Journal of Scientific & Industrial Research
Vol. 84, November 2025, pp. 1216-1229
DOI: 10.56042/jsir.v84i11.21727

Bioinspired Trust Score Weight Optimization for an Improved Zero Trust Architecture

Ashutosh Soni*, Surendra Kumar Nanda & Ganapati Panda

Department of Computer Science and Engineering, C. V. Raman Global University, Bhubaneshwar, 752 054, Odisha, India

Received 17 July 2025; revised 24 October 2025; accepted 16 December 2025

The current digital landscape is highly prone to different cyberattacks that exploit vulnerabilities of an organization. It has posed a significant challenge to traditional security measures, which relied on implicit trust in entities. It resulted in various attacks, including insider attacks. To design a proactive digital posture for organizations, Zero Trust (ZT) has emerged as a better alternative. It is a set of guiding principles and ideologies that frame the way organizations think about trust and access. The literature review reveals that a gap between the conceptual proposals and their actual deployment exists. Existing studies were observed to use a limited number of parameters to evaluate trust. In addition, they use predefined weights for parameters of the trustscore makes the approach unscalable and globally unacceptable. To alleviate this issue, a ZT Architecture (ZTA) is proposed with a detailed incorporation strategy. A comprehensive list of parameters based on six categories is presented to evaluate the trust of the device. This investigation contributes by integrating trustscore and optimizing its weights associated using different metaheuristic optimization techniques. This makes the approach scalable and organization-agnostic. Various performance measures are used to evaluate the efficacy of the optimization techniques. The accumulation of parameters and the final score are published as a publicly available dataset. The differential evolution was found to be the best optimizer for the current scenario. This work aims to make a meaningful contribution to ZT, benefiting both the technological and societal domains.

Keywords: Artificial intelligence, Bioinspired optimization, Dynamic access control, Network security, Zero trust

Introduction

In today's digital world, cyber adversaries are finding new ways to penetrate the network for malicious purposes and gain. Securing digital identity, assets, etc., has become increasingly difficult. Before the birth of the Internet, organizations often relied on credential-based authentication schemes, signature-based antivirus, etc. Protecting mainframes, maintaining a list of Access Control (AC), etc., were also used. Such techniques were prone to various cyberattacks, including sophisticated malware attacks and unauthorized access.

The period of 1990–2000 gave birth to various technologies such as Intrusion Detection/Prevention System (IDS/IPS), stateful inspection firewall, etc. Organizations often rely on various technologies, including Virtual Private Networks (VPN) and endpoint security devices for security. However, they are based on the castle-and-moat model. It assumes that the adversaries are only outside the network. It can lead to insider attacks, data theft, etc.[1]

Around 2004, Paul Simmonds proposed the concept of 'deperimeterization.' Although it was effective, the architecture relies solely on VPN, which itself is an inherently trusted channel. Various tech giants, including Google, were attacked by 'Operation Aurora.' Similar attacks, including Operation Shady RAT, highlight the need for a secure design to withstand cyberattacks. In 2009, John Kindervag coined the term 'Zero Trust (ZT).' ZT is a set of principles that are tied together with the objective of protecting the enterprise. It opposes the concept of inherent trust in the network and aims to design a reactive posture. In 2014, Google initiated Beyond Corp and became one of the pioneers of ZT, followed by ZT Architecture (ZTA) of the National Institute of Standards and Technology (NIST).[2] This led to a wide adoption of ZTA in various domains, including healthcare.[3]

Various components of ZT orchestrate the entire scenario. They are designed with high cohesion and optimum coupling for a modularized architecture. Although the design and components can differ according to architectures, few core components are relevant in all:

*Author for Correspondence
E-mail: asonibgh@gmail.com

- *Policy Enforcement Point (PEP)*: It passes the access request to the appropriate components and enforces the authorization decisions.
- *Policy Engine (PE)*: It is the decision-making authority that evaluates the access request sent by the PEP.
- *Policy Administrator (PA)*: It is responsible for orchestrating the PEP to initiate or terminate sessions based on the decision made by the PE.

The PE and PA together constitute the Policy Decision Point (PDP). Few other components include policy information point, identity and access management, and Multi-Factor Authentication (MFA).

Although there are various architectures and implementation methods, a significant number of them lack concrete details of implementation and do not focus on all tenets of the NIST. The earlier methods relied extensively on the static methods of the trustscore calculation. They lacked the flexibility to adapt according to the organization's needs and improve security. Several studies have used fixed weights without proper selection criteria. Such approaches are not suitable for the modern security environment. This highlights the need to find suitable weights of the trustscore for an organization, rather than one's personal judgment. This paper, aims to address and bridge the gaps between the proposed ZTAs and their real-life implementation. A trust scoring scheme for continuous evaluation has been proposed using a wide range of parameters. An analysis of different Metaheuristic Optimization Techniques (MOT) has been performed to optimize the trustscore's weights.

Motivation

The US recorded the highest average data breach cost of $9.36 million among 16 regions for 14 consecutive years.[4] The Indian cybercrime coordination center projected an annual loss of 1.2 trillion INR due to cyberattacks.[5] Although organizations spend millions to improve security, they are prone to various cyberattacks. SonicWall reports an 8% year-over-year increase in the number of unseen malware variants identified each day.[6] In 2021–23, the adoption of ZT has doubled, with 61% of organizations implementing it and 35% planning to follow.

Objectives

The key objectives of this paper are as follows:

- Propose a ZTA that enhances security by assuming no trust.
- Demonstrate the implementation details of ZTA that can be designed for different organizational needs.
- Explore the use of Artificial Intelligence (AI) techniques in optimizing the trustscore weights.

Key Contributions

- *Comprehensive view of ZTA*: A detailed exploration of various ZTAs is provided that covers its principles, implementation, and real-world applications.
- *Implementation-oriented architecture for practical deployment*: A ZTA is proposed from an implementation-focused perspective to emphasize real-world implementation.
- *Comprehensive list of parameters for trust evaluation*: A holistic set of parameters that covers all relevant aspects was found to be missing in various studies. To address it, a comprehensive set of parameters for trust scoring is identified and evaluated.
- *Comparative analysis for score optimization*: A comparative analysis is performed on various MOT to optimize the weights used for trust evaluation.
- *Dataset for experimentation*: The dataset is released in the public repository to help researchers in their further experimentation.

Related Work

Among numerous studies on ZTA, the following studies are included as they are closely aligned with the focus and objectives of this research.

Tsai *et al.* proposed a cyclic ZTA, inspired by NIST,[2] to secure the entire process from the initial request to the grant decision.[7] The ZTA is logically divided into: the untrusted zone, the intelligence decision, and the service farm. Ahn *et al.* adopted the MITRE ATT&CK matrix to design ZTA and evaluate the same using various time-based metrics and efficiency-based metrics.[8]

Edge and cloud computing expand the attack surface by decentralizing data which makes ZTA important to mitigate risks.[9] Paul and Rao proposed a ZTA to secure the smart manufacturing industry, including Operational Technology (OT) in cloud-based infrastructure.[10] Chuan *et al.* proposed a ZTA implementation strategy, focusing on applications where users access intranet data through a cloud-

deployed application server.[11] Similarly, Ali *et al.* proposed a ZT maturity framework and a Proof-of-Concept (POC) to secure the Multi-access Edge Computing (MEC) environment.[12]

Wang *et al.* proposed a ZTA called 'SIX-Trust,' consisting of three layers to secure the essential aspects of 6G networks.[13] Chen *et al.* proposed a ZTA for a 5G-based healthcare system.[14] It evaluates the risk score for the devices by presenting the percentage as the sum of the weights of all the risk parameters.

In addition to the above literature, various reviews are included in the study, such as [15–17], etc., which provide a brief and detailed insight into the domain.

A brief comparison among various ZTA is highlighted in Table 1. Although existing ZTAs are observed to evaluate trust through different measures, they provide limited details on the parameters used. In addition, using predefined weights limits the adoptability of the approach in real-life. Although few researchers propose to use AI-based approaches to collect context, they are unclear about the features to be collected. The lack of existing datasets on the trustscores makes it difficult to build and test the efficacy of the AI models.

Proposed Architecture

The proposed ZTA aims to be innovative and easy to design. It is depicted in Fig. 1 and its working steps are as follows:

1. Device onboarding
2. Posture assessment and trustscore calculation
3. Microsegmentation
4. Dynamic access policy evaluation
5. Policy refinement & adaptive security

Device Onboarding

This is the first line of action where new devices are registered onto the platform. It begins with entering through a NGINX reverse proxy server.

Resource Discovery

To identify resources within an organization's network, detail about devices, users, applications, traffic, etc., shall be gathered. Various tools are used in this approach for resource gathering:

Table 1 — Analysis of various existing ZTAs

Problem addressed & Methods used [Ref]	Contribution of the paper	Conclusion [C]; Future Scope [FS]; Weakness [W]
A cyclic ZTA is proposed that revolves around the entire ecosystem[7]	Fine-grained AC Illustrated the importance of MITRE ATT&CK matrix	[C] Effective mitigation of challenges in designing and developing ZTA. [FS] The researchers aim to use the entire network telemetry data for decision-making. [W] The evaluation logic and parameters of the trustscore is not discussed.
Analyzing the impact of security strategies and management in ZTA[8]	Integration of ATT&CK matrix with ZTA	[C] Improved performance of ZTA, tested within a simulated test network [FS] The researchers aim to simulate the network in a real-life test environment. [W] The parameters and quantitative measurement of trust are not considered.
Proposed ZTA to secure devices in a manufacturing environment[9]	Lightweight signature-based entity authentication	[C] The proposed design integrates various components of ZTA to make it cyber-resilient. [FS] The researchers aim to simulate and test the ZTA under real-life scenarios. [W] The technical details for real-life replication are not proposed.
Proposed a ZTA, considering AC within a small-scale infrastructure[11]	Trustscore evaluation strategy is proposed	[C] The simulation provides insight into real-life implementation. [FS] The researchers aim to design an OS-agnostic ZTA in future research. [W] A small set of parameters, primarily related to OS and browsers, with fixed weights are used for trustscore evaluation.
Presented a POC for ZTA to secure the MEC environment[12]	Simulation of ZTA and its assessment using various quantitative parameters	[C] The research presents a minimum operational setup for MECs. [FS] The researchers aim to secure edge nodes from various cyber-attacks. [W] Evaluation of trust and parameters needed for it are not discussed.
Proposed a ZTA to secure 6G networks[13]	The feature selection and reinforcement-based decision-making engines conduct the optimum utilization of the context	[C] The integration of components from multiple domains in different layers strengthens the overall approach. [FS] The researchers aim to integrate explainability. [W] The implementation strategy gives limited knowledge on the datasets and parameters to evaluate trust.
A ZTA is proposed to secure 5G-based healthcare devices[14]	The exhaustive tests based on functionalities and modules were conducted	[C] The extensive experiments provide valuable information to test ZTAs. [FS] The researchers plan to test the approach in real-world environment and optimize it. [W] The predefined parameter weights limits scalability.

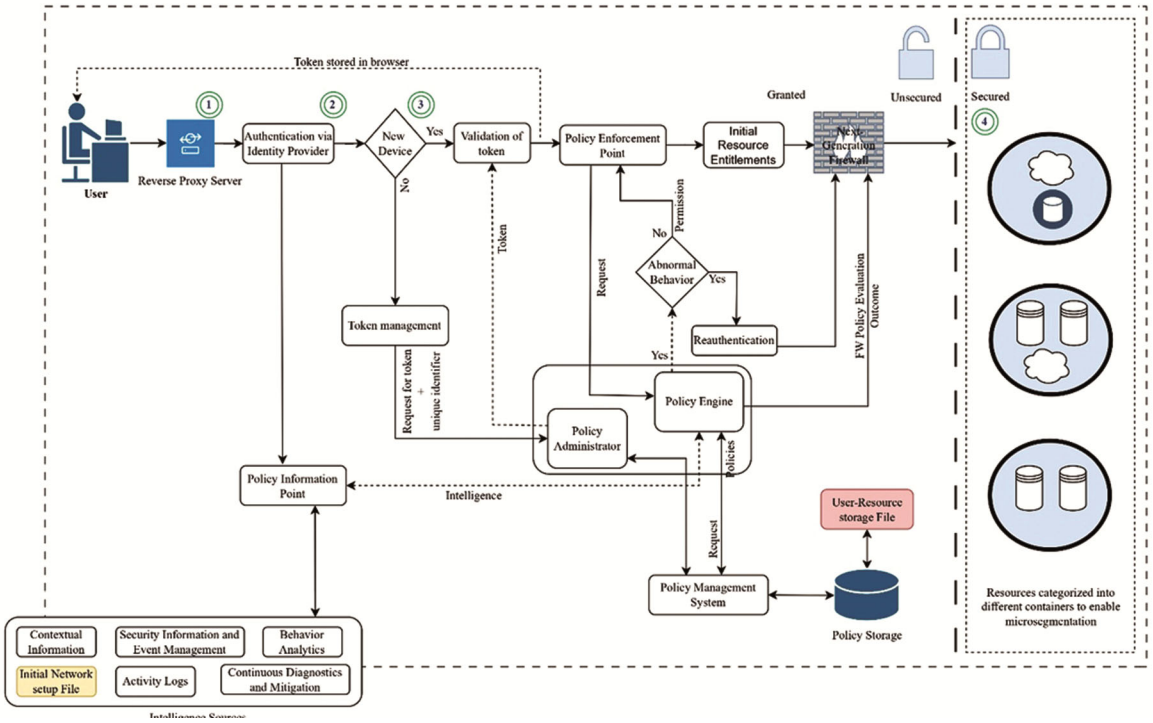

Fig. 1 — Detailed workflow of the proposed architecture

- Masscan: Network scanning.
- Scapy: Network traffic analysis.
- Elasticstack: Network monitoring.
- OSquery: Information gathering.

User Authentication

Okta is used for user authentication using Single Sign-On (SSO). SSO involves authentication using a single credential for login in multiple applications without revealing the original credentials to the relying party.

Access Token Generation and Validation

Short-lived access tokens can be used to prevent unauthorized use of credentials and as a means of reauthentication. To access resources within the organization, a valid token is required. JSON Web Tokens (JWT) are used here due to their wide popularity, stateless authentication, cryptographically signed payload, etc.

A new Elliptic Curve (EC) private key is generated and saved to a file. The corresponding public key is extracted and stored separately. A Flask app uses these keys for signing JWTs using the Elliptic Curve Digital Signature Algorithm with SHA-256 (ES256) algorithm. The username is used as the identity in the tokens, which could be extended using an '*additional_claims*' parameter. Tokens are created with the '*create_access_token()*' function from the '*flask-jwt-extended*' library.

Tokens are stored in the browser's local storage and in a database. Unlike cookies, local storage has a higher storage capacity and is not automatically sent with every request to the server. As it does not auto-expire itself, the current token's validity is matched with that of user's last stored token from the database.

Posture Assessment and Trustscore Calculation

The posture of a device is calculated on the basis of various parameters extracted from the device. They are extracted using various system calls.

Trustscore Optimization and Policy Reformulation

Trustscore can be calculate based on the parameters using Eq. (1).

$$S = \sum_{i=1}^{N} P_i W_i \quad \ldots (1)$$

where, S represents the final trustscore, N represents the total number of parameters. P_i and W_i represent the value and weight of the i^{th} parameter, respectively.

The corresponding weights for each parameter are used to emphasize the contributions of each parameter. It normalizes the score and ensures that scores stay bounded. The initial weights were assigned according to an expert approach and established by the organization. Logarithmic decay function is used such that the sum of all the weights is equal to 1. [18] It is presented in Eq. (2).

$$w_i = \frac{\log(N - i + 2)}{\sum_{j=1}^{N} \log(N - j + 2)} \quad \ldots (2)$$

where, w_i represents the normalized weight of the i^{th} parameter, N represents the total number of parameters, i represents the index of the parameter (higher priority parameters have lower indices), and *log represents* the natural logarithmic function.

It prioritizes the important parameters and allow the lower-rank parameters to contribute. It can help to rank the parameters according to the organization's needs. The extracted data and the final scores are fed to different MOTs with the aim of finding the suitable weights.

Microsegmentation

It is used to divide the network into multiple segments, each having security controls. Calico has been used in this study for microsegmentation. A separate set of policies is created using yet another markup language. It contains the ingress and egress rules with various parameters.

Resource Identification and Categorization

Resources and resource categories are classified according to predefined criteria such as functionality, importance, accessibility, etc. It could differ for each organization. There is no '*one-size-fits-all*' when it comes to security.[2]

The resources are classified into four categories:

- *Public*: Accessible to everyone without restrictions.
- *Internal*: Restricted to authorized employees.
- *Confidential*: Requires strict AC due to sensitive data.
- *Critical*: Essential for operations with high-security measures.

Each resource category is a policy file, created using the Abbreviated Language for Authorization (ALFA). It contains various rules with the access level to be granted in a certain context.[19] These levels are based on logical necessity and operational efficiency. It also reduces the attack surface. These are as follows:

- *No access (NA)*: Completely restricted; unauthorized users cannot view or interact.
- *Read-only with reauthentication (RR)*: Must authenticate to view but cannot modify.
- *Read-only (R)*: Can see basic info but cannot interact.
- *Partial access with reauthentication (PR)*: Limited actions but requires reauthentication for various tasks.
- *Partial access (P)*: Can perform basic tasks but lacks critical permissions.
- *Full access (F)*: Unrestricted control over the system.

Dynamic Access Policy Evaluation

Dynamic policies ensure that decisions are made in real time based on contextual factors. ALFA is used to design them. They are evaluated using a PDP, designed using Python. Various Policy Combining Algorithms (PCA) are used to take effective decisions. The following PCAs are used which can be toggled by the organization:

- *Permit-overrides*: The request is granted if at least one policy grants access.
- *Deny-overrides*: The final decision is denied if even one policy denies access.
- *First-applicable*: The decision of first policy determines the final decision.
- *Only-one-applicable*: Decision from a single policy is enforced. It results in an indeterminate outcome if multiple policies are applied.

The access to resources is based on various criteria such as current score, context, abiding by resource category policy, access policies, the level of access gained, etc. A sample AC based on trustscore is mentioned in Table 2. It shows different levels of access granted to different users for different categories of resources in different contexts. Although initial access was granted according to the policies, the final decision will vary dynamically.

Retaining Access Logs & Continuous Monitoring

Retaining access logs and continuous monitoring of resources is an important for securing the environment. Logs are collected using different sources, as listed in Table 3. They are also stored in a database.

Table 2 — Sample AC scenarios for different user categories and resource sensitivity levels

Resource	Role	Sensitivity	General access	Trustscore-based access				
				R1(0.1–1.9)	R2 (2–3.9)	R3 (4–5.9)	R4 (6–7.9)	R5 (8–10)
R1	Admin	Critical	Y	RR	PR	PR	P	F
R2	Admin	Confidential	N	N	N	N	N	N
R3	HR manager	Internal	Y	RR	R	PR	P	F
R4	Admin	Public	Y	R	PR	P	F	F

Table 3 — Categories of logs extracted for the proposed zero trust architecture

Log type	Purpose	Extraction method
API request logs	Track API access and security	Flask middleware
Authentication logs	Track login/logout activities	/var/log/auth.log
Network traffic logs	Monitor incoming/outgoing packets	Tcpdump, iptables logging
Privileged access logs	Identify role-based access violations	Auditd, /var/log/auth.log
System logs	OS-level events and errors	Journalctl, /var/log/syslog

Policy Refinement & Adaptive Security

In ZTA, efficient access policy management and retrieval are important. Graph databases provide a flexible way to store them by capturing their complex relationships. Neo4j is a widely used graph database that supports this need. It also provides encryption for data at rest and in transit using volume encryption and encrypted Bolt communication, respectively. Organizing policies as connected nodes makes querying faster and more intuitive. For stronger protection, additional encryption can be applied using attribute-based encryption. Even if Neo4j is compromised, the data remains unintelligible without the decryption key. This supports secure and responsive policy handling in dynamic environments.

Implementation Details

The proposed architecture was built using Angular framework and Python with the help of various open-source tools. It was designed and deployed on a local server with a Red Hat Enterprise Linux 8.4, 64GB RAM, an Intel Xeon Silver 4144 CPU (2.20GHz × 20), and a 64-bit architecture system. In order to simulate a real-life network, a lab having 10 client systems was designed and connected to the server. The client systems were running on an Ubuntu 20.04.06 LTS, 8GB RAM, an Intel Core i7-10700 CPU @ 2.90GHz x 16 processor, and a 64-bit architecture system. The logical schema of the lab is presented in Fig. 2. The lab is logically segmented into four groups using appropriate network devices, with Group A having internet access, Group B accessing the intranet with authentication, Group C blocked from the internet, and Group D configured for limited admin access, similar to a DMZ.

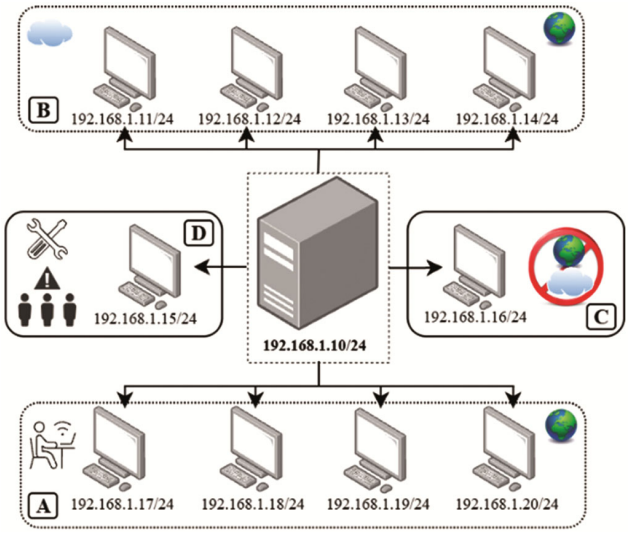

Fig. 2 — Network arrangement of the proposed architecture

In order to optimize the parameters of the trustscore, various MOTs were used, and a comparative analysis is conducted. The objective is to find suitable weights for each parameter and minimize the error. The overall steps are as follows:

- *Importing libraries and setting up the environment*: Various Python libraries were used for statistical analysis and visualizations.
- *Data preparation*: The dataset was loaded in the environment. Since the data were already preprocessed, no further alterations were made.
- *Define objective function*: Mean Squared Error (MSE) is used as the objective function since the algorithm can search for weights that minimize the difference between the model's predictions and the true values. It leads to better performance.

- *Define bounds for weights*: The weights (W) for each input parameter are bounded between -1 and 1. This is done by defining a list of bounds.
- *Optimization*: Since there is no empirical way to determine the best-fit algorithm for a specific domain, different MOTs, including Particle Swarm Optimization (PSO), Simulated Annealing (SA), Genetic Algorithm (GA), and Differential Evolution (DE) were tested. They are useful when the problem space is non-linear, non-convex, or has multiple local optima, with a flexible global search strategy. Therefore, their selection provides a suitable choice for the criteria of the current study.
- *Making Predictions and Evaluating the Model*: Predictions (*y_pred*) are made by taking the dot product of the feature matrix *X* and the optimal weights found by each algorithm. The predictions are compared with the true values *(y)* using various metrics.

Since there are multiple parameters to optimize but a single objective (to minimize the error), this problem can be considered a high-dimensional single-objective optimization problem.

Evaluation and Analysis

This section presents the evaluation measures and analysis gained from the experimentation. A brief comparison among various ZTAs based on Stafford [2], is presented in Table 4.

Device Onboarding

The landing of any device with NGINX helps to mask the server identity, mitigate Distributed Denial of Service (DDoS) attacks, etc. Studies show that NGINX outperforms other proxies based on various parameters.[21] SSO for authentication reduces the entry points for potential attackers. Self-hosted SSO requires manual configuration, high-availability setup, etc., which could be costly. Okta provides quick setup for platform-independent integration compared to other providers. Its compliance with various standardizations, adaptive authentication, etc., makes it a suitable choice for integration with ZT.[21,22] JWT tokens and verification adds another layer of security. JWT is stateless and terminates the need for server-side session storage. It reduces the risk of session hijacking and replay attacks. It also makes it a better choice for distributed systems and microservice architectures.[23]

Posture Assessment and Trustscore Calculation

The frequent calculation of the trustscore at every *N* number of seconds indicates that the parameters are also extracted, and posture is evaluated. It improves trust calculation and assigns a new level of privilege for the resources that were entitled to users. Dynamic AC and adaptive risk-based decision-making can be achieved using this measure. Various attacks, including insider threats, could be defended against using the selected approach. Using the weighting technique to evaluate scores gives us the flexibility to prioritize and rank parameters according to the organizations' needs.[24] Since a universal approach does not work to calculate trustscores, the use of optimized weights helps provide more accurate trustscores.[25]

Dataset

Various logs, parameters, scores, etc., were extracted from each client of the environment. The dataset used in the study is the accumulation of trustscore and its parameters over multiple runs of the testbed on the client system. 5,000 samples were collected, with 29 features (parameters) and an output

Table 4 — Comparison of various ZTAs

Type of study [Ref]	Broad domain	Identity verification	Network segmentation	Reauthentication	Device posture assessment	Logging & monitoring
Architecture[7]	Multipurpose	MFA + PKI + Biometrics	✓	✓	✓	✓
Architecture[10]	Smart manufacturing	MFA	✓	✓	✓	✗
Architecture[14]	5G-based healthcare	Credentials	✓	✓	✓	✓
Architecture[13]	6G network security	DPKI + NFVI	✗	✗	✓	✓
Implementation[11]	Cloud-based infrastructure	Credentials	✓	✗	✓	✓
Architecture + Implementation[12]	MEC environment	Identity + Location	✗	✗	✓	✓
Architecture + Implementation (Proposed)	Multipurpose	SSO + MFA + PKI-aligned JWT	✓	✓	✓	✓

Table 5 — List of parameters used for score calculation

Category	Parameters	Values				
		0	0.25	0.50	0.75	1
Security & compliance	Firewall status	Off	—	—	—	On
	Sudo privilege	No	—	—	—	Yes
	Failed SSH counts	Critical (>10)	High (6–10)	Moderate (3–5)	Low (1–2)	0
	Secure boot status	Disabled	—	—	—	Enabled
System maintenance	APT version	Superseded	—	—	—	Latest
	OS version	Superseded	—	—	—	Latest
	Reputable BIOS vendor	No	—	—	—	Yes
	Reauthentication	No	—	—	—	Yes
Network & location	Location	Distant	—	—	—	Within the range
	Institute Ethernet	No	—	—	—	Yes
	Disk backup status	Disabled	—	—	—	Enabled
	IP reputation	Blacklisted (0–19)	Suspicious (20–39)	Moderate (40–69)	Good (70–89)	Trusted (90–100)
	Network latency (in milliseconds)	>300	200–300	100–199	50–99	<50
	Traffic transfer rate (in Mbps)	>500	101–500	51–100	10–50	<10
Security & patch management	Upgradable apps	>15	10–15	5–9	1–4	0
	Trusted apps	<5	5–10	11–15	16–20	>20
	Insecure apps	>15	10–14	5–9	1–4	0
	Pending security updates	Critical (>10)	High Risk (6–10)	Moderate Risk (3–5)	Minimal Risk (1–2)	Up to Date (0)
	Number of open ports	Critical (>1000)	High Risk (201–1000)	Moderate Risk (51–200)	Secure (11–50)	Highly secure (0–10)
Power & resource monitoring	CPU Temperature	>90%	75–90 %	50–74%	25–50%	<25%
	Power consumption	>90%	75–90 %	50–74%	25–50%	<25%
System performance & resource management	CPU usage (per second)	Critical (90–100%)	High (70–90%)	Moderate (40–70%)	Optimal (10–40%)	Idle (0–10%)
	Memory usage	>90%	75–90%	50–74%	25–50%	<25%
	Disk usage	>90%	75–90%	50–74%	25–50%	<25%
	Swap usage	>90%	75–89%	50–74%	25–50%	<25%
	Number of active processes	>200	150–200	101–149	50–100	<50
	I/O R/W ops/s	>1000	501–1000	101–500	51–100	<50
	Zombie process count (per second)	>10	6–10	3–5	1–2	0
	Context switches count (per second)	>10000	5001–10000	1001–5000	500–1000	<500

label (trustscore), kept in a CSV file. It is publicly available on GitHub.[26] The parameters are categorized into six categories, as presented in Table 5. They are selected because they can contribute to identifying the overall posture of the device, detecting anomalies, etc. For example, a high number of failed Secure Socket Shell (SSH) counts suggests intrusion attempts and brute-force attacks. Advanced Package Tool (APT) version ensures secure package verification and helps maintain overall system integrity of the OS. In addition, selecting a trusted Basic Input/Output System (BIOS) vendor ensures firmware integrity,

reduces attack surface, etc. Reputation of Internet Protocol (IP) can be used to detect spam, malware distribution, etc. Monitoring anomalies and sudden spikes in Input/Output (I/O) Read/Write (R/W) operations per second (ops/s) indicate malware activity, intrusions, etc.

The dataset is preprocessed and does not contain missing values. The categorical values are converted to 0 or 1 to represent their binary nature. Parameters having values within a certain range are normalized to the range of 0 to 1. It removes potential biases due to different scales.

Analysis of the Optimization Techniques

The performance achieved for various MOTs are presented in Table 6. The error distributions and comparative visualization are presented in Figs 3–7.

The objective function uses MSE as a loss function. Linear Regression (LR) is used as the prediction model. It creates a smooth error surface when combined with MSE. It provides a well-defined optimization with a single global optimum. With comparable parameters and nearly zero MBD, DE and SA performed the overall best. Their error rates are also quite similar for this setting. Low MSE indicates that they successfully converged to the optimal point. It shows their reliability for LR-based optimization tasks.

SA achieved the same level of performance 48% faster than DE. PSO and GA performed significantly worse. GA had even higher MSE and R^2 than PSO. However, GA was the fastest algorithm, followed by PSO. Although this indicates their computational efficiency, they are not suitable for this problem. These results highlight a trade-off: SA provides both

Table 6 — List of parameters used for score calculation

Algorithm	MSE	R^2	MAE	EVS	MBD	Execution time (s)
DE	0.000010	0.998553	0.002648	0.998553	−0.000011	38.824610
SA	0.000010	0.998553	0.002648	0.998553	−0.000013	20.130772
PSO	0.062381	−8.245306	0.201282	−8.243489	0.003501	1.263597
GA	0.111772	−15.565339	0.269006	−15.521610	0.017177	0.636139

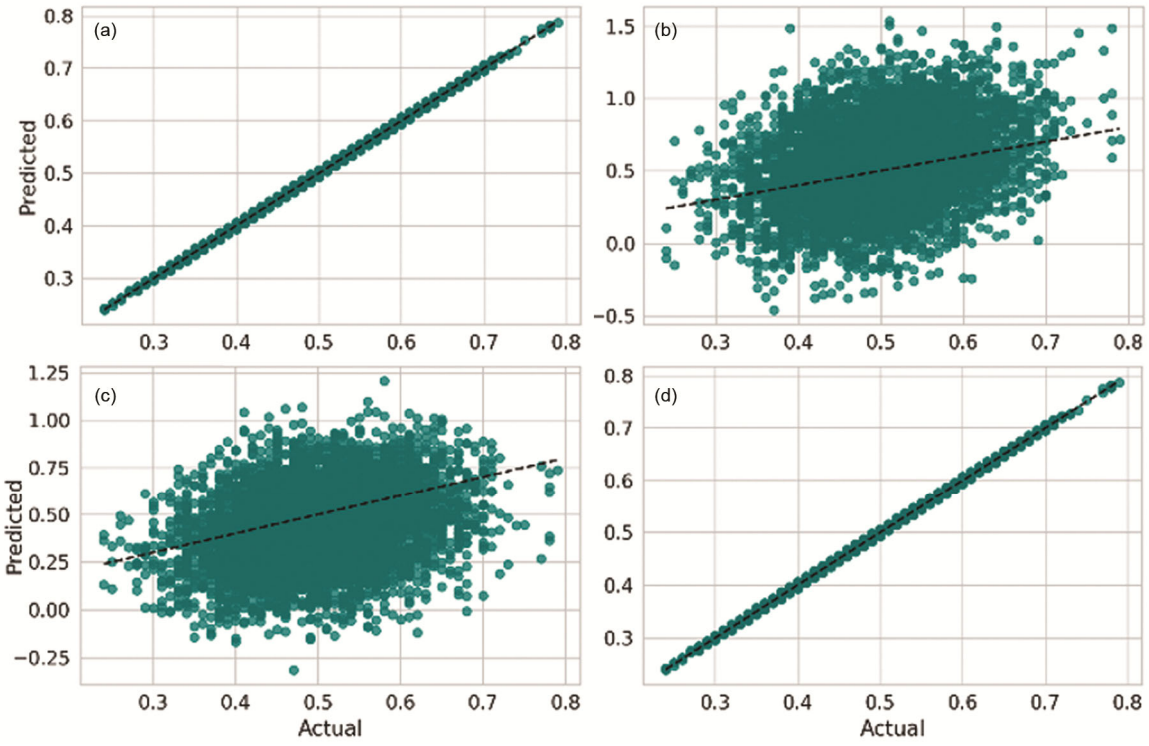

Fig. 3 — Visualization of predicted vs. actual values plot for different algorithms: (a) Differential evolution, (b) Particle swarm optimization, (c) Genetic algorithm, (d) Simulated annealing

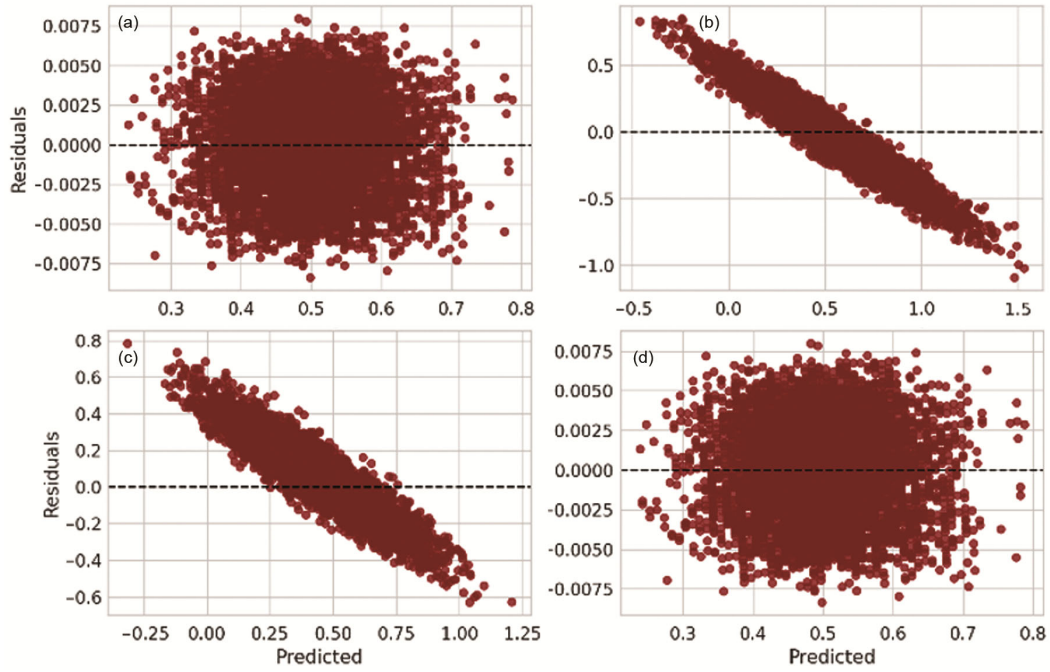

Fig. 4 — Visualization of residual plot for different algorithms: (a) Differential evolution, (b) Particle swarm optimization, (c) Genetic algorithm, (d) Simulated annealing

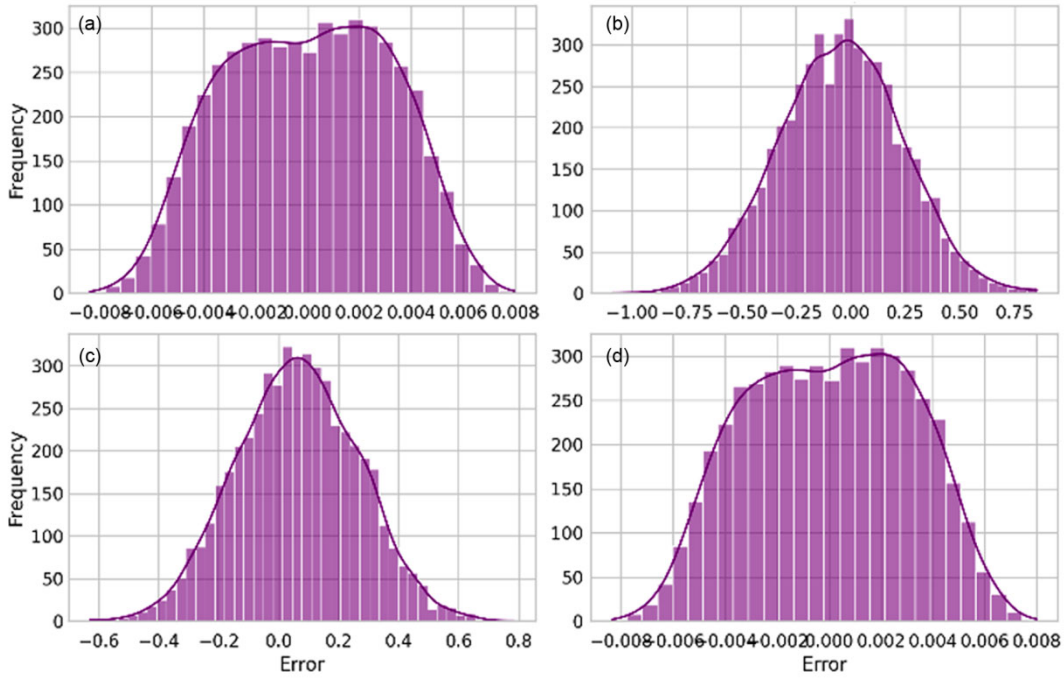

Fig. 5 — Visualization of error distributions for different algorithms: (a) Differential evolution, (b) Particle swarm optimization, (c) Genetic algorithm, (d) Simulated annealing

better prediction and low execution time; GA and PSO favor speed over prediction in this case.

Execution time is directly impacted by the fundamental differences between the MOTs. DE uses vector-based mutation and crossover operations between individuals. It focuses on global search using population-based sampling. SA uses a simpler perturbation and acceptance mechanism. Although

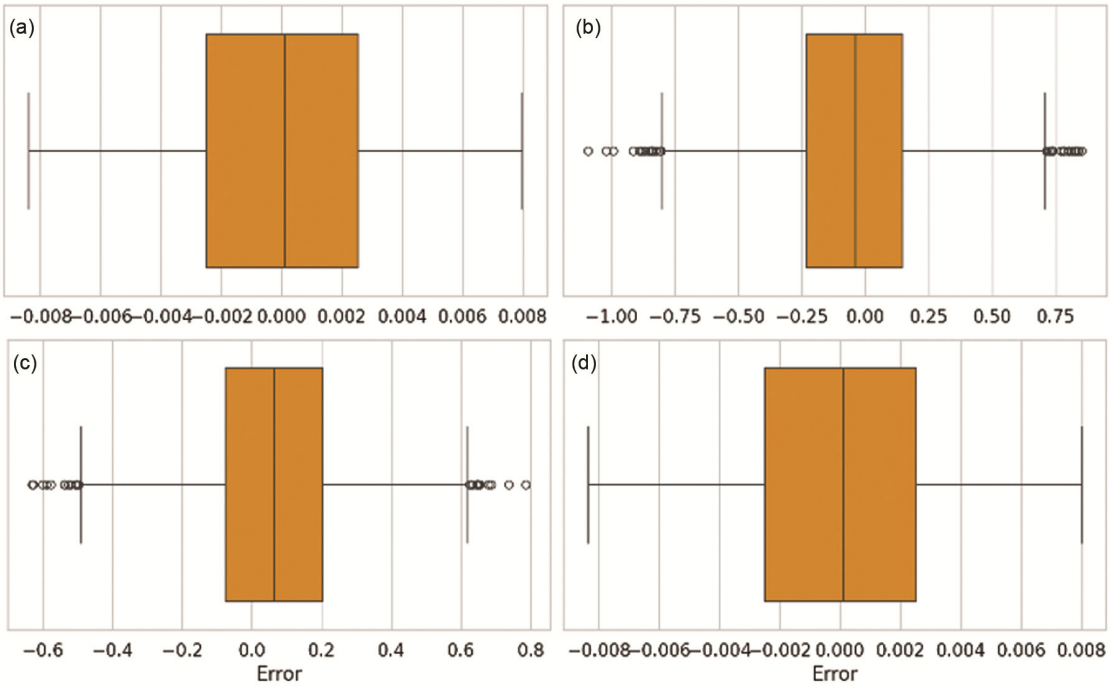

Fig. 6 — Visualization of error boxplot for different algorithms: (a) Differential evolution, (b) Particle swarm optimization, (c) Genetic algorithm, (d) Simulated annealing

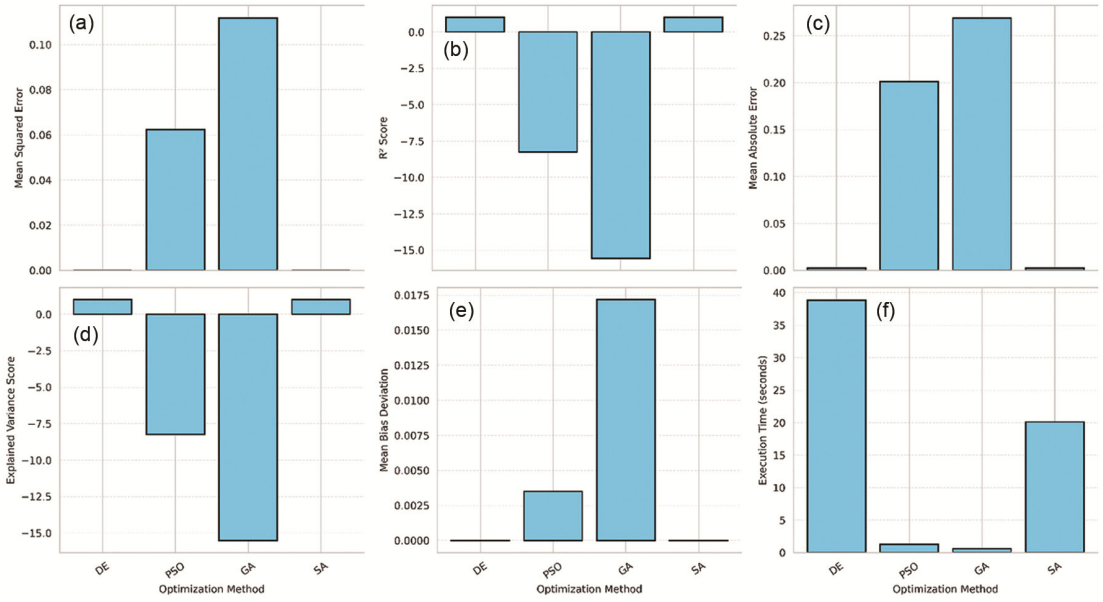

Fig. 7 — Performance comparison among various optimization techniques: (a) MSE, (b) R2, (c) MAE, (d) EVS, (e) MBD, (f) Execution time

SA is greedier, which can reduce time, but it can risk getting stuck in local minima. As a result, it is important to further explore DE for optimum results. Similar experiments are conducted using eight different DE's strategies.

It could be observed from Table 7 that all strategies produced nearly similar results. Among them, *best1bin* achieved high performance with the lowest execution time. It was the most time-efficient strategy because of its balance between exploration and exploitation. It was 62% faster than the runner-up strategy, *best2exp*. The *rand2bin* required the longest execution time. Strategies such as *rand2bin* and *best2bin* perform more extensive exploration due to

Table 7 — List of parameters used for score calculation

Algorithm	MSE	R^2	MAE	EVS	MBD	Execution Time (s)
rand1bin	0.000010	0.998553	0.002648	0.998553	−0.000012	103.922256
rand1exp	0.000010	0.998553	0.002648	0.998553	−0.000013	105.191409
rand2bin	0.000010	0.998553	0.002648	0.998553	−0.000012	105.792503
rand2exp	0.000010	0.998553	0.002648	0.998553	−0.000012	104.874862
best1bin	0.000010	0.998553	0.002648	0.998553	−0.000016	39.343369
best1exp	0.000010	0.998552	0.002649	0.998552	−0.000010	104.718917
best2bin	0.000010	0.998553	0.002648	0.998553	−0.000013	105.431687
best2exp	0.000010	0.998553	0.002648	0.998553	−0.000011	103.863048

complex mutation operations. The *best1exp* achieved the lowest MBD compared to all. Although performance metrics remain constant, differences in runtime highlight trade-offs between exploration and computational efficiency.

Microsegmentation

Organizations can enforce strict ACs and reduce the lateral movement of threats.[27] Calico is used to perform microsegmentation. It is faster than various platforms like NSX-T and Cisco ACI, due to its lightweight architecture based on Border Gateway Protocol (BGP) and native IP routing. It also outperformed other platforms in terms of container network interface by being 6×faster.[28] Maintaining an independent and separate policy file for different resource categories gives the flexibility to modify one without affecting others. It ensures that permissions are clearly defined and auditable. The nomenclature of different resource categories and sensitivity levels is inspired by various platforms that maintain different categories for different purposes. The details are drawn from various leading companies in different domains. Most commonly used and applicable sensitivity levels were selected. The resource access levels are also determined in a similar way from the same platforms, based on the selected sensitivity level of the resource.

Dynamic Access Policy Evaluation

The real-time adaptation of permissions is provided by the dynamic nature of access policies. Although the core syntax and structure of the policies might remain the same, the techniques used to evaluate these policies dynamically differ from static policies. The use of contextual data in real-time can help to enforce dynamic policies. ALFA makes the policies shorter in length and size, with the same dynamics compared to other languages.[19] Dynamic access evaluation mitigates account takeover, unauthorized privilege escalation, DDoS, session hijacking, etc.

Discussion

The proposed architecture lays the foundation for new readers to learn and redefine the security measures of an organization. The real-life application of the proposed architecture may vary, depending on the organization in which it is deployed. The key areas of concern and possible improvements based on a critical evaluation are discussed below:

- *Open-source tools—a double-edged sword*: Their use in ZTA requires careful consideration, since the source code is publicly available, even to adversaries.[29] Although they are widely used for various purposes,[30] some advocate that third-party applications introduce security concerns and should be avoided.[2]
- *Exploring more optimization algorithms*: Since there are hundreds of optimization algorithms in existence, their exhaustive analysis could be fruitful in the future with the aim of finding the best optimizer.
- *Need for standardization*: There is a need for standardization of the ZTAs and the parameters to consider when designing, deploying, evaluating, and stress-testing it. Different ZTAs use distinct set of parameters that is not be applicable to all.[8,15]
- *Context collection*: Information, its extraction strategy, processing, and storage differ for each architecture that needs standardization. In addition, collecting too much could raise privacy concerns, while collecting too little could lead to ineffective decision-making.
- *Integrating various Next-Gen technologies*: Various emerging technologies can be incorporated to leverage their strengths.[31–33]
- *Dynamic Policy Adaptation and refinement:* The proposed approach provides a foundation for improving security systems through the use of dynamic access policies and adaptation. The

proposed policy refinement and dynamic adaptation approach requires further maturation.[34–36] In addition, various limitations in the existing works, including limited context awareness and interoperability in heterogeneous systems, makes it difficult for real-world deployment.

Conclusions

The proposed architecture and testbed aim to bridge the gap between theoretical architectures and real-life implementation. A comprehensive explanation of the implementation steps, tools, and techniques is provided. This study aims to reduce the use of static weights for trustscore calculation using MOTs. An analysis among various MOTs has been performed to optimize weights to calculate the trustscore. It provides the appropriate adjustment of the weights according to the organization. The data from the testbed is accumulated to form a dataset. The proposed architecture can be considered platform-agnostic with few improvements. Future studies can integrate emerging technologies to further strengthen ZTA. In addition, the simplification and enhancement of the privacy while safeguarding the personal identities is a challenging task in ZTA, especially in a federated identity environment.

References

1 Tripwire, Insider threats: root causes and mitigation practices, https://www.tripwire.com/state-of-security/insider-threats-root-causes-mitigation-practices, accessed on 23 Feb 2025.

2 Stafford V, Zero trust architecture, *NIST Spec Publ*, 800 (2020) 207, DOI: 10.6028/NIST.SP.800–207.

3 Zyoud B & Lutfi S L, The role of information security culture in zero trust adoption: insights from UAE organizations, *IEEE Access*, **12** (2024), 72420–72444, DOI: 10.1109/ACCESS.2024.3402341.

4 IBM Data Breach Report, https://www.ibm.com/reports/data-breach, accessed, 23 Feb 2025.

5 The Hindu, Cyber fraud losses could amount to 0.7% of GDP, MHA study projects, https://www.thehindu.com/sci-tech/technology/cyber-fraud-losses-could-amount-to-07-of-gdp-mha-study-projects/article68788093.ece, accessed, 23 Feb 2025.

6 SonicWall, SonicWall cyber threat report, https://www.sonicwall.com/resources/white-papers/2025-sonicwall-cyber-threat-report, accessed, 23 Feb 2025.

7 Tsai M, Lee S & Shieh S W, Strategy for implementing of zero trust architecture, *IEEE Trans Reliab*, **73(1)** (2024) 93–100, DOI: 10.1109/TR.2023.3345665.

8 Ahn G, Jang J, Choi S & Shin D, Research on improving cyber resilience by integrating the Zero Trust security model with the MITRE ATT & CK matrix, *IEEE Access*, **12** (2024) 89291–89309, DOI: 10.1109/ACCESS.2024.3417182.

9 Ashfaq F, Ahad A, Hussain M, Shayea I & Pires IM. Enhancing zero trust security in edge computing environments: Challenges and solutions, in *World Conf Inf Syst Technol*, **2024**, 433–444 (Springer Nature, Switzerland), DOI: 10.1007/978-3-031-60221-4_41.

10 Paul B & Rao M, Zero-trust model for smart manufacturing industry, *Appl Sci*, **13** (2023) 221, DOI: 10.3390/app13010221.

11 Chuan T, Lv Y, Qi Z, Xie L & Guo W, An implementation method of zero-trust architecture, *J Phys Conf Ser*, 1651 (2020) 012010, DOI: 10.1088/1742-6596/1651/1/012010.

12 Ali B, Hijjawi S, Campbell L H, Gregory M A & Li S, A maturity framework for zero-trust security in multiaccess edge computing, *Secur Commun Netw*, **2022** (2022) 3178760, DOI: 10.1155/2022/3178760.

13 Wang Y, Kang X, Li T, Wang H, Chu C K & Lei Z, SIX-Trust for 6G: Toward a secure and trustworthy future network, *IEEE Access*, **11** (2023) 107657–107668, DOI: 10.1109/ACCESS.2023.3321114.

14 Chen B, Qiao S, Zhao J, Liu D, Shi X, Lyu M, Chen H, Lu H & Zhai Y, A security awareness and protection system for 5G smart healthcare based on zero-trust architecture, *IEEE Internet Things J*, **8** (2020) 10248–10263, DOI: 10.1109/JIOT.2020.3041042.

15 Cao Y, Pokhrel S R, Zhu Y, Doss R & Li G, Automation and orchestration of zero trust architecture: Potential solutions and challenges, *Mach Intell Res*, **21** (2024) 294–317, DOI: 10.1007/s11633-023-1456-2.

16 Dhiman P, Saini N, Gulzar Y, Turaev S, Kaur A, Nisa K U & Hamid Y, A review and comparative analysis of relevant approaches of zero trust network model, *Sensors*, **24** (2024) 1328, DOI: 10.3390/s24041328.

17 Syed N F, Shah S W, Shaghaghi A, Anwar A, Baig Z & Doss R, Zero trust architecture (ZTA): A comprehensive survey, *IEEE Access*, **10** (2022) 5714379, DOI: 10.1109/ACCESS.2022.3174679.

18 Borda M, Fundamentals in Information Theory and Coding, Berlin: *Springer Science & Business Media*; 2011.

19 Dimitrakos T, Dilshener T, Kravtsov A, La Marra A, Martinelli F, Rizos A, Rosetti A & Saracino A, Trust aware continuous authorization for zero trust in consumer internet of things, in *IEEE 19th Int Conf Trust Secur Privacy Comput Commun (TrustCom)*, **2020,** 1801–1812, DOI: 10.1109/TrustCom50675.2020.00247.

20 W3Techs, Usage statistics and market share of Apache vs NGINX, https://w3techs.com/technologies/comparison/ws-apache,ws-nginx, accessed, 17 Mar 2025.

21 Zscaler, Zscaler and Okta enhance enterprise cybersecurity with new zero trust integrations, https://www.zscaler.com/press/zscaler-and-okta-enhance-enterprise-cybersecurity-new-zero-trust-integrations; 2025. Accessed 2025 Mar 01. Available from:.

22 Wagenseil P. Who's using what: Results from the 2025 Okta Businesses at Work, https://www.scworld.com/resource/whos-using-what-results-from-the-2025-okta-businesses-at-work-report, accessed, 01 Mar 2025.

23 Shikhverdiyev I, Babayev E, Rahimli C, Rahimli N & Aslanova H. Secure authentication in e-government 2.0: a comparative analysis of traditional session-based and modern JWT-based authentication, *Int Sci J Eng Agric*, **3** (2024) 117–129, DOI: 10.46299/j.isjea.20240306.12

24 Wang Z, Yu X, Xue P, Qu Y & Ju L, Research on medical security system based on zero trust, *Sensors*, **23** (2023) 3774, DOI: 10.3390/s23073774.

25 Bhattarai H, Kulkarni A & Niamat M, Trust Score-based zero trust architecture for advanced metering infrastructure security, in IEEE *Natl Aerosp Electron Conf* (NAECON) **2024**, 334–339, DOI: 10.1109/NAECON61 878.2024. 10670642.

26 Asonibgh, Trustscore-dataset, https://github.com/asonibgh/Trustscore-dataset.git, GitHub repository, Accessed, 20 Mar 2025.

27 Zanasi C, Marchetti M & Colajanni M, Cybersecurity domains: A design pattern for creating zero trust architectures through microsegmentation, in *IEEE Conf Dependable Auton Secure Comput (DASC)*, 2024 15–22, IEEE, DOI: 10.1109/DASC64200.2024.00009.

28 Tigera, Calico delivers WOW effect with 6x faster encryption than any other solution, confirms leadership in latest independent CNI benchmark tests, https://www.tigera.io/blog/calico-delivers-wow-effect-with-6x-faster-encryption-than-any-other-solution-confirms-leadership-in-latest-independent-cni-benchmark-tests, Accessed 20 Mar 2025.

29 Hilbig T, Schreck T & Limmer T, 'State of the union': evaluating open source zero trust components, in *Int Worksh Secur Trust Manage*, **2023,** 42–61, Springer, DOI: 10.1007/978-3-031-47198-8_3.

30 Federici F, Martintoni D & Senni V, A zero-trust architecture for remote access in industrial IoT infrastructures, *Electronics*, **12** (2023) 566, DOI: 10.3390/electronics 12030566.

31 Joshi H, Emerging technologies driving zero trust maturity across industries, *IEEE Open J Comput Soc*, **6** (2025) 25–36, DOI: 10.1109/OJCS.2024.3505056.

32 Das S, Rout J & Mishra M, Blockchain technology: applications and open issues, in *IEEE Int Conf Commun Comput Internet Things* (IC3IoT), **2022,** 1–6, DOI: 10.1109/IC3IOT53935.2022.9767871.

33 Aleisa M A, Blockchain-enabled zero trust architecture for privacy-preserving cybersecurity in IoT environments, *IEEE Access*, **13** (2025) 18660–18676, DOI: 10.1109/ACCESS. 2025.3529309.

34 da Silva G R & dos Santos A L, Adaptive access control for smart homes supported by zero trust for user actions, *IEEE Trans Netw Serv Manage*, **2024,** Article 3492379, DOI: 10.1109/TNSM.2024.3492379.

35 Alnaim A K, Adaptive zero trust policy management framework in 5g networks, *Mathematics*, **13** (2025) 1501, DOI: 10.3390/math13091501.

36 Zhang X, Wang D, Zhu Y, Chen W, Chang Z & Han Z, Zero-trust based robust federated learning against betrayal behaviors, *IEEE Trans Mob Comput*, **2025,** Article 3591632, DOI: 10.1109/TMC.2025.3591632.