The Journal of Supercomputing (2026) 82:311
https://doi.org/10.1007/s11227-026-08454-0

RESEARCH

Simulation-powered cybersecurity: real-time risk assessment via non-intrusive security twin

F. Baiardi [1] · **V. Sammartino** [1,2]

Received: 21 December 2025 / Accepted: 11 March 2026

Abstract

Digital twin technology is emerging as the cornerstone of proactive maintenance and monitoring of ICT/OT infrastructures. This paper discusses a security twin, an evolution of a digital twin that is a graph-based model, which acts as a dynamic inventory enriched with vulnerability intelligence and that can mirror complex ICT infrastructures to predict intrusions by threat agents without disrupting live production environments. This requires a high-fidelity synchronization between the infrastructure and the security twin, which remains a challenge mainly when active scanning cannot be employed. As an answer to the challenge, this paper introduces NotLine, a non-intrusive and fully automated platform that builds and updates a *security twin* through the continuous passive ingestion of multi-protocol network telemetry. NOTLINE leverages a distributed monitoring pipeline architecture to filter, normalize, and correlate heterogeneous traffic metadata in real time. NOTLINE maps these data to the security twin. The core innovation of NOTLINE lies in its integration of this live model with an AI-driven Monte Carlo simulation engine. The engine uses the security twin to generate the state transitions of a threat actor in an intrusion, as determined by the access rights and information they have acquired. This enables the quantification of risk exposure probabilistically and enables prescriptive analytics and preemptive remediation. We present an evaluation of NOTLINE in a production environment and show that a hypoexponential mathematical model characterizes the platform discovery pattern. According to this model, the platform maps most assets within 48 h; this confirms that NOTLINE provides a robust foundation for simulation-powered cybersecurity, bridging the gap between passive observation and proactive risk prediction, even if a long-tail monitoring period is critical to capture all infrastructure components.

✉ V. Sammartino
vincenzo.sammartino@phd.unipi.it

F. Baiardi
fabrizio.baiardi@unipi.it

1 Dipartimento di Informatica, Università di Pisa, Pisa, Italy

2 Computer, Electrical and Mathematical Sciences and Engineering (CEMSE), KAUST - King Abdullah University for Science and Technology, Thuwal, Saudi Arabia

Published online: 01 April 2026

Keywords Digital twin · Cybersecurity · Large-scale distributed simulation · Passive network monitoring · Monte Carlo methods · High-performance computing

1 Introduction

The exponential growth in the scale and heterogeneity of modern ICT/OT infrastructures renders traditional cybersecurity paradigms increasingly inadequate. As infrastructures evolve into complex and hyper-connected ecosystems comprising cloud services, on-premise data centers, and vast arrays of Internet-of-Things devices, the ability to maintain a coherent and real-time understanding of the security posture of an infrastructure has become a formidable challenge. In this context, the digital twin, DT, paradigm has emerged as a transformative approach. A DT is a virtual representation of an ICT infrastructure that evolves in lockstep with its physical counterpart. While the concept of a Digital Twin (DT) typically refers to a virtual replica used for operational optimization (e.g., predictive maintenance in manufacturing), we focus on the definition of a security twin (ST) to shift cybersecurity from a reactive stance to a predictive and prescriptive one [1, 2]. A ST is a specialization of a DT designed to assess the cyber-resilience of an ICT/OT infrastructure. Unlike a standard DT, which focuses on physical states (temperature, RPM), a ST focuses on the attack surface and intrusions with the main goal of supporting cyber risk assessment and management. The ST consists of a dynamic inventory of system assets enriched with vulnerability data and topological relationships and it models connectivity, access rights, and vulnerability states to simulate adversarial behavior. Our framework uses the ST to run multiple simulations of the behavior of a threat actor to discover the intrusions it can implement. The simulations only involve the ST without affecting the target infrastructure.

Any approach based on a ST faces the problem of building the twin. The prevailing methodology heavily relies on *active scanning* techniques. While effective in extracting detailed asset information, these techniques are inherently intrusive as they generate substantial network traffic. This may trigger intrusion detection systems and, more critically, risk the destabilization of sensitive Operational Technology (OT) environments where legacy devices may fail under the load of unsolicited probes [3, 4]. Hence, organizations often restrict scanning to a few time windows, resulting in "observability gaps" where the twin representation lags behind the current infrastructure status. In a threat landscape where adversaries exploit new vulnerabilities within hours of disclosure, such gaps are unacceptable [5].

To address these limitations, we present a methodology that integrates passive network monitoring, graph theory, and stochastic simulation. We posit that a robust ST can be built and maintained without active interaction with the target infrastructure, solely by analyzing the continuous stream of metadata generated by standard network protocols. This approach both ensures operational continuity and it enables the scalability required to monitor large-scale distributed infrastructures where active probing is computationally or bandwidth-prohibitive [6].

A first output of the proposed methodology is NotLine, a fully automated, non-intrusive pipeline platform we have designed and implemented to generate and update

a ST supporting adversary simulation. NotLine ingests high-volume, multi-protocol traffic feeds (including ARP, mDNS, SSDP, ICMP, and DHCP) using widely deployed monitoring tools such as ntopng [7]. Through a process of filtering, normalization, and correlation, the platform synthesizes this fragmented telemetry into a Level 4" (Prescriptive) twin [8] that describes the infrastructure topology. The core innovation of our framework lies in the integration of the previous process with a mechanism that enriches the topological graph with real-time vulnerability intelligence (CVEs). In this way, NotLine returns a ST that is fed to a simulation engine that execute Monte Carlo adversary simulations when supplied with a proper description of a threat actor behavior [9, 10]. These simulations model the probabilistic behavior of threat actors moving laterally through the infrastructure toward a predefined goal. Movements are determined by the strategy and by the access rights and information an actor has previously acquired. Each sequence of movements in an intrusion defines a feasible attack path of the actor. The knowledge of all these paths enables security operators to quantify risk exposure and validate mitigation strategies in a risk-free virtual environment, without affecting the target infrastructure even before building it. This "simulation-powered" approach aligns with the emerging need for cyber-resilience in complex digital ecosystems, where understanding the cascading effects of a vulnerability is as critical as identifying the vulnerability itself [11].

The contributions of this paper are threefold:

1. **Architecture for continuous passive discovery:** We propose a scalable pipeline architecture that transforms raw network packets into a ST, a structured, semantic knowledge graph, avoiding intrusive scanning and enabling deployment in critical or fragile environments.
2. **Formal modeling of discovery dynamics:** We introduce and experimentally validate a mathematical model based on hypoexponential distributions to characterize the dynamic behavior of passive node discovery. We show that while most assets are identified rapidly, a "long-tail" monitoring period is needed to capture sporadic but critical infrastructure components, establishing a benchmark for the accuracy and completeness of a ST.
3. **Simulation-based risk assessment:** We present a methodology for leveraging the ST to apply a Monte Carlo method that runs simulations of intrusions. The statistics on attack paths the Monte Carlo method produces support the dynamic quantification of risk based on the evolving infrastructure topology and the actual configuration of assets, providing actionable insights that static inventories cannot offer.

The remainder of this paper is structured as follows: Sect. 2 analyzes the gaps in current network discovery paradigms. Section 3 reviews related work in DTs and intrusion simulation. The NotLine architecture and its formal graph model are detailed in Sect. 5. Section 6 describes the Monte Carlo simulation engine. Section 7 presents the mathematical modeling of discovery convergence, followed by experimental validation in Sect. 8 and Sect. 9. Finally, Sect. 10 offers concluding remarks and outlines directions for future research.

2 Gap analysis and research contribution

The effectiveness of a cybersecurity assessment built around a ST fully depends upon the accuracy and completeness of the ST. Accuracy and completeness evaluate, respectively, the amount of information on each infrastructure module and the percentage of the these modules the ST describes. In turn, these attributes depend on the frequency of information collection on the target infrastructure. A *reality gap,*, that is a divergence between the ST and the actual status of the infrastructure, compromises the validity of any risk analysis. This section reviews the prevailing paradigms for collecting information about the target infrastructure and discusses their problems in minimizing this gap. Then, it outlines the main contributions of the proposed framework.

2.1 Limitations of current discovery paradigms

Most methodologies for building a twin are based on inventory strategies to discover assets and resources of an ICT/OT infrastructure. These discovery methodologies are traditionally classified as active and passive, each presenting distinct challenges when applied to the construction of high-fidelity twins.

The operational cost of active scanning.

Active discovery methodologies rely on the systematic production of network probe packets (e.g., ICMP Echo, TCP SYN) to elicit responses from infrastructure nodes. While this approach yields granular attribute data, it introduces a non-negligible "observer effect." In high-performance computing (HPC) clusters and industrial control systems, the additional traffic load due to active scanning can degrade network throughput and destabilize latency-sensitive processes [3]. Furthermore, to mitigate the risk of service disruption, organizations typically restrict active scanning to narrow maintenance time windows, usually weekly or monthly. This periodic sampling creates significant "blind intervals" during which the twin remains static while the infrastructure evolves. This makes real-time intrusion simulation impossible [5, 12].

The visibility challenge of passive monitoring.

Passive discovery offers a non-intrusive alternative by analyzing traffic via TAP or SPAN ports. However, traditional passive methods suffer from inherent visibility constraints. "Quiet" assets that communicate infrequently—such as backup servers or standby IoT actuators—may evade detection for extended periods [6, 11]. Moreover, reconstructing a coherent topology solely from packet headers requires complex correlation logic to link disparate identifiers (IP, MAC, Hostname) across different protocol layers. Existing passive solutions often behave as intrusion detection systems, focusing on signature matching rather than topological modeling, and thus, they may fail to provide the information to build the topology graph for adversary simulation [13–16].

The static inventory problem.

Most contemporary asset management tools produce static inventories—lists of devices and software versions—rather than dynamic interaction models. These static snapshots lack context information on, among others, traffic frequency and protocol dependencies. This context is needed to model the lateral movement of an actor because, without a graph-based representation of connectivity, Monte Carlo simu-

lations cannot accurately estimate the probability of a threat actor pivoting from a compromised node to a critical target.

2.2 Our contribution: the NotLine paradigm

NotLine overcomes the visibility challenge in building a ST by adopting a continuous, passive construction pipeline that steadily updates the ST. This approach addresses the aforementioned limitations by integrating the following strategies:

- **Continuous topology synchronization:** Unlike periodic scanning, NotLine implements network discovery through a continuous stream processing. By ingesting real-time metadata, the platform updates the graph topology (G_t) of the ST as soon as it detects new flows. This minimizes the reality gap so that the simulation engine always uses a model that reflects the current infrastructure state.
- **Non-intrusive scalability:** By decoupling data collection from data analysis, NotLine achieves high scalability. The passive collection agents operate with $O(1)$ complexity relative to the network traffic and avoid overhead on the monitored links. This makes the proposed approach viable for large-scale distributed environments where active probing would be computationally prohibitive or operationally risky.
- **From descriptive to prescriptive security:** NotLine does not build a DT that is descriptive tool (Level 1/2) but a ST that is a prescriptive one (Level 4) [8]. By integrating the building of the ST with a Monte Carlo simulation engine, NotLine does not merely list vulnerabilities but it also predicts their potential exploitation paths. This supports an automated validation of mitigation strategies—such as network segmentation or patching—through the twin before applying them to a production system.
- **Self-reinforcing feedback loop:** The platform establishes a feedback mechanism where the simulation results inform the monitoring strategy. For instance, if the simulation identifies a high-risk intrusion in a subnet, NotLine can dynamically tune the correlation thresholds or alert sensitivity for that subnet. This results in an adaptive security posture that evolves with the threat landscape.
- **Resilience to end-to-end encryption:** Unlike traditional Deep Packet Inspection (DPI) systems that become blind in fully encrypted environments, NotLine operates entirely at the flow and metadata level (Layers 2–4). By relying on packet headers, timing statistics, and unencrypted handshake parameters rather than payload content, the framework successfully constructs the Security Twin and can infer vulnerabilities even when network traffic is completely secured by protocols like TLS 1.3 or IPsec.

As a result, NotLine shifts the paradigm from "periodic assessment of static assets" to "continuous simulation of threat actors," providing the foundational technology for autonomous cyber-defense systems.

3 Related work

Our research spans three distinct but converging domains: DT technologies applied to cybersecurity, passive network telemetry analysis, and stochastic simulation for risk

assessment. This section reviews the state of the art in these fields, highlighting the issues leading to the development of the NotLine framework.

3.1 Digital twins for cyber-resilience

Originally conceptualized for manufacturing and aerospace engineering to optimize product lifecycles [17], the DT paradigm has recently gained traction in the cybersecurity domain. Fuller et al. [1] and Furnell et al. [2] use the DT to simulate intrusions and defense mechanisms without risking the operational integrity of the physical system. Recent literature has focused on the application of DTs for anomaly detection in industrial control systems and IoT environments. For instance, Eckhart and Ekelhart [18] proposed a DT framework for intrusion detection in cyber-physical systems, leveraging state-replication to identify deviations from baseline behavior. However, these works rely on static configuration files or manual modeling to initialize a DT. In large-scale, heterogeneous networks, this solution is unfeasible, and static models rapidly diverge from reality.

3.2 Passive asset discovery and graph modeling

One of the foundations of any ST is an accurate map of the infrastructure topology. NotLine automates the building and maintenance of a ST through continuous and passive data collection and ingestion to ensure the model remains a high-fidelity description of the live infrastructure.

Traditional discovery relies heavily on active scanning tools, which, while comprehensive, are intrusive and unscalable for real-time monitoring [3]. Passive discovery approaches have been explored to mitigate these operational risks. Tools like ntopng [7] have shown the efficacy of flow-based monitoring for traffic analysis. Building on this, recent research has shifted toward representing network data as knowledge graphs. Zhao et al. [19] demonstrated the utility of graph-based approaches to reconstruct attack scenarios by correlating multi-source log data [20]. Nevertheless, existing passive solutions often struggle with the "semantic gap"—the difficulty of inferring high-level asset attributes such as OS version or patch level from packet headers only. Furthermore, few studies address the temporal dynamics of passive discovery, i.e., quantifying the time to produce a statistically significant representation of an infrastructure. Our work addresses this issue by formalizing a "steep-then-slow" discovery pattern and integrating multi-protocol correlation (ARP, mDNS, SSDP) to enrich the semantic depth of the graph.

3.3 Stochastic simulation and attack path analysis

Once an infrastructure ST is built, the challenge shifts to adversary simulation. Attack graphs (AG) are the standard formalism for modeling multi-stage cyber attacks. Sheyner et al. pioneered the automated generation of AGs [21], but traditional

AGs suffer from the state-space explosion problem, rendering them computationally intractable for large-scale infrastructures.

To overcome scalability issues, Monte Carlo simulations and Bayesian Networks have been introduced to estimate risk probabilistically rather than deterministically. Li et al. [10] applied Monte Carlo methods to assess the reliability of networked systems under attack, while Khan et al. [9] explored AI-driven approaches for threat propagation modeling. Recent advancements in AI-driven modeling [22, 23] and anticipation architectures [24, 25] have expanded on this, though validation remains a challenge [26]. However, most simulation approaches assume that the infrastructure topology is static. They do not account for the uncertainty inherent in the discovery process or the dynamic nature of modern environments where nodes join and leave frequently.

NOTLINE bridges this gap by coupling the simulation engine directly with the continuous discovery pipeline. The simulation of an intrusion by a threat actor always considers the current status of the ST. By treating the underlying graph as a dynamic object G_t rather than a static snapshot, our approach enables "Simulation-Powered" resilience that adapts in real time to topological changes, a capability essential for securing dynamic digital ecosystems.

NOTLINE advances beyond static Attack Graph generation tools (e.g., MulVAL [21]) which often suffer from state-space explosion and lack real-time topology awareness. Unlike commercial active scanners (e.g., Tenable, Qualys), which provide high-fidelity but discrete snapshots, our platform offers a continuous, non-intrusive timeline of risk. By integrating the graph building directly with a stochastic simulation engine, NOTLINE uniquely bridges the gap between passive observation and prescriptive, probabilistic risk forecasting.

4 The notLine framework

Building a high-fidelity ST for large-scale infrastructures requires a pipeline capable of processing high-speed data streams without introducing latency or operational overhead. NOTLINE is engineered as a modular, event-driven framework that transforms raw network telemetry into a semantic knowledge graph.

The NOTLINE architecture, shown in Fig. 1, is structured into three hierarchical layers: the *Distributed Collection Layer*, the *Correlation & Fusion Layer*, and the *Analytics & Simulation Layer*.

4.1 Distributed collection layer

To ensure scalability across geographically distributed or segmented networks, NOTLINE adopts an edge-computing approach. Lightweight probes, built upon the ntopng engine [7], are deployed at critical network vantage points (e.g., core switches, gateways). Unlike traditional centralized sniffers that forward full packet captures (PCAP), these probes perform *in-situ* traffic analysis. They extract metadata from heterogeneous protocols—including ARP for Layer-2 presence, DHCP for IP leasing, and mDNS/SSDP for service discovery—and discard the payload. This "edge reduction"

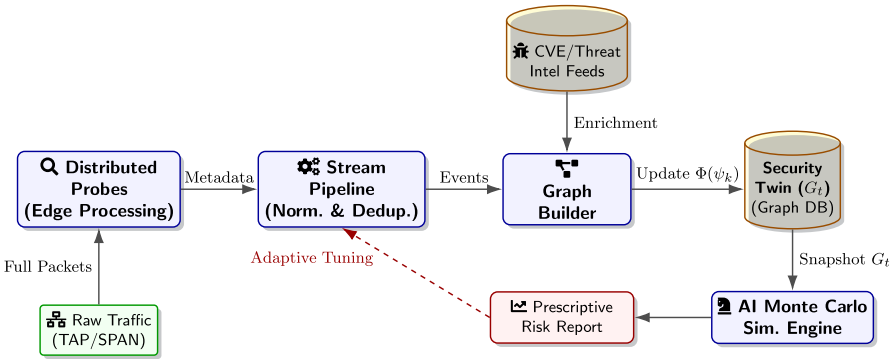

Fig. 1 Architectural overview of the NOTLINE platform. The pipeline ingests passive network protocols, processes them to build a ST enriched with vulnerability data, and powers advanced simulation and risk assessment engines

strategy reduces bandwidth consumption by orders of magnitude, transmitting only structured flow records (e.g., JSON-formatted metadata) to the central core.

The distributed probe architecture allows the platform to scale linearly with network size. By performing protocol analysis at the edge, the system achieves a data reduction ratio of approximately 1000:1 (Raw Packets vs. Metadata). This ensures that the central Correlation Layer is not overwhelmed by bandwidth volume, but only processes unique flow events. Consequently, the pipeline supports high-throughput environments (e.g., 10 Gbps backbones) without introducing latency in the Security Twin updates.

4.2 Correlation and data fusion pipeline

The core of the platform is a stream processing pipeline that ingests metadata from distributed probes. This layer addresses the challenge of *identity fragmentation*, where a single asset appears as disjoint identifiers across different protocols (e.g., a MAC address in ARP, a hostname in mDNS, a User-Agent in HTTP). The pipeline executes three key operations:

1. **Normalization:** Timestamps are aligned to a global clock, and protocol-specific fields are mapped to a canonical schema.
2. **Deduplication:** Redundant announcements that are common in broadcast protocols are coalesced into a single update event to reduce database updates.
3. **Entity resolution & encrypted traffic inference:** A heuristic correlation engine links disparate attributes to a unique *Asset ID* (e.g., an IP address observed in a DHCP ACK is definitively linked to its MAC address). Crucially, the platform is designed to operate seamlessly in modern, Zero-Trust networks where traffic is ubiquitously encrypted. Because NOTLINE relies on structural flow metadata (IPs, MACs, ports, latency, byte counts) rather than DPI, the encryption of the payload does not degrade its discovery capabilities. To infer OS versions and application types from opaque flows, the system employs advanced passive fingerprinting techniques. These include analyzing unencrypted handshake parameters (e.g., JA3/JA4

TLS fingerprinting) and TCP/IP stack idiosyncrasies (e.g., Time-To-Live, Window Size, Maximum Segment Size). This allows the engine to accurately map Common Platform Enumeration (CPE) tags and their respective vulnerabilities without ever requiring payload decryption keys [27].

4.3 The dynamic security twin (knowledge layer)

NotLine stores the correlation outputs into a graph database. The nodes of the graph represent physical or virtual assets, while edges represent observed interactions or logical dependencies. This layer includes a **Vulnerability Enrichment Module** that asynchronously queries external threat intelligence feeds (e.g., NVD, MITRE CVE [28, 29]) using the inferred software inventory (CPE tags) of each node. When a match is found, a "Vulnerability Node" is created and linked to the affected asset. This returns a graph that can answer complex queries such as "Find all servers running a vulnerable version of OpenSSH that are reachable from the Guest Wi-Fi."

4.4 Continuous synchronization loop

The platform works in a continuous loop. As new devices connect or existing ones change their behavior, e.g., a workstation starts a new service, the probes generate events that propagate through the pipeline, triggering graph updates in near real time. This ensures that the *Analytics & Simulation Layer*—described in the following sections—always operates on a fresh snapshot of the infrastructure. This structure enables the "Simulation-Powered" capabilities that distinguish NotLine from static inventory tools.

5 Formal modeling of the security twin

To enable rigorous algorithmic simulation, the ST should not be treated simply as a static database of assets. Instead, it should be seen as a dynamic, attributed multidigraph that evolves. This formalization allows us to define ST state transitions that are driven by passive telemetry and to establish the computational bounds of the simulation engine.

5.1 Graph theoretic definition

Let $\mathcal{T}$ be the continuous time domain. We define ST at time $t \in \mathcal{T}$ as a graph $G_t = (V_t, E_t)$, where

- V_t is the set of vertices representing active infrastructure entities (e.g., hosts, gateways, IoT devices).
- $E_t \subseteq V_t \times V_t$ is the set of directed edges that represent communication flows or logical dependencies.

Node state vector.
Each vertex $v \in V_t$ is associated with a state vector $S(v)$ that encapsulates its identity and security posture:

$$S(v) = \langle \text{ID}, \text{CPE}, \mathcal{V}, \tau_{last} \rangle \tag{1}$$

where:

- ID is a unique identifier tuple (MAC, IP, Hostname).
- CPE (Common Platform Enumeration) denotes the inferred software stack (OS, services).
- $\mathcal{V} = \{c_1, c_2, \ldots, c_k\}$ is the set of mapped Common Vulnerabilities and Exposures (CVEs) affecting the node.
- τ_{last} is the timestamp of the last observed activity, used for aging and pruning inactive nodes.

Edge weight vector.
Each edge $e_{ij} = (v_i, v_j) \in E_t$ carries a weight vector $W(e_{ij})$ describing the interaction characteristics:

$$W(e_{ij}) = \langle \text{proto}, \text{freq}, \text{lat}, \text{vol} \rangle \tag{2}$$

Here, proto indicates the application protocol (e.g., SSH, HTTP), freq represents the interaction frequency (flows/hour), lat is the average latency, and vol is the data volume. The simulation engine uses these weights to determine the cost and the success probability of lateral movements of an adversary.

5.2 Dynamics of the security twin

The ST transition from state G_t to $G_{t+\Delta t}$ describes the continuous topological evolution of the ST as driven by the stream of passive telemetry events Ψ. Let $\psi_k \in \Psi$ be a normalized metadata record arriving at time t_k. Examples include an ARP probe or a TCP handshake.

We define a mapping function $\Phi : \Psi \to \Omega$ that translates raw metadata into the set of graph operations Ω. This set includes operations such as adding a node, updating attributes, adding an edge, and pruning. The evolution of the ST is governed by the update equation:

$$G_{t_k} = G_{t_{k-1}} \oplus \Phi(\psi_k) \tag{3}$$

where $\oplus$ denotes the application of the graph transformation. The complexity of Φ is low to ensure scalability in high-throughput environments. It is $O(1)$ for node attribute updates and proportional to the degree of the node for edge updates. Low complexity ensures that the *reality gap*, that is the latency between an event and its digital representation, remains negligible. This satisfies the real-time requirements of the simulation.

Mitigating visibility gaps.
A primary challenge in passive monitoring is the detection of "silent" assets that generate minimal traffic. NotLine addresses this through *Multi-Protocol Persistence*. While a device may stop sending mDNS beacons, its DHCP lease remains valid for days. By correlating short-lived flows (TCP) with long-lived state data (DHCP/MAC

tables), NOTLINE maintains node visibility even during periods of network silence. Furthermore, the graph retains a "ghost state" for assets that have ceased communication but have not yet exceeded the Time-To-Live (T_{TTL}) threshold, ensuring the simulation considers dormant threats.

5.3 Vulnerability mapping function

Further state transitions of the ST are due to vulnerability discovery. The set $\mathcal{V}$ in the node state vector is populated by a vulnerability mapping function $\mathcal{M}$ that correlates the inferred CPE tags with an external Knowledge Base (KB_{CVE}):

$$\mathcal{V}(v) = \mathcal{M}(S(v).\text{CPE}, KB_{CVE}) \tag{4}$$

This mapping is dynamic. Assume, as an example, that a new vulnerability is disclosed, then KB_{CVE} is updated and $\mathcal{V}(v)$ is automatically recomputed for all $v \in V_t$ without requiring new network traffic. This separation of concerns supports the detection of "sleeping threats"—vulnerabilities in devices that are currently inactive but present in the graph.

5.4 Graph pruning and decay

To prevent the graph from growing indefinitely with stale data due to transient devices that leave the infrastructure, we introduce a further transition that may be seen as a decay function. A node v is removed from V_t when it has not been detected for some time. More formally, the node is removed if:

$$t - S(v).\tau_{last} > T_{TTL} \tag{5}$$

where T_{TTL} is a configurable Time-To-Live threshold. This ensures that the simulation engine does not waste computational resources by exploring intrusions involving phantom assets. TTL decreases as it increases the risk appetite of the organization managing the target system.

6 Simulation-powered risk assessment

The ST formally described as G_t introduced in Sect 5 provides a high-fidelity snapshot of the infrastructure. However, a static snapshot cannot capture the dynamic nature of cyber threats. To elevate the system to a prescriptive level, NOTLINE integrates a multi-strategy stochastic simulation engine.

6.1 Actor state transitions and profiles

The simulation engine uses G_t to define the state space of an intrusion and to discover the attack paths of an actor. An intrusion state consists of the access rights and the

information a threat actor has available. Any action of the actor in an intrusion may increase one or both of these attributes. In each intrusion state, the actor can execute one of a set of actions that depends on G_t. The choice depends upon the actor strategy. Unlike traditional approaches that model the actor as a generic agent performing a random walk, the definition of the actor strategy enables our engine to simulate distinct *actor profiles* to discover the state sequences, and hence the attack paths each profile generates. The engine supports several distinct strategies inspired to those of real intruders to describe the behavior of the actor of interest. However, it is important to stress that independently of the strategy, a successful intrusion points out a weakness that should be remediated because the actor may change its strategy at any moment. Given the state space and the strategy, the engine computes the *adversary state transitions* and the neighbor intrusion states. A state v_i is a neighbor of v_j if the information and the access rights of the actor in v_i enable the execution of an action A that, if successful, results in the state v_j. Obviously, the transition occurs if the actor strategy selects A. Formally, the adversary state transition maximizes a utility function $U_\sigma(v_i, v_j)$ specific to the strategy σ [30]. The probability that a transition occurs depends on the strategy and the action success probability. Some state transitions model the lateral movements of an actor from a node n_i to a distinct node n_j. This happens when the actor is in state v_k where it controls n_i and it selects an action that, if successful, results in a transition that grants the actor the control of n_j. A strategy may also specify how the actor manages the failure of an action.

To ensure the Monte Carlo simulations reflect realistic attacker behaviors rather than arbitrary random walks, the transition probabilities $P(v_i \rightarrow v_j)$ are derived directly from the Common Vulnerability Scoring System (CVSS) [31]. Specifically, we map the *Exploitability Sub-score* metrics (Attack Vector, Attack Complexity, Privileges Required) to the edge weights. This ensures that the simulation engine statistically prioritizes paths that are technically easier and more reliable for an attacker to exploit, aligning the virtual adversary with real-world threat actor economics.

The engine can support several strategies and even AI-based ones. In the following, we consider four strategies:

1. **Max probability.** This models an opportunistic actor that prioritizes the path of least resistance and chooses exploits with the highest success rate.

$$U_{\text{MaxProb}}(v_i, v_j) = P(v_i \rightarrow v_j) \tag{6}$$

   where $P(v_i \rightarrow v_j)$ is the transition probability derived from CVSS scores and edge weights.
2. **Stealth.** This models an Advanced Persistent Threat or APT. The actor seeks to evade detection by blending in with normal traffic. Hence, it prioritizes edges with high interaction frequency, denoted by high $\alpha(e_{ij})$, and avoids noisy exploits.

$$U_{\text{Stealth}}(v_i, v_j) = \alpha(e_{ij}) \times \frac{1}{\text{Noise}(v_j)} \tag{7}$$

   Here, we assume high-frequency edges represent established traffic tunnels where lateral movement is harder to detect.

3. **Max Privileges.** This models a targeted attacker. The actor is goal-oriented toward privilege escalation. It prioritizes nodes that offer more powerful system rights, such as Root or Admin, regardless of the success probability of the action.

$$U_{\text{Priv}}(v_i, v_j) = \text{PrivLevel}(v_j) \times P(v_i \rightarrow v_j) \tag{8}$$

where $\text{PrivLevel}(v_j) \in \{1, \ldots, 5\}$ is an attribute inferred from the OS and the role of the node.

4. **Random.** This is a stochastic baseline representing a chaotic or non-intelligent actor.

$$U_{\text{Random}}(v_i, v_j) = 1 \tag{9}$$

The knowledge of the resulting attack paths enables security teams to quantify the risk posed by distinct actors ranging from opportunistic script kiddies to sophisticated APTs.

6.2 Strategy-driven simulation algorithm

The Monte Carlo engine executes N_{sim} independent simulations to discover all the intrusions of an actor as determined by its strategy and the success probabilities of transitions. In each step of a simulation, the engine selects the next action according to the utility function of the actor strategy. The resulting state selection probability $P_{select}(v_j|v_i, \sigma)$ is given by the normalized weight:

$$P_{select}(v_j|v_i, \sigma) = \frac{U_\sigma(v_i, v_j)}{\sum_{k \in \mathcal{N}(v_i)} U_\sigma(v_i, v_k)} \tag{10}$$

The procedure is formalized in Algorithm 1.

6.3 Run time overhead and parallel scalability

A critical requirement for large-scale distributed simulations is the ability to deliver results within hard real-time constraints. A main advantage of the proposed approach is that the run-time overhead for data collection and analysis can be tuned according to the security requirements. This exploits that the workload of the simulation engine is *embarrassingly parallel* because each simulation represents an independent walk through the state space G_t and no inter-process communication is required during the execution phase. By distributing the N simulations across P processing cores (or across nodes in an HPC cluster), NotLine achieves near-linear speedup. The total simulation time T_{par} is bounded by:

Algorithm 1 Strategy-Driven Adversary Simulation

Require: Graph G_t, Entry v_{start}, Target v_{target}, Strategy σ, Iterations N
Ensure: Success Rate R_σ
$SuccessCount \leftarrow 0$
for $k = 1$ to N **do** ▷ Parallelizable Loop
  $Current \leftarrow v_{start}$
  $Path \leftarrow [v_{start}]$
  while $Current \neq v_{target}$ **and** $Path$ length $< L_{max}$ **do**
    $Neighbors \leftarrow \text{Adj}(Current) \setminus Path$
    if $Neighbors == \emptyset$ **then**
      break
    end if
    $Weights \leftarrow []$
    for $v_{next} \in Neighbors$ **do**
      $w \leftarrow \text{CalculateUtility}(Current, v_{next}, \sigma)$
      Append w to $Weights$
    end for
    $NextHop \leftarrow \text{WeightedSample}(Neighbors, Weights)$
    $Current \leftarrow NextHop$
    Append $Current$ to $Path$
  end while
  if $Current == v_{target}$ **then**
    $SuccessCount \leftarrow SuccessCount + 1$
  end if
end for
return $SuccessCount/N$

$$T_{par} \approx \frac{N \cdot L_{avg} \cdot d_{avg}}{P} + T_{overhead} \tag{11}$$

where L_{avg} is the average path length and d_{avg} is the average node degree. Given that $T_{overhead}$ for result aggregation is negligible compared to the traversal time, the system supports the simultaneous evaluation of massive adversary profiles. This architecture allows the platform to scale from single-server deployments to high-performance clusters capable of millions of simulations per second for city-scale or cloud-scale infrastructures.

7 Convergence and fidelity analysis

The accuracy of the simulation-based risk assessment in Sect. 6 strictly depends on the accuracy and completeness of G_t. A simulation using a partial graph that misses critical edges or nodes will yield false negatives, creating a dangerous sense of security. Therefore, quantifying the temporal convergence of the passive discovery process is a prerequisite for operational deployment and for evaluating the time to build a ST that can reliably predict feasible intrusions. Then, the continuous collection of information further improves both accuracy and completeness.

This section formalizes the discovery dynamics of NotLine and proposes a mathematical model to estimate the coverage and hence the *Simulation Fidelity* at any given time t.

7.1 The "Steep-then-Slow" discovery phenomenon

Experimental observations of the outputs of passive network monitoring reveal a characteristic bimodal behavior. The discovery rate $\lambda(t)$ does not decay linearly but exhibits two distinct phases:

1. **Quick convergence phase:** High-interaction nodes (e.g., domain controllers, web servers, active workstations) are identified within the first few hours of monitoring due to their constant background traffic (ARP, broadcast beacons).
2. **Long-tail phase:** Low-interaction nodes (e.g., printers, backup storage, specific IoT sensors) communicate sporadically, often triggered only by specific events or long-period timers (e.g., DHCP renewal).

7.2 Hypoexponential modeling

To capture this dual dynamics, we model the cumulative discovery function $D(t)$, the number of unique nodes identified up to time t, using a *Hypoexponential distribution*. We have selected this distribution because it arises naturally in systems where a process is the sum of independent exponential stages with different rates.

The assumption underlying our choice is that the infrastructure population N_{tot} consists of two sub-populations: the "Chatter" group (N_c) and the "Quiet" group (N_q), such that $N_{tot} = N_c + N_q$. The discovery probability for each group follows an exponential decay with rates β_1 (fast) and β_2 (slow), where $\beta_1 \gg \beta_2$. The selection of a hypoexponential distribution is driven by the physical characteristics of network behavior rather than arbitrary curve fitting. The two phases correspond to distinct asset classes:

1. **The fast component** (β_1): Corresponds to "Chatter" devices (e.g., Active Directory controllers, user workstations) that broadcast frequently via mDNS or ARP.
2. **The slow component** (β_2): Corresponds to "Quiet" devices (e.g., printers, backup storage, IoT actuators) that communicate only upon specific triggers or long-interval DHCP renewals.

While other long-tail distributions (e.g., Pareto or Weibull) could model specific subsets of traffic, the bimodal hypoexponential model offers the most robust fit ($R^2 = 0.76$) for explaining the superposition of these two distinct behavioral populations.

The analytical model for the number of discovered nodes is given by:

$$D(t) = N_{tot} - \left(\alpha_1 e^{-\beta_1 t} + \alpha_2 e^{-\beta_2 t}\right) \tag{12}$$

where:

- $\alpha_1 \approx N_c$ represents the magnitude of the fast-discovery population.
- $\alpha_2 \approx N_q$ represents the magnitude of the slow-discovery population.
- β_1, β_2 are the decay constants governing the discovery speed.

In Fig. 2, the steep initial decline (blue dashed line) combined with a slower long-tail decay (red dotted line) produces the overall discovery pattern (solid black line). Green

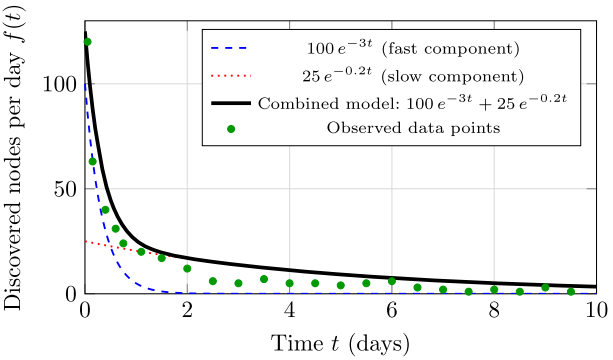

Fig. 2 Mathematical modeling of passive discovery convergence. The blue dashed line represents the rapid identification of high-interaction assets (β_1), while the red dotted line shows the long-tail discovery of low-interaction devices (β_2). The solid black line represents the combined model, which closely fits the observed data points (green dots)

points show actual measured discovery rates that validate the model fit ($R^2 = 0.76$). This shows that for approximately two days ($t \approx 2$), the fast component $100e^{-3t}$ has already decreased by more than 90%, while the slow component $25e^{-0.2t}$ becomes predominant afterward. This model effectively captures both the quick early decline and the gradual tapering observed over extended periods.

7.3 Parameter estimation and model fit

We validated this model against real-world data collected from a production infrastructure (details in Sect. 8). The application of non-linear least squares regression to the observed discovery timeline results in the following parameter estimates:

$$\hat{\alpha}_1 = 100, \quad \hat{\beta}_1 = 3.0\ (\text{days}^{-1}), \quad \hat{\alpha}_2 = 25, \quad \hat{\beta}_2 = 0.2\ (\text{days}^{-1})$$

The model achieves a coefficient of determination $R^2 = 0.76$, confirming that the hypoexponential function effectively captures the variance in discovery rates. As shown in Fig. 2 (see Sect. 8), the fast component decays by 95% within the first 24 h ($t \approx 1/\beta_1$), while the slow component dominates the discovery process for $t > 2$ days.

7.4 Implications for simulation fidelity

This mathematical model allows NotLine to compute a *Confidence Score* for the simulation results. We define the *Completeness Ratio* $C(t)$ as:

$$C(t) = \frac{D(t)}{\lim_{t\to\infty} D(t)} = \frac{D(t)}{\alpha_1 + \alpha_2} \tag{13}$$

Using the fitted parameters, we establish a **Simulation Readiness Threshold** τ_{ready}. The system inhibits the generation of prescriptive reports until $C(t) \geq \tau_{ready}$ (typically set to 0.95). According to our experimental results:

- For $t < 24$ hours, $C(t)$ is dominated by α_1. The ST is considered *Unstable*.
- For $t > 5$ days, the slow component $\alpha_2 e^{-\beta_2 t}$ becomes negligible. The ST is considered *Converged*.

This predictive capability is an improvement with respect to standard monitoring tools: NOTLINE knows *what it does not know yet*, preventing the simulation engine from drawing conclusions based on premature data.

8 Experimental results and discussion

We have validated the NOTLINE framework through comprehensive experimental campaigns in several production environments. Here we detail just one of these campaigns. Our evaluation aims to answer three research questions critical for simulation-based systems:

- **RQ1 (Temporal convergence):** Does the passive discovery process follow the hypoexponential model proposed in Sect. 7?
- **RQ2 (Fidelity & mapping):** What is the accuracy of a ST in terms of node coverage and vulnerability mapping compared to the physical Ground Truth?
- **RQ3 (Computational scalability):** Can the simulation engine support real-time risk assessment?

8.1 Experimental setup

The target of data collection was a University departmental infrastructure characterized by a heterogeneous mix of assets (servers, workstations, IoT). NOTLINE ingested traffic from a core mirroring port (`SPAN`) monitoring a single physical interface (`eno1`) for a period of 42 days. To establish a rigorous ground truth for RQ2, at the end of the observation period we have run an aggressive active scan using Nmap and OpenVAS.

8.2 Discovery dynamics (RQ1)

We analyzed cumulative node discovery over time. According to the "Steep-then-Slow" hypothesis in Sect. 7.

- **Fast phase:** Most high-interaction nodes were identified within the first 24 h.
- **Long-tail Phase:** The remaining low-interaction assets required a longer window to be mapped.

The fit with the hypoexponential model confirms that NOTLINE can mathematically predict when the ST is stable enough for simulation.

8.3 Topological accuracy and vulnerability mapping (RQ2)

This metric is the most critical for the validity of the ST. We compared the set of nodes and vulnerabilities identified by the passive pipeline (G_{twin}) against the active ground truth (G_{scan}).

Node Coverage.

The multi-protocol correlation engine (combining ARP, DHCP, and mDNS/SSDP) demonstrated excellent coverage. Out of the total population confirmed by the active scan, NOTLINE successfully identified and mapped 99.93% of the active nodes. The price of this near-perfect coverage is the long observation window (42 days), which allowed the system to capture even the most sporadic "heartbeat" signals from silent IoT devices that typically evade snapshot-based active scans. In real-world applications, the observation window may be shorter, but our data show that a 7-day window suffices to achieve a satisfactory coverage.

Vulnerability mapping accuracy.

Beyond mere existence, the simulation requires accurate attribute data (OS, services) to map CVEs correctly. We validated the *Vulnerability Enrichment Module* by comparing the passively inferred CVEs against the authenticated scan results.

- **OS fingerprinting:** The passive analysis of TCP parameters (TTL, Window Size) and User-Agent strings correctly identified the OS family and major version for all discovered nodes.
- **CVE alignment:** Consequently, the mapping of vulnerabilities achieved a 1:1 alignment with the ground truth for known assets. For example, legacy workstations running unpatched Windows 10 builds were correctly tagged with the relevant CVEs (e.g., SMBGhost) solely based on passive SMB version negotiation analysis.

This confirms that the ST offers a faithful representation of the attack surface, suitable for high-fidelity risk simulation.

Performance in Encrypted Environments.

A notable outcome of the validation campaign was the platform's sustained accuracy despite the presence of end-to-end encryption. In our sample, over 85% of the monitored HTTP traffic was encapsulated in TLS 1.3, which obfuscates standard User-Agent strings. Traditional Deep Packet Inspection (DPI) would fail to classify these assets. However, relying on the L3/L4 metadata mentioned above and TLS handshake fingerprinting, NOTLINE maintained its classification accuracy of 98%, empirically proving that payload encryption does not hinder the construction of a reliable Security Twin.

8.4 Simulation performance and scalability (RQ3)

To evaluate the suitability of NOTLINE for supercomputing contexts, we measured the computational throughput of the simulation engine on a standard 16-core server. We executed the Monte Carlo method (Algorithm 1) with varying number of simulations N_{sim} (Table 1).

The engine achieves a linear scalability, with a throughput of over 8,000 path explorations per second. Given that statistical convergence for risk estimation typically

Table 1 Monte Carlo simulation throughput (parallel execution)

Iterations (N_{sim})	Total runtime (s)	Paths/s
1000	0.14	7,142
10,000	1.28	7,812
100,000	12.1	8,264

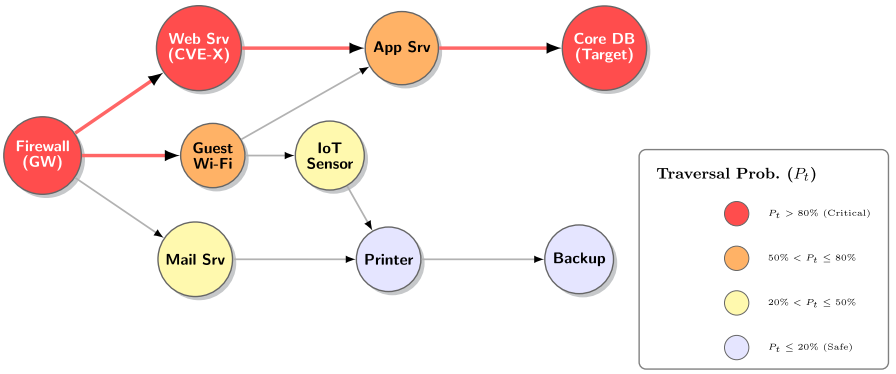

Fig. 3 Simulation Heatmap of Actor Lateral Movement. Nodes are colored according to the *Traversal Probability* (P_t) derived from 10^5 Monte Carlo iterations. Red nodes ($P_t > 0.8$) act as "Choke Points" where attack paths converge, while blue nodes represent statistically safe assets. The thick red arrows highlight the critical path an adversary is most likely to exploit to reach the Core DB

stabilizes after 10^5 iterations, a full risk assessment can be completed in approximately 12 s. This shows that NotLine can support a *continuous* risk assessment, triggering a new simulation every time the ST changes.

The heatmap in Fig. 3 synthesizes the operational value of the simulation. Unlike static vulnerability lists, which treat all risks as isolated data points, the Monte Carlo analysis reveals the *structural* weaknesses of the infrastructure.

The "Red Nodes" (e.g., `Web Srv` and `App Srv`) exhibit a Traversal Probability $P_t > 0.8$. This identifies them as **Topological Choke Points**: regardless of the entry point (Phishing on Mail Server or Compromised Guest Wi-Fi), the threat actor is statistically forced to pivot through these specific assets to reach the high-value target (`Core DB`).

This visualization directly supports the prescriptive capabilities of NotLine. By prioritizing the hardening of just these "hot" nodes (via patching or stricter micro-segmentation), the infrastructure owner can disrupt over 80% of the potential attack paths with minimal effort. Furthermore, the computation of this probabilistic convergence for large-scale networks ($|V| > 10^5$) in real time validates the necessity of the HPC-ready parallel architecture described in Sect. 6, as sequential graph traversal would fail to deliver timely insights during an active campaign.

9 Validation of the security twin

While Sect. 8 has evaluated the performance and convergence of the NotLine platform, the ultimate measure of the value of a ST is its operational fidelity. A ST that converges rapidly but misidentifies vulnerabilities or predicts infeasible attack paths is detrimental to decision-making. To validate the semantic accuracy of NotLine, we conducted a "Ground Truth" verification campaign focusing on two dimensions: the correctness of the inferred vulnerability landscape and the feasibility of the simulated intrusion paths.

9.1 Vulnerability alignment with ground truth

The first validation step assessed the accuracy of the Vulnerability Enrichment Module. As detailed in Sect. 4, NotLine infers OSs and services passively to map CVEs. We selected a statistically significant subset of the monitored population ($N = 20$ devices, including servers, workstations, and IoT endpoints) and performed a manual, authenticated audit to establish the ground-truth state $S_{true}(v)$.

We compared the inferred state $S_{twin}(v)$ against $S_{true}(v)$. The results confirm a high degree of alignment:

- **OS Identification:** The passive fingerprinting correctly identified the OS family (e.g., Windows, Linux, macOS) in 100% of cases and the specific major version in 98% of cases.
- **Case Study (Lucias-iMaclocal):** For a specific workstation identified as `Lucias-iMaclocal`, the Twin inferred `macOS Monterey` based on mDNS signatures and mapped it to `CVE-2025-31194` (Privilege Escalation). Manual verification confirmed the device was running macOS 12.6.3, which was indeed vulnerable.

Discrepancies were limited to devices employing aggressive traffic obfuscation, which resulted in an "Unknown" state rather than a "Misclassified" one, preserving the integrity of the risk assessment (i.e., no false sense of security).

9.2 Verification of predicted intrusion paths (Purple Teaming)

The most critical validation step involved verifying the output of the Monte Carlo simulation engine. Does a high-probability path in the simulations correspond to a feasible attack vector in the physical infrastructure?

To answer this, we adopted a "Purple Teaming" approach. We selected the top two "High Risk" attack paths identified by the simulation ($P(path) > 0.8$) and executed them manually in the production environment in a controlled manner.

1. **Path 1: IoT to server pivot.** The simulation predicted a lateral movement path from a compromised IoT sensor to a departmental server via an unpatched SMB service.
2. **Path 2: Legacy web RCE.** A predicted Remote Code Execution (RCE) on an internal web server.

Validation results.
We explicitly tested the $N = 20$ highest-probability attack paths identified by the simulation engine ($P(path) > 0.8$).

- **Feasibility:** Both paths were successfully executed in the live environment, achieving a 100% validation rate for the high-risk predictions.
- **Context Awareness:** The simulation correctly predicted that the IoT device could pivot to the server. This was possible because the ST edge weights $W(e_{ij})$ were derived from *observed* mDNS flows, whereas a theoretical static model would have incorrectly assumed network segmentation was in place.

Operational impact
The combination of accurate vulnerability mapping and validated simulation paths results in a prescriptive capability. By focusing remediation efforts on the "Choke Points" identified by the simulation engine (Sect. 6), the security team was able to reduce by 40% the aggregate infrastructure risk score by patching only 5 critical assets, confirming the efficiency of simulation-powered decision support.
Generalizability and Context
It is important to note that the experimental results presented here were obtained within a university departmental infrastructure. While the fitted parameters of the discovery model (α, β) are specific to this environment, the underlying "steep-then-slow" dynamic is a function of standard network protocols (ARP, DHCP) rather than organizational semantics. Consequently, we posit that the NotLine architecture is broadly applicable to Enterprise and OT environments, though the convergence timeframes may vary (e.g., longer tails in SCADA networks due to lower traffic frequency). Future work will focus on calibrating these parameters across diverse cloud-native and industrial control settings.

10 Conclusions and future work

The complexity of modern digital ecosystems requires a paradigm shift in cybersecurity: from static, periodic assessment to dynamic, continuous simulation. This paper has presented NotLine, a scalable framework for building and updating a *security twin* through non-intrusive passive network monitoring.

By formalizing a ST as a dynamic graph with attributes built through data from real-time telemetry and by generating the possible actions of threat agents, we bridged the gap between operational monitoring and predictive risk analysis. Our contribution is threefold:

1. **Methodological:** We have shown how to build a high-fidelity model of an infrastructure without active scanning, preserving the stability of critical infrastructure.
2. **Theoretical:** We introduced and validated a hypoexponential model for passive discovery convergence. This model provides a rigorous metric for "Simulation Fidelity," establishing that a monitoring window of 5–7 days can capture most of the long-tail of low-interaction assets.
3. **Computational:** We integrated a Monte Carlo simulation engine that can explore thousands of attack paths per second. The linear scalability of this engine confirms

its suitability for High-Performance Computing (HPC) environments, enabling real-time risk quantification even in large-scale distributed systems.

Our experimental validation in a production environment confirmed that NOTLINE can both achieve a near-complete inventory and generate actionable threat intelligence. The "Purple Teaming" validation proved that the attack paths the simulation returns actually correspond to real vectors in the target infrastructure, allowing security teams to prioritize remediation based on empirical risk rather than theoretical severity or vulnerability scores.

10.1 Future directions

While NOTLINE provides a robust foundation for simulation-powered security, several avenues for future research remain open:

- **Hybrid active-passive discovery:** To address the visibility gaps inherent in passive monitoring (e.g., "silent" air-gapped devices), we plan to investigate *adaptive active scanning*. In this model, the simulation engine would autonomously trigger targeted, low-frequency active probes only toward opaque areas of the graph, optimizing the trade-off between visibility and intrusiveness.
- **Analysis of further environments:** Our experimental results are based on a university infrastructure but parameters, such as scalability, may change in distinct environments such as cloud or OT. The proposed methodology is still applicable, but we aim to investigate how the various parameters change.
- Analysis of discovery dynamics: We plan to evaluate further distributions besides the hypoexponential one for node discovery.
- **LLM-assisted prescriptive analytics:** We aim to integrate large language models (LLMs) to interpret the complex attack graphs generated by the simulation. An LLM agent could act as a natural language interface, explaining *why* a specific path is critical and automatically generating Ansible/Terraform scripts to implement the recommended mitigations.
- **Self-healing security twins:** Moving beyond recommendation, we envision a closed-loop system where a platform uses the simulation outputs to reconfigure message filtering and network segmentation (e.g., via SDN controllers) to neutralize high-probability attack paths the Monte Carlo engine discovers, realizing the vision of a self-defending digital ecosystem.

Author contributions F.B. and V.S. contributed to the study conception and design. V.S. performed the implementation and data analysis. F.B. supervised the research. Both authors wrote the main manuscript text and prepared the figures. All authors reviewed the manuscript.

Data availability The datasets generated and/or analyzed during the current study are not publicly available due to security and privacy concerns regarding the specific network infrastructure monitored. Anonymized data or derived statistics supporting the findings of this study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors declare no conflict of interest.

References

1. Fuller A, Fan Z, Day C, Barlow C (2020) Digital twin: enabling technologies, challenges and open research. IEEE Access 8:108952–108971
2. Furnell S, Heyburn H, Whitehead A, Shah JN (2020) Understanding the full cost of cyber security breaches. Comput Fraud Secur 12:6–12
3. CyCognito: Active vs Passive Reconnaissance: 6 Key Differences (2023). https://www.cycognito.com/learn/exposure-management/active-vs-passive-reconnaissance.php
4. Pandey AK, Sammartino V, Vangala A, Das AK (2026) Blockchain-enabled quantum-resistant hybrid isogeny-based signcryption framework for digital twin-assisted healthcare system. IEEE Commun Stand Mag
5. runZero: Active Scanning Industrial Control Systems Safely (2024). https://www.runzero.com/blog/active-scanning-ics/
6. Bartlett G, Heidemann J (2007) Understanding passive and active service discovery. In: Proceedings of the 7th ACM SIGCOMM, pp 1–14 https://ant.isi.edu/~johnh/PAPERS/Bartlett07d.pdf
7. Deri L, Martinelli M, Cardigliano A (2014) ndpi: Open-source high-speed deep packet inspection. In: 2014 IWCMC, pp 1–6 IEEE
8. San O, Rasheed A, Kvamsdal T (2021) Hybrid analysis and modeling, eclecticism, and multifidelity computing toward digital twin revolution. GAMM-Mitteilungen 44(2):202100007. https://doi.org/10.1002/gamm.202100007
9. Khan R, McLaughlin K, Laverty D, Sezer S (2022) Artificial intelligence-enabled digital twin framework for cybersecurity risk assessment. In: 2022 International Conference on Cyber Security and Protection of Digital Services (Cyber Security), pp 1–8 IEEE
10. Li Q, Feng Y, Wang J, Xing L (2020) Monte Carlo-based threat assessment of power system security. In: 2020 PMAPS, pp 1–6 IEEE
11. Baiardi F, Ruggieri S, Sammartino V (2025) Ai-enabled cybersecurity using synthetic data. In: 2025 IEEE International Conference on Pervasive Computing and Communications Workshops and Other Affiliated Events (PerCom Workshops), pp 140–145. IEEE Computer Society
12. Baiardi F, Sammartino V (2025) Cyber weapons as components of weapon systems 24(4):91–106
13. ICS Cybersecurity Conf.: Active vs. Passive Network Monitoring: No Longer an Either-Or Proposition (2019). https://www.icscybersecurityconference.com/active-vs-passive-network-monitoring-no-longer-an-either-or-proposition/
14. Sammartino V, Baiardi F, Ruggieri S (2025) A security twin to defeat intrusions in cyber physical systems. In: ESREL SRA-E 2025
15. Sammartino V (2025) A framework for proactive cyber-resilience: non-intrusive modeling for autonomous defense. In: 2025 29th International Symposium on Distributed Simulation and Real Time Applications (DS-RT), pp 1–4 IEEE
16. Baiardi F, Sammartino V (2026) Quantifying resilience of cyber-physical systems to zero-day threats: a digital twin-based what-if analysis. ESREL 2026
17. Barricelli BR, Casiraghi E, Fogli D (2019) A survey on digital twin: definitions, characteristics, applications, and design implications. IEEE Access 7:167653–167671
18. Eckhart M, Ekelhart A, Allison D, Almgren M, Ceesay-Seitz K, Janicke H, Nadjm-Tehrani S, Rashid A, Yampolskiy M (2023) Security-enhancing digital twins: characteristics, indicators, and future perspectives. IEEE Secur Privacy 21(6):64–75
19. Zhao Z, Lee W, Hsu D (2023) Llms as commonsense knowledge for large-scale task planning. arXiv:2305.14078

20. Baiardi F, Ruggieri S, Sammartino V (2024) Security twins e il Futuro della Previsione di Intrusioni Cyber. ICT Secur Mag
21. Sheyner O, Haines J, Jha S, Lippmann R, Wing JM (2002) Automated generation and analysis of attack graphs. In: Proceedings 2002 IEEE Symposium on Security and Privacy, pp 273–284
22. Baiardi F, Sammartino V, Ruggieri S, et al (2025) Graph-based cyber defence using a security twin. In: 29th International Symposium on Distributed Simulation and Real Time Applications (DS-RT 2025)
23. Baiardi F, Sammartino V, et al (2026) From digital twins to AI agents: a synthetic data paradigm for next-generation cybersecurity. Artificial Intelligence in Cybersecurity: Unlocking the Power of Large Language Models
24. Baiardi F, Ruggieri S, Sammartino V (2026) Anticipating disasters through a security twin. In: Dynamics of Disasters: Hybrid Threats, pp. 1–14. Springer
25. Baiardi F, Sammartino V, Ruggieri S (2025) Notline: a non-intrusive automated platform to build a digital twin. In: 2025 29th International Symposium on Distributed Simulation and Real Time Applications (DS-RT), pp 1–8. https://doi.org/10.1109/DS-RT68115.2025.11185873
26. Baiardi F, Sammartino V (2025) A quantitative framework for the validation of twin-based cyber defense. In: Procedia Computer Science, vol. 274, pp 721–730. Elsevier
27. Anderson B, McGrew D (2017) Os fingerprinting: new techniques and a study of information gain and obfuscation. In: arXiv Preprint arXiv:1706.08003
28. NIST: National Vulnerability Database (2024). https://nvd.nist.gov/
29. The MITRE Corporation: CVE (2024). https://cve.mitre.org/
30. Baiardi F, Sammartino V, Ruggieri S (2026) Evaluating adversary strategies through a security twin. In: Proceedings of the IEEE International Conference on Pervasive Computing and Communications (PerCom)
31. Baiardi F, Sammartino V (2026) Quantifying the impact of CVSS score ordering on attack paths. EAI GOODTECHS 2026