IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511164

NetTwin: An AI-Driven Self-Adapting Digital Twin for Cyber Defense

Aswin B[1], Aswani A[2], Darshan Saj[3], Shammas Khan S[4], Assoc Prof. Shani Raj[5], Asst Prof. Ajeesh S[6]

B.Tech Student, Department of Computer Science and Engineering, College of Engineering Kottarakkara, Kottarakkara, Kerala, India[1-4]

Project Guide, Department of Computer Science and Engineering, College of Engineering Kottarakkara, Kottarakkara, Kerala, India[5,6]

Abstract: Threshold-based network monitoring struggles with distributed and low-rate attacks, and it offers no safe place to test a defensive action before it is applied to a live service. This paper presents NetTwin, a proposed telemetry-synchronized, behavior-aware and self-adapting digital twin for cyber defense. NetTwin observes traffic and server telemetry in an isolated laboratory, extracts statistical and Shannon-entropy features, combines them with behavioral indicators, and classifies traffic sources as legitimate, suspicious or uncertain. A digital-twin engine keeps a representation of the monitored environment, predicts the effect of candidate defenses such as rate limiting, throttling, filtering and temporary blocking, and compares each prediction with the outcome later observed in the lab. Verified outcomes are stored in a knowledge engine that accepts or rejects updates to the decision logic. The paper describes the architecture, the digital-twin formulation, the planned DoS and DDoS experiments, and the evaluation metrics, and it states clearly which parts are design and which are implemented. No experimental results are claimed here.

Keywords: Digital Twin, Cyber Defense, DDoS Detection, Shannon Entropy, Adaptive Defense, Network Telemetry, Traffic Classification

I. INTRODUCTION

Denial-of-Service (DoS) and Distributed Denial-of-Service (DDoS) attacks remain a practical threat to network services. A rule such as “block any address that sends more than 1,000 packets” is easy to implement, but it fails in several common situations. A distributed attack can keep every source below the limit while their combined load overwhelms the target. A legitimate user can generate heavy traffic. Many legitimate users can share one public address behind NAT, and dynamic addressing makes attribution unreliable. A wrong blocking decision therefore causes both a missed attack and a loss of service for real users.

A second weakness is that monitoring tools observe and alert, but they rarely give operators a safe environment in which a candidate response can be evaluated before it is applied. A digital twin, which is a digital representation of a physical system that is kept up to date with observations from that system [1], is a natural candidate for this role. The idea has been studied in power systems, industrial control, 5G networks and IoT, but, as Section II shows, the existing work is mostly tied to its own domain.

This paper proposes NetTwin, a digital twin for the cyber defense of a server environment. A virtual machine is not a digital twin by itself, and NetTwin is not meant to be a dashboard or a conventional intrusion detection system. The aim is a closed loop of the form Observe → Understand → Predict → Evaluate → Defend → Learn → Adapt, in which the twin is synchronized with telemetry, candidate defenses are tested on the twin first, predictions are compared with measured outcomes, and verified feedback changes later decisions. The research question is stated as follows:

How can a telemetry-synchronized and behavior-aware digital twin analyze network traffic, identify potentially malicious behavior, evaluate candidate defense strategies, and adapt future decisions using verified experimental feedback?

The contributions of this work are: (i) an architecture that links packet-level features, entropy and behavioral indicators, three-class source classification, a digital-twin engine, defense evaluation and a feedback-driven knowledge engine; (ii)

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511164

a precise statement of what the twin represents, what is synchronized and how predictions are validated; and (iii) an experimental plan with explicit metrics for an isolated, reproducible laboratory.

II. RELATED WORK

A survey of twenty entries was prepared for this project. Because one entry duplicates another, nineteen distinct studies are discussed here, and they fall into five groups (Table I).

TABLE I GROUPED SUMMARY OF THE LITERATURE REVIEWED

Group	Representative studies	Gap relevant to NetTwin
Twin-based attack detection in energy systems	DC microgrids with local and global twins using RBF networks [2]; power systems with multidimensional telemetry [10]; hydroelectric plant with explainable AI [14]; power-grid twin with edge computing [16]	Domain-specific signals and assumptions; transfer to IP traffic is not shown.
Twin-based detection in industrial and critical infrastructure	Industrial control intrusion detection [11]; real-time industrial monitoring [18]; neural-network threat detection for critical infrastructure [19]; DoS-resilient distributed dispatch [9]	Control-system focus; source-level classification of network traffic is not the central goal.
Twin-based DDoS detection in networks	DDoS detection in the metaverse [8]; intelligent DDoS detection for autonomous core networks [15]; cyber twin for IoT [20]	Context-specific (metaverse, core network, IoT); defense evaluation on the twin is limited.
Network twin design and simulation	Twins for 5G [6]; AI-driven 5G/6G network twin [17]; simulation tools for cyber-threat modeling [13]	Management-oriented or survey-level; no synchronized defensive twin for a small server lab.
Surveys and predictive defense	Moving target defense with AI [3]; twin security threats and requirements [5]; enabling technologies and open problems [7]; threat-oriented twinning [4]; attack prediction with hierarchical reinforcement learning [12]	Surveys give no end-to-end evaluation; learning-based methods are costly for a lightweight lab.

Three observations follow from this review. First, most twin-based security work addresses a specific physical domain, so evidence for general server traffic is thin. Second, DDoS-oriented twins concentrate on detection, whereas the comparison of several mitigation actions against a predicted outcome is less developed. Third, the studies pay little attention to how verified feedback should update the stored knowledge, and to how wrong or outdated feedback is rejected. NetTwin targets these three points in a small, isolated and reproducible environment.

Entropy [23] is a long-standing tool for traffic analysis. Feinstein et al. used entropy of source addresses for DDoS detection [21], and Lakhina et al. showed that distributions of traffic features expose anomalies that volume counts miss [22]. Entropy is used here as one indicator among several and not as a stand-alone detector.

III. PROBLEM STATEMENT AND OBJECTIVES

Threshold rules do not capture network context, and monitoring alone cannot evaluate a response before it is applied. The objectives of NetTwin are to: (1) build a controlled virtual network for experiments; (2) collect packet data and server telemetry; (3) extract statistical and behavioral features; (4) study entropy of traffic distributions; (5) detect abnormal behavior from several indicators; (6) classify sources as legitimate, suspicious or uncertain; (7) keep a digital representation synchronized with selected observations; (8) evaluate candidate mitigations inside the twin; (9) compare predicted and actual outcomes; (10) store observations and verified feedback and use them to improve later decisions; (11) provide a dashboard; and (12) evaluate the prototype with controlled, repeatable experiments.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511164

IV. PROPOSED SYSTEM ARCHITECTURE

NetTwin is organized as the pipeline in Fig. 1. The upper row turns raw traffic into a classified view of the environment. The lower row evaluates a response on the twin, checks it against what actually happened, and returns the result to the analysis stage.

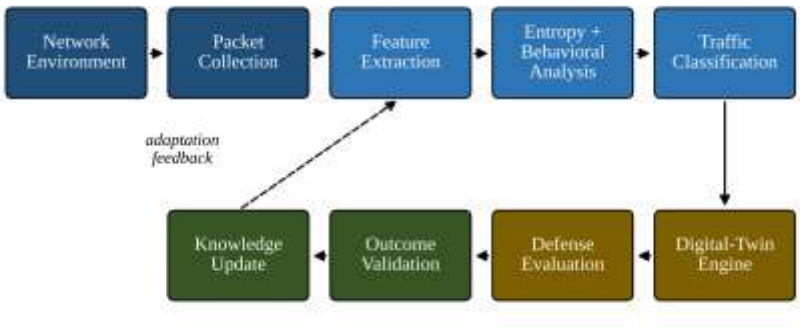


Fig. 1 NetTwin processing pipeline with the adaptation feedback loop

A. Network Environment and Packet Collection

The environment is an isolated laboratory on a Windows host running Oracle VirtualBox, with an Ubuntu Server 24.04 LTS virtual machine as the target, a legitimate client process and controlled simulated traffic sources. The host was earlier planned with about 8 GB of RAM, so the design stays light: separate processes or containers are preferred over one virtual machine per module, and the hardware and VirtualBox network modes are verified before any capture is configured. Packets are captured with tcpdump or Wireshark and parsed with Scapy or PyShark. Each packet record holds the timestamp, source and destination address, source and destination port, protocol, length and TCP flags. Offline processing of PCAP files comes first, and live capture is added later if feasible.

B. Feature Extraction

Packet records are turned into the features in Table II. Packet rate and request rate are treated as different quantities. Empty captures, missing fields, invalid values and zero-duration windows are handled explicitly so that a rate is never computed by dividing by zero.

TABLE II TRAFFIC FEATURES EXTRACTED BY NETTWIN

Feature	Meaning
Packet count / total bytes	Packets and bytes observed in a window
Unique source IPs	Distinct source addresses observed
Packets and bytes per second	Rate over a defined interval
Packets and bytes per IP	Average or per-source traffic volume
Connection frequency	Number of observed connection attempts
Protocol and port distribution	Share of TCP, UDP, ICMP and of destination ports
TCP flag distribution	Frequency of selected flags (for example SYN)
Connection duration	Duration of observed connections

C. Entropy Analysis

For a window of traffic, the probability p(x) of each value x (a source address, destination port or protocol) is estimated from observed counts, and the Shannon entropy is

$$H(X) = -\Sigma\, p(x)\, \log_2 p(x) \quad (1)$$

The entropy is also normalized by its maximum so that windows with a different number of distinct values can be compared:

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511164

$H_n(X) = H(X) / \log_2 N$(2)

where N is the number of distinct values in the window. Source-IP, destination-port and protocol entropy are computed. Entropy is interpreted in context: high entropy does not prove an attack, and low entropy does not prove legitimacy. It is therefore combined with the behavioral indicators below.

D. Behavioral Analysis

Behavior is followed over time instead of against one fixed threshold. The indicators are a sudden rise in packet rate or bytes per second, a rise in distinct sources, repeated connection attempts, an abnormal TCP flag distribution, concentration on one destination port, deviation from a historical baseline, distributed patterns and low-rate activity spread over many sources. A baseline is learned from normal traffic in the laboratory, and deviation from it is measured per window.

E. Traffic Classification

Each source is placed in one of three classes. Legitimate means the traffic is consistent with expected behavior and there is not enough suspicious evidence. Suspicious means several indicators agree on abnormal or potentially malicious activity. Uncertain means the evidence is not sufficient for a reliable decision, so more observation is needed. The uncertain class is deliberate: a source is never called malicious only because its packet count is high, because shared and NAT addresses, dynamic addresses, spoofing, short observation time and distributed attacks all make a single count unreliable. The first implementation uses transparent rules over the indicators so that each decision can be explained. Machine learning is added only if it gives a measurable benefit.

F. Digital-Twin Engine

The twin is the central component. It is defined by six elements.

1) Counterpart:

The isolated server and network environment in the laboratory.

2) Representation:

Server state, traffic statistics, active sources, protocol and port distributions, connection behavior, CPU and memory use, historical observations, current classifications and candidate defense configurations.

3) Synchronization:

Packet observations, traffic rates, per-source statistics, protocol distributions, CPU and memory utilization and server availability are passed to the twin at a fixed interval. The interval is a design parameter that will be reported with the results.

4) Analysis and prediction:

For a candidate action the twin estimates the effect on suspicious traffic, legitimate request success, availability and latency. The action is evaluated on the twin and does not touch the real environment.

5) Validation:

After the same action is applied in the laboratory, the observed outcome is compared with the prediction. For a metric m, the prediction error is

$e_m = | m_predicted - m_observed |$(3)

6) Adaptation:

Verified errors feed the knowledge engine described below.

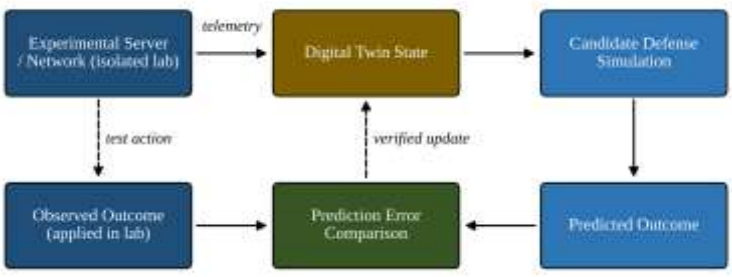


Fig. 2 Synchronization and validation loop of the digital twin



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.1511164

G. Defense Evaluation

Six candidate actions are compared: monitor only, generate an alert, rate limit, throttle, filter selected traffic, and temporarily block a simulated suspicious source. The criteria are the reduction in suspicious traffic, legitimate request success rate, service availability, response latency, resource use, false-positive impact and the gap between predicted and observed outcome. Protection of legitimate users has priority: an action that lowers suspicious traffic but blocks legitimate users is not counted as an unconditional success.

H. Adaptive Knowledge Engine

The engine stores feature records, classifications, context, decisions, predicted and actual outcomes, verified labels, false-positive events and baselines. Storing data does not by itself make a system self-adapting, so the update rule is part of the design. A feedback record is used only if its label is verified. Stale records are down-weighted so that changes in traffic patterns (concept drift) do not keep old behavior in force. A proposed update to the baseline or to the decision logic is accepted only if it does not worsen the prediction error or the false-positive rate on held-out earlier experiments; otherwise it is rejected and logged.

I. Dashboard and Backend

The backend is a modular Python application, with Flask or FastAPI if an API is needed, and CSV or JSON files in early prototypes and SQLite for persistent experiment records. The dashboard shows server status, packet and byte rates, unique sources, protocol and port distribution, entropy values, source classification, alerts, candidate actions, twin state, predicted and actual outcomes and experiment history. Live measurements, predicted values, user-entered values and mock values are labeled separately. The dashboard is a visualization layer and not the research contribution.

V. EXPERIMENTAL DESIGN

A. Scenarios and Safety

The planned scenarios are normal traffic, high-volume legitimate traffic, DoS, DDoS, TCP flood, UDP flood, SYN flood, abnormal connection behavior, distributed low-rate traffic and mixed legitimate and suspicious traffic. All traffic is generated inside the isolated, authorized laboratory. No attack traffic is sent to public systems, production infrastructure, college networks or any unrelated device.

B. Parameters

The initial planning values are an aggregate simulated load of 200 requests per second, of which 20 requests per second are legitimate, from several simulated sources, with several defense configurations. These are planned settings and not findings. For every run the configured target rate, generated rate, observed rate, server processing rate and dropped or rejected traffic are measured and reported separately. Each experiment has a baseline run, a recorded duration and repeated trials.

C. Metrics

Precision, recall, detection rate, false-positive rate, false-negative rate and detection latency are computed for classification, only where ground-truth labels exist. Legitimate request success rate, response latency, resource use, traffic reduction and defense effectiveness are computed for defenses. The prediction error of the twin (Eq. 3) is reported for each metric. Where ground truth is unavailable the corresponding metric is not computed.

VI. IMPLEMENTATION STATUS AND LIMITATIONS

NetTwin is developed in stages: requirements and environment; packet processing on synthetic records; real packet collection from the laboratory; the analysis engine; the twin; the adaptive knowledge engine; and finally the backend and dashboard. At the time of writing, this paper reports the architecture, the formulation and the evaluation plan. No experimental results are presented, and no claim of detection accuracy or defense effectiveness is made. The conclusions of Section VII will be supported only after the staged experiments are completed.

Known limitations of the design are the small scale of the laboratory, the use of simulated traffic instead of production traces, the dependence of the twin on the quality of its synchronized telemetry, and the risk that incorrect feedback labels corrupt the knowledge engine. The update-rejection rule in Section IV-H is intended to limit the last risk, and it is itself part of what the experiments must test.

 This work is licensed under a Creative Commons Attribution 4.0 International License



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.1511164

VII. CONCLUSION AND FUTURE WORK

This paper presented NetTwin, a proposed digital twin for cyber defense that combines packet features, entropy, behavioral indicators, three-class source classification, defense evaluation on a synchronized twin, and feedback-driven adaptation. The design addresses gaps seen in the literature: limited transfer of domain-specific twins to server traffic, little comparison of mitigations against predicted outcomes, and little treatment of how verified feedback should update a system. Future work is to complete the staged implementation, run the planned DoS and DDoS experiments, report the measured classification, defense and prediction-error results, and test the adaptation mechanism under changing traffic patterns.

ACKNOWLEDGMENT

The authors thank the Department of Computer Science and Engineering, College of Engineering Kottarakkara, for the facilities and guidance provided for this project.

REFERENCES

[1]M. Grieves and J. Vickers, “Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems,” in *Transdisciplinary Perspectives on Complex Systems*. Springer, 2017, pp. 85–113.
[2]Y. Lu, M. Zhang, L. Nordström and Q. Xu, “Digital twin-based cyber-attack detection and mitigation for DC microgrids,” 2025.
[3]T. Zhang et al., “Moving target defense meets artificial-intelligence-driven network: A comprehensive survey,” 2025.
[4]“Threat-oriented digital twinning for security evaluation of autonomous platforms.”
[5]“A comprehensive survey on digital twin: Focusing on security threats and requirements.”
[6]“Digital twins for 5G networks: A modeling and deployment methodology.”
[7]“Digital twin: Enabling technologies, challenges and open research.”
[8]“DDoS attack detection through digital twin technique in metaverse.”
[9]“Resilient consensus-based distributed dispatch over a network using digital-twin backup system under multi-channel DoS attacks.”
[10]“Digital twin-based cyber threat detection in digital power systems with multidimensional telemetry.”
[11]“Digital twin-based intrusion detection for industrial control systems.”
[12]“When moving target defense meets attack prediction in digital twins: A convolutional and hierarchical reinforcement learning approach.”
[13]“A survey of network simulation tools and digital twin technology for cyber threat modeling in smart infrastructure.”
[14]“Enhancing cybersecurity of a hydroelectric power plant through digital twin modeling and explainable AI.”
[15]“Digital twin-enabled intelligent DDoS detection mechanism for autonomous core networks.”
[16]“Research on security management and control of power grid digital twin based on edge computing.”
[17]“Design of an AI-driven network digital twin for advanced 5G-6G network management.”
[18]“Digital twin-based real-time industrial monitoring and control.”
[19]“Critical information infrastructures intelligent protection: Digital twins and neural network-based threat detection methods.”
[20]“Cyber twin technology for AI-driven real-time software security in IoT ecosystems.”
[21]L. Feinstein, D. Schnackenberg, R. Balupari and D. Kindred, “Statistical approaches to DDoS attack detection and response,” in *Proc. DARPA Information Survivability Conf. and Exposition (DISCEX)*, 2003.
[22]A. Lakhina, M. Crovella and C. Diot, “Mining anomalies using traffic feature distributions,” *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 4, pp. 217–228, 2005.
[23]C. E. Shannon, “A mathematical theory of communication,” *Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

 This work is licensed under a Creative Commons Attribution 4.0 International License