

Cybersecurity-metric-driven software quality assessment for critical information and communication systems

Roman Hamretskyi^{1,†}, Viktor Gnatyuk^{2,*,†} and Krzysztof Pytel^{3,†}

¹Independent Researcher, Slavuta, 30000, Ukraine

²State Scientific and Research Institute of Cybersecurity Technologies and Information Protection, Maksym Zalizniak Str. 3/6, Kyiv, 03142, Ukraine

³University of the National Education Commission, Podchorążych Str. 2, Krakow, 30-084, Poland

Abstract

Modern information and communication systems (ICS) are increasingly implemented as software-defined, virtualized, and continuously monitored infrastructures. In such environments, software quality is closely related to cybersecurity because insufficient access control, unstable authentication, delayed security-policy enforcement, incomplete monitoring, or loss of security telemetry may reduce the protection level of the infrastructure. This paper presents a cybersecurity-metric-driven model for software quality assessment in ICS. The model treats security-relevant static metrics and runtime cyber-resilience metrics as complementary components of a single assessment procedure. The static component is aligned with the ISO/IEC SQuaRE family and includes security requirements coverage, access controllability, component isolation, and secure adaptability. The runtime component uses measurements obtained in a Proxmox VE-based testbed: unauthorized RESTCONF access-rejection latency for an OpenDaylight SDN controller, authentication-service responsiveness for a Spring Boot microservice, and MQTT telemetry delivery for an IoT component. The real experiment used three profiles: baseline, stressed, and critical. Across these profiles, the SDN access-control scenario preserved a 100% expected 401 response rate, the authentication service preserved a 100% expected 200 response rate, and the IoT component preserved a 100% MQTT publish-success rate. However, latency increased and throughput decreased under stress. The normalized runtime cybersecurity score decreased from 1.000 to 0.250 for the SDN RESTCONF access-control scenario, from 1.000 to 0.200 for the authentication service, and from 1.000 to 0.550 for the IoT telemetry component. The results show that cybersecurity-oriented runtime metrics can expose operational degradation even when expected security responses remain correct.

Keywords

cybersecurity metrics, software quality, information and communication systems, cyber hygiene, runtime assessment, access control, authentication resilience, telemetry integrity, operational resilience, observability, SDN, IoT

1. Introduction

ICS have become software-intensive infrastructures that combine network control planes, microservices, cloud and virtualized platforms, Internet of Things (IoT) telemetry, monitoring stacks, and automated deployment pipelines. In this context, software quality is not limited to implementation correctness, maintainability, or portability. It also influences whether the infrastructure can preserve access control, authentication stability, observability, secure configuration, and cyber resilience under changing operational conditions. Consequently, software quality assessment for ICS should include cybersecurity metrics when the assessed components participate in protection, monitoring, or control functions.

An earlier integrated static-dynamic quality model for ICS combined product-level metrics based on the ISO/IEC SQuaRE family with operational metrics such as latency, throughput, and packet loss. The present paper develops that foundation by focusing on cybersecurity-relevant software

CMiGIN'26: Fifth International Workshop on Conflict Management in Global Information Networks, June 26–27, 2026, Krakow, Poland

*Corresponding author.

†These authors contributed equally.

✉ roman.hamretskyi@icloud.com (R. Hamretskyi); gnatyukvo@gmail.com (V. Gnatyuk); krzysztof.pytel@uken.krakow.pl (K. Pytel)

ORCID 0000-0001-5514-9783 (R. Hamretskyi); 0000-0002-4916-7149 (V. Gnatyuk); 0000-0002-1924-8351 (K. Pytel)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

quality. This extension is necessary because cyber hygiene cannot be evaluated only through generic performance indicators. A service may demonstrate acceptable throughput and still be unsuitable for a protected environment if authentication latency becomes unstable, monitoring channels lose telemetry, security-policy updates are delayed, or access-control-related quality is insufficient.

This problem is particularly relevant for critical communication infrastructures, SDN/NFV environments, IoT platforms, and 5G/6G service ecosystems. These environments rely on software components that support dynamic configuration, remote control, distributed telemetry, and automated scaling. If the software responsible for authentication, policy enforcement, or telemetry exchange becomes unstable, the system may lose the ability to detect incidents, enforce security rules, or maintain service-level agreements. The assessment model therefore has to show not only whether the software is functionally complete, but also whether it preserves security-relevant behavior under stress.

The research problem addressed in this paper is the limited cybersecurity representativeness of software quality assessment when runtime security-relevant behavior is excluded from the model. Static security metrics, such as access controllability, are necessary, but they do not show whether authentication, policy enforcement, or security telemetry remain stable during operational stress. Conversely, runtime metrics without a structured quality model do not explain how degradation affects the overall assessment of the software product. The proposed approach integrates both dimensions.

The goal of the study is to develop and validate a cybersecurity-metric-driven model for software quality assessment in ICS. The model is intended to support release decisions, CI/CD gates, monitoring dashboards, and risk-aware operational control. It makes cybersecurity metrics central to the assessment and interprets the Proxmox VE experimental testbed as a controlled cyber hygiene stress environment.

The main contributions are as follows:

- a cybersecurity-oriented restructuring of an integrated static-dynamic software quality model for ICS;
- a formal cybersecurity quality index that combines static security metrics and runtime cyber-resilience metrics;
- a metric mapping between SQuaRE quality characteristics, information security measurement logic, and observable infrastructure metrics;
- experimental validation based on a Proxmox VE and NetEm testbed with OpenDaylight, Spring Boot, Node.js/MQTT, and a Monitoring VM that generated timestamped probe logs;
- cybersecurity-oriented interpretation of component-wise degradation patterns under 10%, 50%, and 100% stress profiles.

2. Related work

The analytical basis of this paper is formed by studies on software quality assessment, cybersecurity metrics, and decision support in protected information and communication environments. A review of methods and models for software quality assessment in ICS was presented in [1]. It identified limitations that arise when general software quality models are applied to distributed communication systems. A focused analysis of cybersecurity metrics for software quality assessment was provided in [2]; that study is directly relevant because it motivates the transition from generic quality indicators to security-oriented measurement. The method for assessing the quality of 5G core software proposed in [3] shows that software quality assessment should take into account domain-specific telecommunication requirements. The study on organizational and technical cybersecurity support using a virtual assistant [4] complements this context by emphasizing the role of monitoring and decision-support tools in cyber hygiene processes.

The communication-network context is also important. Perez et al. proposed a parametric quality model for tele-operated driving services over 5G networks and showed how QoS indicators can be connected with service performance [5]. Bojovic et al. examined dynamic QoS management for a flexible 5G/6G network core and highlighted programmability as a prerequisite for managing network

quality [6]. Although these studies focus mainly on QoS and service performance, they are relevant to cybersecurity-oriented software quality because many security-control functions in modern ICS depend on the same operational properties: latency, throughput, programmability, and stability.

Dynamic software metrics provide another important research direction. Tahir and MacDonell presented a systematic mapping study on dynamic metrics and software quality [7]. Kuliainin surveyed dynamic analysis methods and discussed their role in understanding software behavior during execution [8]. Schnoor and Hasselbring compared static and dynamic weighted software coupling metrics and showed that runtime information can complement static structural analysis [9]. These works support the general argument that software quality should not be evaluated exclusively at design time when runtime behavior is essential for system performance and reliability.

Decision-support approaches also influence software quality and cybersecurity assessment. Iqbal and Babar proposed a fuzzy-logic-based method for analyzing ISO/IEC 25010 product quality requirements [10]. Shevchenko et al. used fuzzy cognitive mapping as a scenario approach for information security risk analysis [11]. Such approaches are relevant because the selection and weighting of cybersecurity metrics often require expert judgment, risk interpretation, and scenario analysis rather than purely mechanical aggregation.

Standardized quality and security frameworks provide the terminology and measurement context for this research. ISO/IEC 25010 defines software quality characteristics, while ISO/IEC 25022 and ISO/IEC 25023 support measurement of quality in use and product quality [12, 13, 14]. ISO/IEC 27001 and ISO/IEC 27002 define requirements and controls for information security management, while ISO/IEC 27004 addresses information security monitoring, measurement, analysis, and evaluation [15, 16, 17]. These standards do not provide a complete formula for cybersecurity-oriented software quality assessment in ICS, but they define a structured vocabulary and methodological constraints for selecting, interpreting, and documenting metrics.

Existing research therefore supports three conclusions. First, general software quality models provide a necessary but incomplete basis for ICS. Second, dynamic metrics are required when the assessed software operates under changing network and workload conditions. Third, cybersecurity assessment requires metrics that are connected with protection functions, observability, and operational resilience. The model proposed in this paper addresses these conclusions by combining static security-related quality measures with runtime cybersecurity metrics in a single assessment procedure.

3. Assessment problem and cybersecurity requirements

In an ICS environment, a software component may have acceptable static quality and still become a source of cybersecurity risk during operation [18, 19]. For example, an authentication service may be functionally complete, but its latency may increase under load and reduce the stability of access workflows. An SDN controller may expose a correct RESTCONF interface, but delayed control-plane interactions may slow security-policy updates. An IoT telemetry component may generate valid messages, but packet loss may reduce the completeness of monitoring data used for incident analysis.

The assessment problem is therefore to connect product-level security properties with runtime evidence. The model must answer four questions:

1. Which static security-relevant quality characteristics are implemented and measurable?
2. How does the software behave when the infrastructure is exposed to controlled stress profiles?
3. Which runtime metrics are most relevant to the cybersecurity role of each component?
4. How should static and runtime evidence be aggregated into a single interpretable quality score?

The proposed model is based on the cybersecurity requirements summarized in Table 1. These requirements are not introduced as a new standard. They are a structured interpretation of the security-relevant properties needed for the assessed software components.

This formulation differs from a purely performance-oriented experiment. The same measured values may have different significance when cybersecurity is the target. Packet loss in a general application

Table 1
Cybersecurity-Oriented Requirements and Corresponding Measurement Interpretation

Requirement area	Meaning for software quality assessment	Measurement interpretation
Access control	Protected functions must be accessible only through controlled mechanisms	Access controllability, authentication latency, authentication throughput
Security-policy enforcement	Network-control software must support timely policy and control-plane interactions	SDN RESTCONF/policy latency, control-plane throughput, packet-loss resistance
Telemetry integrity	Security telemetry should arrive with limited loss and acceptable delay	MQTT/IoT telemetry latency, throughput, packet-loss resistance
Accountability and observability	Security-relevant events must be measurable and visible to operators	Timestamped probe logs, CSV dataset completeness, monitoring coverage
Cyber resilience under stress	Components should maintain acceptable security-relevant behavior under degraded conditions	Runtime cybersecurity score and relative degradation

can be treated as a performance issue, while packet loss in a security telemetry stream may reduce the ability to reconstruct events. Similarly, latency in an authentication service may affect not only user experience but also the reliability of access-control workflows.

4. Cybersecurity-metric-driven assessment model

The proposed model develops the integrated static-dynamic quality assessment model introduced in [20] and adapts its mathematical representation to cybersecurity metrics. It evaluates the cybersecurity-oriented software quality of an ICS component through two components: static cybersecurity quality and runtime cybersecurity quality. The static component represents design-time and product-level security properties. The runtime component represents the operational behavior of security-relevant functions. The conceptual structure of the model is shown in Figure 1.

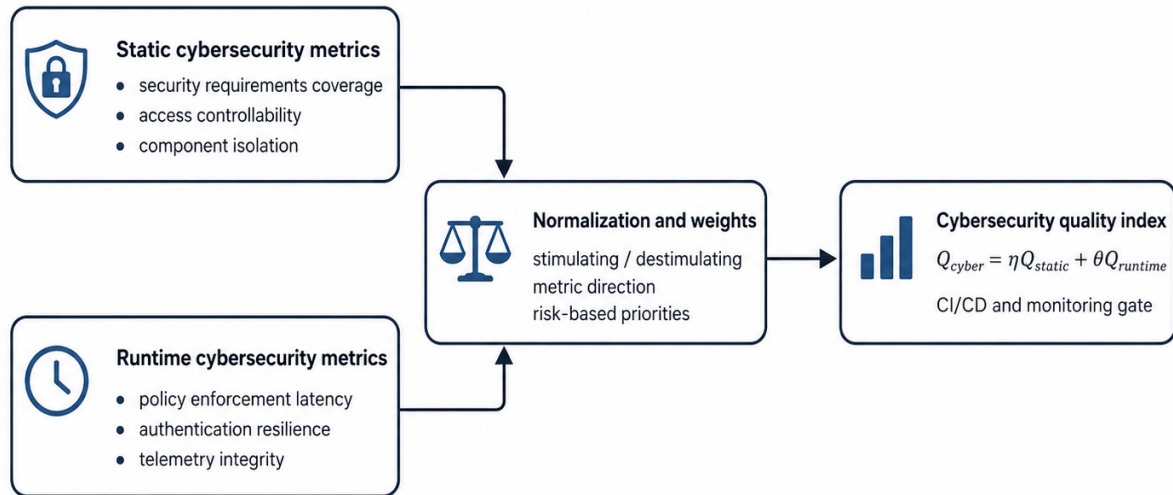


Figure 1: Cybersecurity-oriented interpretation of the integrated software quality assessment model.

The terminology of the static component is aligned with the ISO/IEC SQuaRE family, where ISO/IEC 25010 defines quality characteristics and ISO/IEC 25022 and ISO/IEC 25023 provide measurement-oriented support for quality in use and product quality [12, 13, 14]. The cybersecurity interpretation is additionally coordinated with ISO/IEC 27001 and ISO/IEC 27002, which define requirements for information security management systems and security controls, and with ISO/IEC 27004, which addresses information security monitoring, measurement, analysis, and evaluation [15, 16, 17]. In this paper, these standards are used as terminology and methodological guidance rather than as a ready-made calculation procedure.

4.1. Static cybersecurity quality

The static cybersecurity component is calculated from normalized metrics $c_1, c_2, \dots, c_n$, where each $c_i \in [0, 1]$ and a higher value indicates better quality. The general form is:

$$Q_{sec-static} = \sum_{i=1}^n a_i c_i, \quad \sum_{i=1}^n a_i = 1, \quad a_i \geq 0. \quad (1)$$

Here, a_i is the weight of the i -th static cybersecurity metric. The metric set can be extended for a particular organization or deployment context. The demonstration in this paper uses four metrics based on the integrated model in [20], the SQuaRE logic, and cybersecurity-control interpretation: security requirements coverage, access controllability, component isolation, and secure adaptability.

The numerical values of c_1, c_2, c_3, c_4 used in this subsection should be interpreted as an illustrative reference static cybersecurity profile rather than as the result of a separate source-code audit, configuration audit, penetration test, or certification procedure. They are introduced to demonstrate how the proposed aggregation model combines a predefined static cybersecurity baseline with runtime measurements obtained from the experiment. The experimentally obtained part of the assessment is represented by the runtime metrics derived from timestamped CSV probe logs.

The security-oriented interpretation treats functional coverage as coverage of security-critical functions, component coupling as isolation and modularity relevant to containment, and adaptability as the ability to preserve secure deployment across hardware or virtualized environments. The access-control metric remains directly connected with confidentiality and the control of protected functions.

Table 2
Static Cybersecurity-Oriented Metrics Used in the Model Demonstration

Metric group	Interpretation in cybersecurity-oriented assessment	Symbol	Value	Weight
Security requirements coverage	Coverage of security-critical functions and required controls	c_1	0.80	0.30
Access controllability	Ability to control access to protected software functions	c_2	0.75	0.30
Component isolation	Modularity and reduced coupling as a containment factor	c_3	0.75	0.25
Secure adaptability	Ability to preserve secure behavior in different virtualized environments	c_4	0.70	0.15

The static cybersecurity score is therefore:

$$Q_{sec-static} = 0.30 \cdot 0.80 + 0.30 \cdot 0.75 + 0.25 \cdot 0.75 + 0.15 \cdot 0.70 = 0.7575. \quad (2)$$

The scores are visualized in Figure 2. The values are not introduced as new empirical static-audit results; they define a cautious scenario-based static profile used to demonstrate the aggregation stage of the model.

4.2. Runtime cybersecurity quality

Runtime cybersecurity quality is calculated from operational metrics collected under stress profiles. The model distinguishes between stimulating metrics, where larger values are better, and destimulating metrics, where smaller values are better. For a stimulating metric x , normalization is performed as:

$$S^+(x) = \frac{x - x_{min}}{x_{max} - x_{min}}. \quad (3)$$

For a destimulating metric x , normalization is performed as:

$$S^-(x) = 1 - \frac{x - x_{min}}{x_{max} - x_{min}}. \quad (4)$$

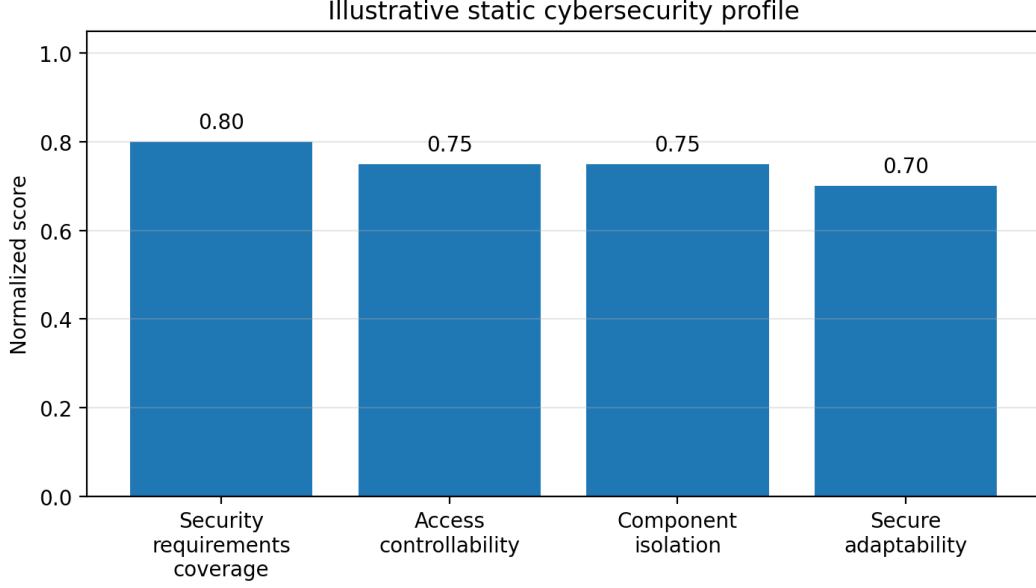


Figure 2: Static cybersecurity-oriented metric scores used in the model demonstration.

The runtime cybersecurity score is:

$$Q_{sec-runtime} = \sum_{j=1}^m b_j r_j, \quad \sum_{j=1}^m b_j = 1, \quad b_j \geq 0, \quad (5)$$

where r_j is a normalized runtime metric and b_j is its cybersecurity importance weight.

For the real experimental assessment, three normalized metric groups were used: latency resistance, throughput sustainability, and security-response or telemetry-delivery correctness. This modification is important because the experiment evaluates not only performance degradation but also whether a component preserves the expected security-relevant response. For the SDN controller, the expected result of the unauthorized RESTCONF scenario is HTTP 401. For the authentication service, the expected availability response of the tested endpoint is HTTP 200. For the IoT component, the expected result is successful MQTT publication.

The cybersecurity meaning of the metric groups depends on the component:

- for the SDN controller, latency resistance reflects the responsiveness of RESTCONF access-control enforcement, while the expected 401 response rate reflects stable rejection of unauthorized requests;
- for the authentication service, latency and throughput reflect the stability of the access-control workflow, while the expected 200 response rate reflects service availability during the experiment;
- for the IoT component, delivery correctness has the highest influence because telemetry loss can reduce monitoring completeness and incident-reconstruction capability.

The component-specific weights are shown in Table 3. They were selected for the model demonstration to reflect the cybersecurity role of each component. In practical deployment, they should be determined through expert evaluation, risk analysis, or sensitivity analysis.

4.3. Integral cybersecurity-oriented quality index

The final cybersecurity-oriented quality index is calculated as:

$$Q_{cyber} = \eta Q_{sec-static} + \theta Q_{sec-runtime}, \quad \eta + \theta = 1. \quad (6)$$

Table 3
Component-Specific Weights for Runtime Cybersecurity Metrics

Component	Latency resistance	Throughput sustainability	Response or delivery correctness	Cybersecurity rationale
SDN RESTCONF access-control	0.50	0.25	0.25	Unauthorized access should be rejected consistently and without excessive control-plane delay
Authentication service	0.45	0.35	0.20	Authentication-related workflows should remain responsive and available under stress
IoT telemetry	0.25	0.20	0.55	Security telemetry should be delivered reliably because loss of events reduces observability

Here, η is the influence of static cybersecurity quality, and θ is the influence of runtime cybersecurity quality. In a low-risk profile, η may be higher. In a critical profile, θ should increase because runtime degradation has a stronger influence on cyber resilience.

The adaptive weighting logic is:

$$\theta^* = \min(\theta + \Delta_{risk}, \theta_{max}), \quad \eta^* = 1 - \theta^*, \quad (7)$$

where Δ_{risk} is activated when a runtime metric crosses an unacceptable threshold. For example, packet loss above a telemetry threshold or a sharp increase in authentication latency can increase the influence of runtime cybersecurity quality.

The model can also be written in expanded form:

$$Q_{cyber} = \eta \sum_{i=1}^n a_i c_i + \theta \sum_{j=1}^m b_j r_j. \quad (8)$$

This expression makes the model transparent: the final score depends on static cybersecurity quality, runtime cyber resilience, metric normalization, and risk-based weights.

5. Experimental design

The experiment was conducted in a Proxmox VE-based testbed and structured around cybersecurity-oriented runtime metrics. The validation uses controlled infrastructure stress profiles and does not claim to reproduce the full complexity of real cyber incidents. Its purpose is to assess how security-relevant software behavior changes when the network and service environment moves from a baseline state to stressed and critical conditions.

The testbed contains five virtual machines:

1. **RouterVM**: a gateway node with NetEm for emulating latency, packet loss, and throughput constraints.
2. **SDN VM**: an OpenDaylight controller in a Karaf environment, exposing RESTCONF and representing software responsible for network control and policy-related interactions.
3. **Auth VM**: a Java Spring Boot authentication microservice with Actuator, representing a security-critical access workflow.
4. **IoT VM**: an MQTT-based telemetry component using Mosquitto, representing security telemetry and event-stream processing.

5. **Monitoring VM:** a measurement and observability node used to execute scripted probes and collect timestamped CSV results. Prometheus and Grafana may be used as an auxiliary observability layer, but they were not the primary source of the numerical values reported in this paper.

The cybersecurity-oriented testbed interpretation is shown in Figure 3. Latency, throughput, and response correctness are mapped to cybersecurity meanings: access-control enforcement, authentication-service availability, and telemetry delivery integrity.

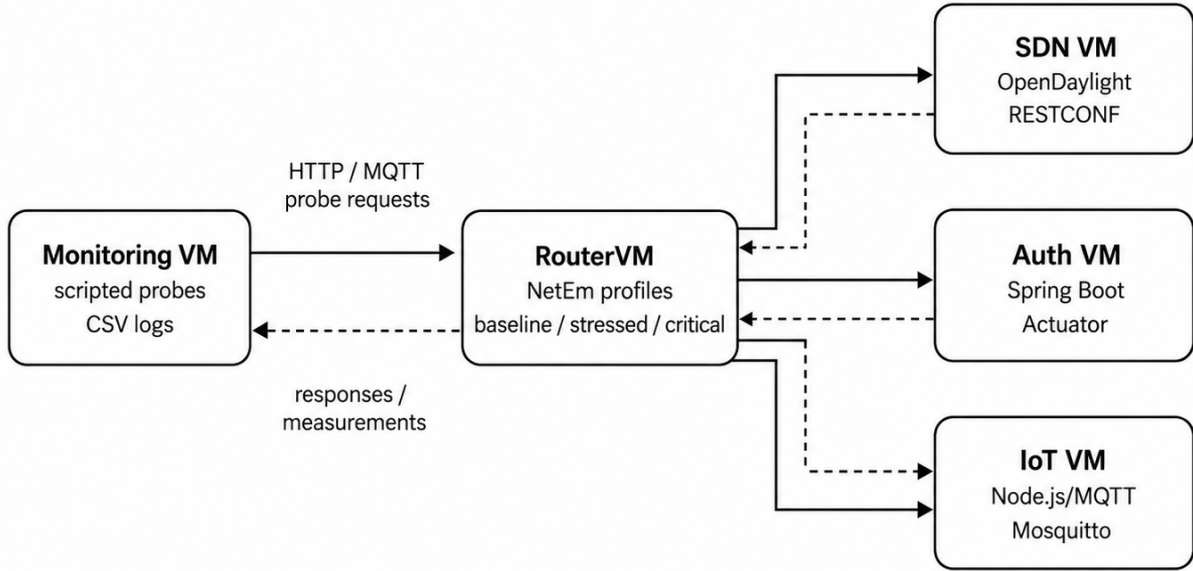


Figure 3: Cybersecurity-oriented interpretation of the Proxmox VE experimental testbed.

Three stress profiles were used:

- **baseline profile:** normal cyber hygiene state with low network degradation and stable service behavior;
- **stressed profile:** elevated cyber hygiene stress with increased latency, moderate packet loss, and lower effective throughput;
- **critical profile:** critical stress profile with significant network degradation and visible risk to security-relevant behavior.

The experiment uses the existing infrastructure as the basis, but the measurement procedure is adapted to the cybersecurity-oriented version of the model. The SDN controller was evaluated through unauthorized RESTCONF requests. Therefore, a stable HTTP 401 response was interpreted as correct access-control enforcement, while response time and request rate were used as runtime indicators. The authentication service was evaluated through the Spring Boot Actuator health endpoint, where HTTP 200 was treated as the expected availability response. The IoT telemetry component was evaluated through MQTT publication, where status code 0 from the publisher was treated as successful delivery to the broker.

The primary experimental dataset was collected as timestamped CSV logs generated by scripted probes executed from the Monitoring VM. Prometheus and Grafana were retained as an optional observability layer of the testbed, but the quantitative values reported in this paper are based on the exported probe logs. The raw measurements are summarized in Table 4. Each profile was executed as a separate run. HTTP-based scenarios lasted approximately 299–300 seconds, while the IoT scenario used 1000 MQTT publications per profile. The table reports measured values from the generated CSV files.

Table 4
Real Runtime Measurements Used for Cybersecurity-Oriented Aggregation

Component	Profile	Count	Avg. latency, ms	Throughput	Expected rate
SDN RESTCONF access-control	Baseline	563	22.277	1.877 req/s	1.000
	Stressed	528	55.810	1.766 req/s	1.000
	Critical	481	112.657	1.609 req/s	1.000
Authentication service	Baseline	1277	22.919	4.271 req/s	1.000
	Stressed	1100	60.536	3.679 req/s	1.000
	Critical	886	126.947	2.963 req/s	1.000
IoT telemetry	Baseline	1000	37.711	10.982 msg/s	1.000
	Stressed	1000	94.524	6.767 msg/s	1.000
	Critical	1000	173.115	4.418 msg/s	1.000

6. Results

Runtime cybersecurity scores were calculated by normalizing the measured latency and throughput values across the observed profile range for each component. Latency was treated as a destimulating metric, while throughput was treated as a stimulating metric. The expected-response or delivery rate was used directly because it already belongs to the interval $[0, 1]$. The component-specific weights from Table 3 were then applied.

Table 5 presents the normalized components of the runtime score. The values show that all three components preserved the expected security-relevant response during the experiment. The SDN controller returned HTTP 401 for unauthorized RESTCONF access attempts in all measured requests; the authentication endpoint returned HTTP 200 in all measured requests; and MQTT publication succeeded for all 1000 messages in every profile. At the same time, latency resistance and throughput sustainability decreased as stress increased.

Table 5
Normalized Runtime Cybersecurity Metrics Derived from Real Measurements

Component	Profile	Latency resistance	Throughput sustainability	Correct response/delivery	$Q_{sec-runtime}$
SDN RESTCONF access-control	Baseline	1.000	1.000	1.000	1.000
	Stressed	0.629	0.587	1.000	0.711
	Critical	0.000	0.000	1.000	0.250
Authentication service	Baseline	1.000	1.000	1.000	1.000
	Stressed	0.638	0.547	1.000	0.679
	Critical	0.000	0.000	1.000	0.200
IoT telemetry	Baseline	1.000	1.000	1.000	1.000
	Stressed	0.580	0.358	1.000	0.767
	Critical	0.000	0.000	1.000	0.550

Table 6 summarizes the resulting runtime cybersecurity scores. The largest relative decrease was observed for the authentication service, whose runtime score decreased by 80.0% from baseline to the critical profile. The SDN RESTCONF access-control scenario decreased by 75.0%, while the IoT telemetry component decreased by 45.0%. This result reflects the fact that telemetry delivery remained correct in all profiles, even though publication time increased and throughput decreased.

Table 6
Runtime Cybersecurity Scores Based on the Real Experiment

Component	Baseline	Stressed	Critical	Relative decrease from baseline to critical
SDN RESTCONF access-control	1.000	0.711	0.250	75.0%
Authentication service	1.000	0.679	0.200	80.0%
IoT telemetry	1.000	0.767	0.550	45.0%

The runtime cybersecurity degradation is shown in Figure 4. The graph indicates that all components degrade under stress, but the degradation pattern differs. The authentication service and the SDN

access-control scenario are more sensitive to the critical profile because their normalized latency and throughput values reach the lowest observed level. The IoT component remains higher in the final score because MQTT publication succeeded in all profiles, although its average publish time increased from 37.711 ms to 173.115 ms and throughput decreased from 10.982 to 4.418 messages per second.

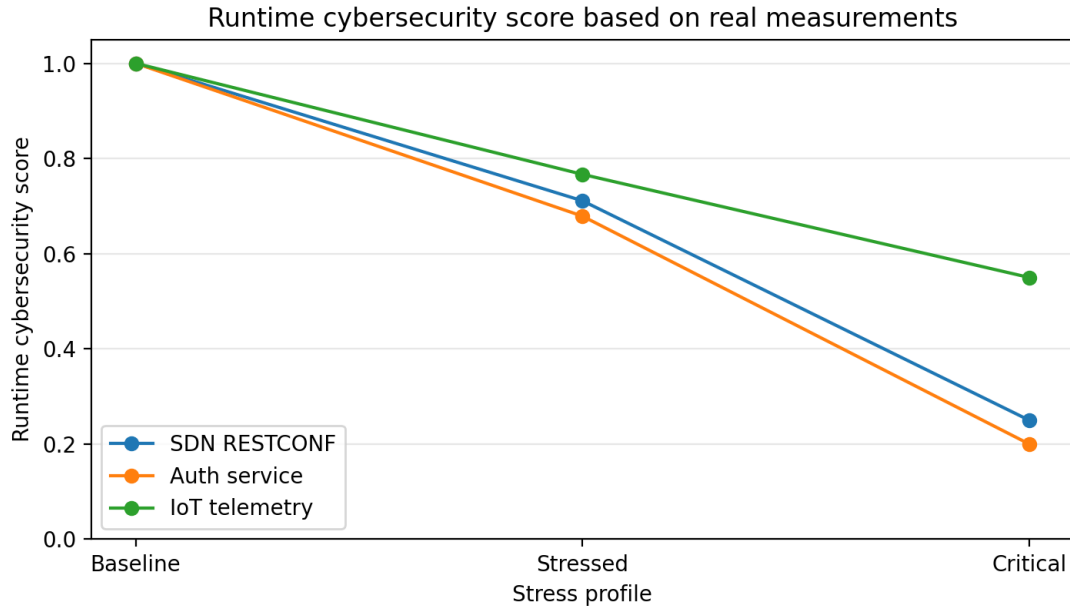


Figure 4: Runtime cybersecurity quality degradation calculated from real experimental measurements.

Figure 5 provides a heatmap representation of the same results. It makes the component-wise risk pattern more visible. The critical profile becomes a risk zone for all components, but the meaning of the risk differs. For SDN, the main issue is increased access-control response latency and lower request rate. For the authentication service, the main issue is reduced responsiveness of the availability endpoint. For IoT telemetry, delivery integrity remains stable, but message publication becomes slower.

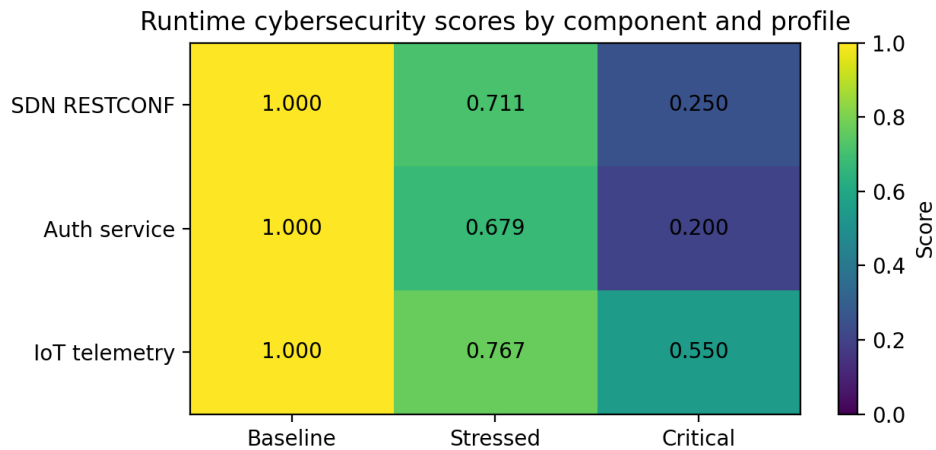


Figure 5: Heatmap of runtime cybersecurity scores for the assessed ICS components.

For the stressed profile, an integral cybersecurity-oriented quality score was calculated using $\eta = 0.45$ and $\theta = 0.55$. The runtime component receives a slightly higher weight because the model is intended to expose cyber-resilience changes under operational stress. The illustrative static cybersecurity score is $Q_{sec-static} = 0.7575$. The integral scores are shown in Table 7 and Figure 6.

Table 7
Integral Cybersecurity-Oriented Quality Scores for the Stressed Profile

Component	$Q_{sec-static}$	$Q_{sec-runtime}$ under stress	Q_{cyber}
SDN RESTCONF access-control	0.7575	0.711	0.732
Authentication service	0.7575	0.679	0.714
IoT telemetry	0.7575	0.767	0.763

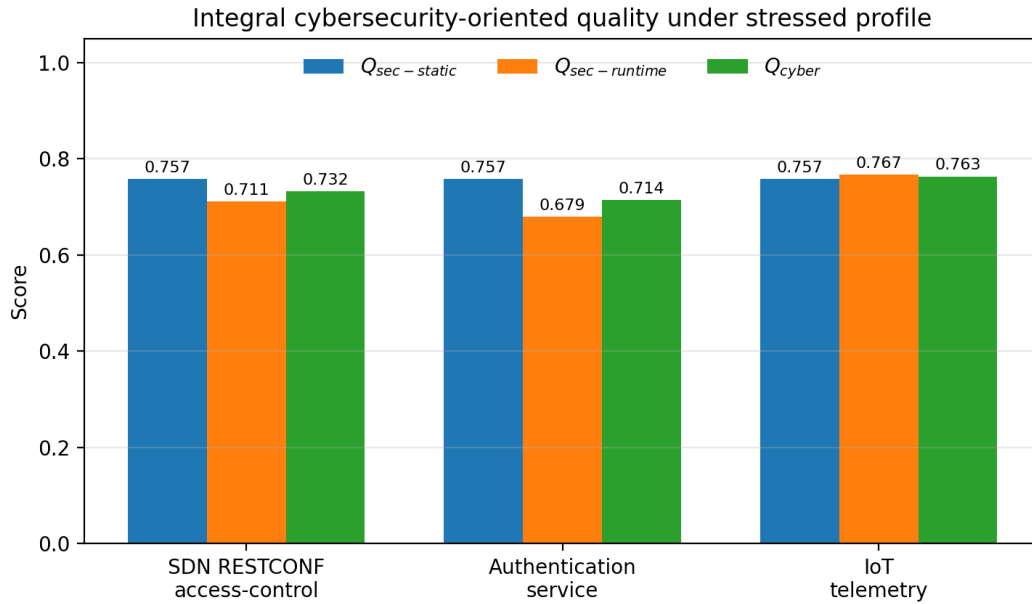


Figure 6: Integral cybersecurity-oriented quality scores under the stressed profile.

The results show why a cybersecurity-oriented quality model should preserve both correctness and degradation dimensions. If only the expected-response rate is considered, all three components appear equally stable because the rates are 1.000 in every profile. If only latency or throughput is considered, the security meaning of the response is lost. The proposed score combines these dimensions and therefore shows that the components preserve correct security-relevant behavior while still experiencing operational degradation.

7. Discussion

The experiment shows that cybersecurity-oriented assessment should also consider whether the component preserves the expected security response under stress. In the SDN case, HTTP 401 is not an error of the experiment but the correct result of unauthorized RESTCONF access rejection. In the authentication case, HTTP 200 from the health endpoint indicates availability of the access-service component. In the IoT case, publish status 0 indicates successful delivery to the broker.

At the same time, correct responses do not eliminate operational risk. The authentication service preserved a 100% expected response rate, but its average response time increased from 22.919 ms to 126.947 ms and throughput decreased from 4.271 to 2.963 requests per second. The SDN RESTCONF access-control scenario also preserved the expected 401 response in all requests, but average latency increased from 22.277 ms to 112.657 ms. These changes are relevant for cyber hygiene because delayed security-control interactions can reduce the timeliness of operational decisions.

The IoT telemetry component demonstrated a different pattern. MQTT publication succeeded in all measured cases, but the average publish time increased from 37.711 ms to 173.115 ms and throughput decreased from 10.982 to 4.418 messages per second. Therefore, the component preserved delivery correctness but became slower. In a production monitoring environment, this could still matter because

delayed telemetry may reduce the timeliness of detection and incident analysis.

The integral score for the stressed profile shows that the IoT component has the highest value in this particular experiment because telemetry delivery correctness received the largest weight and remained stable. This result should not be interpreted as a universal statement that IoT telemetry is always the strongest component. It reflects the measured behavior of the current testbed and the selected weights. In another deployment, with message loss or broker overload, the same model could produce a different ranking.

7.1. Use in CI/CD and cyber hygiene monitoring

The proposed model can be integrated into a CI/CD pipeline in the following sequence. First, static cybersecurity metrics are collected from requirements, architecture review, source-code analysis, configuration review, or security test artifacts. Second, the component is deployed in a staging environment with predefined cyber hygiene stress profiles. Third, runtime metrics are collected from scripted HTTP probes, MQTT publication tests, exported logs, Prometheus exporters, or other observability sources, depending on the deployment. In the present experiment, timestamped CSV probe logs were used as the primary evidence. Fourth, metrics are normalized and aggregated into $Q_{sec-runtime}$ and Q_{cyber} . Fifth, the release is accepted, rejected, or marked for mitigation according to predefined thresholds.

In operational monitoring, the same score can be calculated periodically. This allows engineers to observe not only isolated metrics but also a composite cybersecurity-oriented quality trend. A persistent decrease in Q_{cyber} may indicate that the component is approaching a state where it can no longer support protection requirements. Such an approach is compatible with observability dashboards because it does not replace raw metrics; it adds a decision-oriented layer above them.

7.2. Limitations

The study has several limitations. The experiment is based on a virtualized testbed and does not reproduce the scale or diversity of production 5G, SDN, or IoT infrastructures. The runtime cybersecurity scores are derived from controlled stress measurements rather than from a full attack dataset. Therefore, the results should be interpreted as evidence of security-relevant behavior under controlled degradation, not as proof of resistance to cyberattacks.

The static metric set is also limited. A complete cybersecurity assessment could include vulnerability density, patch latency, secure-configuration compliance, cryptographic configuration, audit logging completeness, incident detection coverage, and recovery time. These metrics were not added as factual experimental values because the available experimental dataset does not provide the corresponding measurements. Instead, the paper defines how such metrics can be incorporated into the model.

The static values and weights used in the integral calculation are demonstrative and should be validated for a particular deployment. They do not represent a completed static cybersecurity audit of the tested components. In a real critical infrastructure, static scores and weights should be selected through requirements analysis, configuration review, expert evaluation, risk analysis, or sensitivity analysis. Scenario-oriented approaches, including fuzzy cognitive mapping for information security risk analysis, may be used to formalize this stage in future work. Future studies should also compare the model with additional stress profiles, authorized RESTCONF operations, service-level thresholds, and controlled security-event datasets.

8. Conclusions

This paper presented a cybersecurity-metric-driven model for software quality assessment in critical ICS. The model develops the integrated static-dynamic assessment approach and makes cybersecurity metrics the central assessment dimension. The static component is aligned with the ISO/IEC SQuaRE family and includes security-relevant product metrics. The runtime component evaluates cyber resilience

through access-control response stability, authentication-service responsiveness, telemetry delivery integrity, throughput sustainability, and latency resistance.

The experiment used a Proxmox VE testbed with NetEm, OpenDaylight, Spring Boot, MQTT, and a Monitoring VM. Real measurements were collected from timestamped CSV probe logs for three profiles: baseline, stressed, and critical. The SDN RESTCONF scenario preserved a 100% expected 401 response rate, the authentication service preserved a 100% expected 200 response rate, and the IoT telemetry component preserved a 100% MQTT publish-success rate. However, the measured runtime characteristics degraded under stress. The normalized runtime cybersecurity score decreased from 1.000 to 0.250 for the SDN RESTCONF access-control scenario, from 1.000 to 0.200 for the authentication service, and from 1.000 to 0.550 for the IoT telemetry component.

The results indicate that cybersecurity-oriented software quality assessment provides more useful evidence for release decisions and monitoring than a static-only or generic performance-oriented model. The approach can support CI/CD quality gates, cyber hygiene dashboards, SLA-risk control, and prioritization of mitigation actions. Future research should extend the metric set with vulnerability, patching, secure configuration, audit logging, detection, and recovery metrics and validate the model using larger-scale infrastructure, authorized control-plane operations, and real security-event datasets.

Acknowledgements

The authors acknowledge the academic and technical environment that supported the preparation and experimental validation of this study. No specific external funding is declared for this work.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

- [1] R. M. Hamretskyi, V. O. Gnatyuk, Review and Analysis of Methods and Models for Software Quality Assessment in Information and Communication Systems, *Science-Based Technologies* (2024) 435–448. doi:10.18372/2310-5461.63.19752.
- [2] R. Hamretskyi, V. Gnatyuk, Analysis of Cybersecurity Metrics for Software Quality Assessment, in: S. Semenov, M. Muchacki (Eds.), *Problems of Scientific, Technical and Legal Support for Cybersecurity in the Modern World*, Wydawnictwo Naukowe UKEN, Krakow, 2024, pp. 126–136.
- [3] R. M. Hamretskyi, V. O. Gnatyuk, Method for Assessing the Quality of 5G Core Network Software, *Herald of Khmelnytskyi National University. Technical Sciences* (2025) 106–111. doi:10.31891/2307-5732-2025-355-15.
- [4] V. Gnatyuk, O. Batrak, R. Hamretskyi, M. Golovan, Organizational and Technical Support of Cyber Security Using a Virtual Assistant, in: *Cyber Hygiene & Conflict Management in Global Information Networks 2024 (CH&CMiGIN 2024)*, volume 3925 of *CEUR Workshop Proceedings*, Kyiv, Ukraine, 2025, pp. 317–324. URL: <https://ceur-ws.org/Vol-3925/short08.pdf>. doi:10.5281/zenodo.19119346.
- [5] P. Perez, J. Ruiz, I. Benito, et al., A Parametric Quality Model to Evaluate the Performance of Tele-operated Driving Services over 5G Networks, *Multimedia Tools and Applications* 81 (2022) 12287–12303. doi:10.1007/s11042-021-11251-x.
- [6] P. D. Bojovic, T. Malbasic, D. Vujosevic, G. Martic, Z. Bojovic, Dynamic QoS Management for a Flexible 5G/6G Network Core: A Step toward a Higher Programmability, *Sensors* 22 (2022) 2849. doi:10.3390/s22082849.
- [7] A. Tahir, S. G. MacDonell, A Systematic Mapping Study on Dynamic Metrics and Software Quality, 2021. URL: <https://arxiv.org/abs/2101.03669>. arXiv:2101.03669.

- [8] V. V. Kuliainin, A Survey of Software Dynamic Analysis Methods, *Programming and Computer Software* 50 (2024) 90–114. doi:10.1134/S0361768824010079.
- [9] H. Schnoor, W. Hasselbring, Comparing Static and Dynamic Weighted Software Coupling Metrics, *Computers* 9 (2020) 24. doi:10.3390/computers9020024.
- [10] H. Iqbal, M. Babar, An Approach for Analyzing ISO/IEC 25010 Product Quality Requirements Based on Fuzzy Logic and Likert Scale for Decision Support Systems, *International Journal of Advanced Computer Science and Applications* 7 (2016). doi:10.14569/IJACSA.2016.071232.
- [11] S. Shevchenko, Y. Zhdanova, O. Kryvytska, H. Shevchenko, Fuzzy Cognitive Mapping as a Scenario Approach for Information Security Risk Analysis, in: *Cybersecurity Providing in Information and Telecommunication Systems II*, volume 3826 of *CEUR Workshop Proceedings*, 2024, pp. 356–362. URL: <https://ceur-ws.org/Vol-3826/short28.pdf>.
- [12] ISO/IEC, ISO/IEC 25010:2023, Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (SQuaRE) – Product Quality Model, 2023. URL: <https://www.iso.org/standard/78176.html>.
- [13] ISO/IEC, ISO/IEC 25022:2016, Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (SQuaRE) – Measurement of Quality in Use, 2016. URL: <https://www.iso.org/standard/35746.html>.
- [14] ISO/IEC, ISO/IEC 25023:2016, Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (SQuaRE) – Measurement of System and Software Product Quality, 2016. URL: <https://www.iso.org/standard/35747.html>.
- [15] ISO/IEC, ISO/IEC 27001:2022, Information Security, Cybersecurity and Privacy Protection – Information Security Management Systems – Requirements, 2022.
- [16] ISO/IEC, ISO/IEC 27002:2022, Information Security, Cybersecurity and Privacy Protection – Information Security Controls, 2022.
- [17] ISO/IEC, ISO/IEC 27004:2016, Information Technology – Security Techniques – Information Security Management – Monitoring, Measurement, Analysis and Evaluation, 2016. URL: <https://www.iso.org/standard/64120.html>.
- [18] O. Holubnychyi, et al., Self-organization technique with a norm transformation based filtering for sustainable infocommunications within CNS/ATM systems, in: I. Ostroumov, M. Zaliskyi (Eds.), *Proceedings of the 2nd International Workshop on Advances in Civil Aviation Systems Development*, volume 992 of *Lecture Notes in Networks and Systems*, Springer Nature Switzerland, Cham, 2024, pp. 262–278. doi:10.1007/978-3-031-60196-5_20.
- [19] O. Solomentsev, M. Zaliskyi, O. Kozhokhina, T. Herasymenko, Efficiency of data processing for uav operation system, in: *2017 IEEE 4th International Conference Actual Problems of Unmanned Aerial Vehicles Developments (APUAVD)*, 2017, pp. 27–31. doi:10.1109/APUAVD.2017.8308769.
- [20] V. O. Gnatyuk, R. M. Hamretskyi, A Comprehensive Model for Software Quality Assessment in Information and Communication Systems Considering Dynamic Parameters, *Science-Based Technologies* (2026) 109–117. doi:10.18372/2310-5461.69.20951.