: Stefano Longari

Abstract:

Machine Learning (ML) is playing an increasingly important role in the security of modern software systems. Learned models are embedded into defensive mechanisms that analyze software artifacts and support tasks such as malware detection. At the same time, models are themselves serialized, distributed, loaded, and executed through complex software ecosystems. Yet, ML models are frequently analyzed as isolated statistical decision functions or merely as sets of weights. This thesis argues that such a view is incomplete. Once deployed, models become components of broader software systems, and their security properties depend on the pipelines, execution environments, framework mechanisms, and trust assumptions through which they operate.

The first part of this thesis investigates this perspective in the context of ML-based malware detection. We focus on dynamic detectors that classify executable programs from sequences of Application Programming Interface (API) calls observed during sandbox execution. In this setting, the attacker manipulates executable artifacts rather than abstract feature representations, and adversarial behavior must remain effective after being realized and processed by the complete analysis pipeline. We study this problem at both inference and training time. First, we introduce Tarallo, an end-to-end framework for problem-space evasion that combines adversarial sequence generation, binary modification, and sandbox re-execution. We propose a position-sensitive adversarial optimization method and strategies that explicitly account for behavioral nondeterminism. Second, we study clean-label backdooring through runtime triggers, showing how

triggers can be implemented as concrete program behavior and propagated through the dynamic-analysis pipeline. Across both settings, the results show that moving from abstract feature manipulation to the problem space introduces additional constraints and substantially increases attack difficulty, while still leaving realistic adversarial manipulation feasible.

The second part applies the same perspective to ML model artifacts themselves. We first analyze model-sharing and loading mechanisms across widely used frameworks and hubs, showing that security cannot be inferred solely from the apparent structure of a file format or from security-oriented terminology. Through manual analysis of framework loading mechanisms, we identify six previously undisclosed vulnerabilities enabling arbitrary code execution, including the first officially recognized CVEs affecting Keras's `safe_mode`. A user study further shows that security-oriented names, flags, and platform safeguards influence perceived trust, exposing a gap between expected and actual guarantees. Building on these findings, we propose MOAT, a lifecycle-aware approach for securing model execution through dynamic analysis of interactions with the host system, and RE-MOAT, its syscall-based reference implementation. Evaluated on 31 attack PoCs, 77,974 real-world model artifacts, and 334 models from a state-of-the-art dataset, RE-MOAT detects all malicious behaviors considered under the adopted threat model with no observed false positives on the evaluated datasets.

Across both parts, the thesis shows that ML shifts the relevant unit of security analysis from the isolated model to the integrated software system in which it participates. When ML is used defensively, the surrounding pipeline shapes which attacks are realizable and which guarantees persist after integration. When models are shared and deployed, their artifacts become security-relevant components of the surrounding system whose risks emerge from the mechanisms that interpret and execute them. More broadly, the thesis argues that ML systems should be evaluated in the environments for which they are designed. Model-level analysis remains necessary, but it is insufficient when security properties depend on interactions with concrete artifacts, preprocessing stages, execution environments, frameworks, and other system components. Evaluating an ML model in isolation is therefore analogous to testing a software component without assessing its behavior once integrated into the complete system. In this sense, ML models are not just sets of weights, but components of software systems.

PhD Committee

Prof. Alessandro Barengi, Politecnico di Milano

Prof. Emilio Coppa, Libera Università Internazionale degli Studi Sociali

Prof. Marco Squarcina, TU Wien