Section 4.2. Operational Mechanisms and Environment Models of the Multi-Agent System

Karyna Khorolska[1], Bohdan Bebeshko[2]
[1]Ph.D. (Computer Science), Associate Professor at the Department of Information and Cyber Security named after professor Volodymyr Buriachok, Borys Grinchenko Kyiv Metropolitan University, Kyiv, Ukraine, ORCID: https://orcid.org/0000-0003-3270-4494
[2]Ph.D. (Computer Science), Associate Professor at the Department of Information and Cyber Security named after professor Volodymyr Buriachok, Borys Grinchenko Kyiv Metropolitan University, Kyiv, Ukraine, ORCID: https://orcid.org/0000-0001-6599-0808

Citation:
Khorolska, K. & Bebeshko, B. (2025). Operational Mechanisms and Environment Models of the Multi-Agent System. In P. Kolisnichenko (Ed.), *Insider threats and security in corporations.* 274 p. (pp. 233–257). Scientific Center of Innovative Research. https://doi.org/10.36690/ITSC-233-257

Abstract. *Enterprise information systems are exposed to persistent and fast-changing cyber threats, while their distributed structure makes single-point detection and response increasingly fragile. This chapter proposes an operational model for a multi-agent system (MAS) that distributes sensing, analysis, and counteraction across interconnected environments, including workstations, network segments, subnetworks, servers, and perimeter routers. Agent interactions are adaptively driven by threat levels inferred by neural networks, and communication is refined through load balancing and delay-minimisation principles to avoid congestion. The objective is to specify operational mechanisms and environment models that enable scalable, resilient, and intelligent cyber defence in enterprise infrastructures, with coordinated decisions that remain robust under advanced and persistent attacks. The MAS is defined as a set of agents and environments, with formalised interaction intensity, distribution constraints, and communication delays to control overhead and preserve responsiveness. Threat evaluation is grounded in neural outputs with adaptive thresholds, complemented by fuzzy logic and trust modelling to cope with uncertainty and unreliable signals. Collective decision-making is operationalised through voting procedures, including Condorcet and Nanson methods, with trust-aware weighting to mitigate cyclical preferences. The study delivers a hierarchical MAS architecture with multi-level message propagation and modular subsystems for sensing, analysis, trust evaluation, and secure communication, coordinated by a central control component. Simulation in OMNET++ under realistic scenarios, including DDoS, indicates reduced response time, improved accuracy, and productive cooperation among specialised agents. A formally structured, trust-aware MAS can strengthen enterprise cyber defence by combining distributed monitoring with coordinated, adaptive responses, while maintaining communication efficiency and operational scalability across heterogeneous infrastructure layers.*

Keywords: *multi-agent system; enterprise cybersecurity; environment modelling; attack detection; incident response; adaptive thresholds; neural networks; fuzzy logic; trust modelling; Condorcet voting; Nanson method; OMNET++ simulation.*

ISBN 978-9916-9320-4-9 (pdf)
© Scientific Center of Innovative Research, 2025

1. Agent Action Selection, Threat Detection and Adaptive Response. Depending on the level of danger assigned by the neural network to the analyzed event, agents can perform various actions within a multi-agent attack detection and counteraction system (Logesh et al., 2023). These actions include recording the event in the system log, informing the administrator, blocking or interrupting the relevant process, and disconnecting the connection (Taher et al., 2019). Such responses enable agents to adapt their strategies to emerging threats (Javaid et al., 2016), ensuring prompt reaction to potential attacks and minimizing adverse consequences. A multi-agent attack detection and counteraction system is organized as a set of environments comprising various agents and their interactions. The system is formally defined as MAS = (A, M), where MAS denotes the multi-agent system, A denotes the set of agents, and M denotes the set of environments. Since each agent receives information from only one source, its perception is inherently limited; thus, the environment imposes constraints on interactions between agents (Samoilenko et al., 2024). Considering the potentially large number of agents, their interactions can impose substantial network load, necessitating a carefully designed and optimized communication structure (Kostiuk & Dovzhenko et al., 2025). The architecture facilitates distribution of functional responsibilities among agents based on their specialization and location, enabling effective coordination in response to detected threats. Each environment serves as a distinct monitoring and response zone, wherein agents exchange information to enhance attack detection accuracy. Optimizing agent interactions reduces transmission delays and alleviates network infrastructure load (Roshan et al., 2023; Kostiuk & Skladannyi et al., 2025; Kostiuk & Zhyltsov et al., 2025; Kostiuk & Dovzhenko et al., 2025).

As agents operate within the set of environments M, their behavior can be characterized by an interaction function. Accordingly, the formal representation of agent interactions within environments is defined as follows:

$$I(A_i, A_j, M_k) = f(A_i, A_j) \cdot g(M_k), \tag{4.54}$$

where $I(A_i, A_j, M_k)$ − the intensity of interaction between agents A_i and A_j in the environment M_k, $f(A_i, A_j)$ − the function of dependence between agents that determines how much information they can exchange, $g(M_k)$ − the function of the environment's influence on interaction.

This equation demonstrates that each environment can either enhance or restrict interactions between agents, necessitating appropriate control over the environment to prevent system overload (Logesh et al., 2023; Kostiuk & Rzaieva et al., 2025; Naseeer et al., 2018; Rzaieva et al., 2024).

To minimize the load on the system, agents should be distributed among the environments in an optimal way. This can be expressed as a load minimization problem L:

$$L = \sum_{k=1}^{|M|} \sum_{i=1}^{|A|} X_{ik} C_{ik}, \tag{4.55}$$

where X_{ik} − variable that takes the value 1 if agent A_i located in environment M_k and 0 otherwise, C_{ik} − the cost of communication between agent A_i and environment M_k.

For efficient operation, it is necessary to optimize the distribution of agents to reduce the cost of interaction and the overall load on the network.

Since the interaction of agents creates a communication load, an important parameter is the delay in data transmission between agents (Kostiuk & Vorokhob et al., 2025; Samoilenko et al., 2025; Wu et al., 2020; Ricciato & Fleischer, 2004):

$$D_{ij} = \frac{S}{B_{ij}} + T_{comp}, \tag{4.56}$$

where D_{ij} − the total delay in the exchange of information between agents A_i and A_j, S − the amount of information transmitted, B_{ij} − the bandwidth of the communication channel between agents, T_{comp} − e processing time of the received information.

This formula highlights the critical importance of balancing network bandwidth with the efficient allocation of computational resources, as these factors directly influence the performance of a multi-agent system (Wu et al., 2020).

Since each environment imposes its own restrictions on the interaction of agents, their adaptation to changes can be expressed through the function of dynamic parameter updating:

$$A_i^{(t+1)} = A_i^{(t)} + \lambda \sum_{j \in N_i} w_{ij}(A_j^{(t)} - A_i^{(t)}) + \mu \cdot F(M_k), \tag{4.57}$$

where $A_i^{(t)}$ − the current state of agent A_i at the time t, λ − the coefficient of adaptation to neighboring agents, N_i − the set of neighboring agents, w_{ij} − the weighting factor of agent interaction, μ − the coefficient of environmental influence, $F(M_k)$ − the agent's adaptation function to the environment M_k.

This equation enables agents to modify their behavior based on interactions with other agents and the influence exerted by the environment, thereby enhancing the overall system's resilience to change (Kostiuk & Rzaieva et al., 2025).

To assess the effectiveness of the entire system, you can use the coefficient of coherence of agent communication C_{eff}:

$$C_{eff} = \frac{1}{|A|}\sum_{i=1}^{|A|}\sum_{j\in N_i}\frac{I(A_i,A_j,M_k)}{D_{ij}}, \tag{4.58}$$

where the numerator - is the total intensity of interaction between agents, and the denominator - is the total communication delay.

The higher the value of C_{eff}, the more efficiently agents interact with each other and the faster they exchange information, which is critical for the system's rapid response to threats or changes in the environment (Naseer et al., 2018).

The integration of modern artificial intelligence methods into multi-agent systems can significantly enhance their adaptability and resilience against attacks. Machine learning techniques predict potential threats and dynamically adjust agent parameters in real time (Skladannyi et al., 2025). Consequently, multi-agent systems acquire the capacity to respond effectively to known threats and anticipate emerging attack types, optimizing their configuration and response strategies (Rzaieva et al., 2024; Skladannyi et al., 2025). However, increasing agent intelligence alone is insufficient. It is essential to integrate a comprehensive defense framework capable of countering both isolated and coordinated attacks, employing traffic analysis algorithms, distributed intrusion detection systems, behavioral anomaly models, and mechanisms that compare threat indicators with continuously updated databases.

Moreover, optimizing the architecture is crucial, encompassing load balancing, fault tolerance, and efficient resource management. Distributed computing and cloud technologies can markedly improve performance and scalability (Ricciato & Fleischer, 2018; Kostiuk & Sokolov et al., 2025; Ma et al., 2009). Additionally, platforms such as Node-RED serve as valuable tools for real-time visualization of agent interactions via the Modbus protocol (Logesh et al., 2023; Kostiuk & Zhyltsov et al., 2025; Kostiuk & Kriuchkova et al., 2024).

2. Environment Models of the Multi-Agent System. The following environment models are formulated:

- Environments $M_w \subset A_w$, consisting of a subset of workstation agents. Workstation agents $A_{WI} = \{a_{WI}^{i}, \dots a_{WI}^{n}\}$ interact to make joint decisions regarding incidents. Workstation agent environment:

$$D_W = \sum_{i=1}^{n} w_i f(a_{WI}^i, C), \qquad (4.59)$$

where D_W − the overall decision of workstation agents regarding the incident, w_i − the weight of the influence of each agent, $f(a_{WI}^i, C)$ − the function of evaluating event C by agent a_{WI}^i.

The environment provides collective assessment and coordinated response to incidents.

- Environments $M_{NS} \subset \{A_W^1, A_N\}$ - subset of network segment agents. Network agent a_{Np} interacts with the workstation agent $a_{WI}^i\ \forall I \in p$, to provide a coordinated decision about possible threats at the network segment level. Network segment agent environment:

$$D_{NS} = \frac{1}{|P|} \sum_{p \in P} g(\alpha N_p, a_{WI}^i), \qquad (4.60)$$

where D_{NS} − decision on threats at the network segment level, P − set of network segments, $g(\alpha N_p, a_{WI}^i)$ − the function of interaction between network and workstation agents.

- Environments $M_N \subset \{A_R^v, A_N\}$ - subset of subnetwork agents. Router agent a_{Rt}^v interacts with network agents A_N to make security decisions at specific network segments. Subnetwork agent environment:
-

$$D_N = \sum_{t=1}^{T} \sum_{v=1}^{V} h(a_{Rt}^v, A_N), \qquad (4.61)$$

where D_N − is a security decision in a subnetwork, $h(a_{Rt}^v, A_N)$ − is the function of interaction between the router of the t -th level and network agents A_N, T − is the number of routers, V − is the hierarchy level.

- Environments $M_S \subset A_S$ - subset of server agents. The server agent $A_{SI} = \{a_{SI}^i, \dots a_{SI}^m\}$ interact to detect and neutralize attacks at the server level. Server agent environment:

$$D_S = \prod_{i=1}^{m} \sigma(a_{SI}^i, A_S), \qquad (4.62)$$

where D_S − is a joint decision of server agents, $\sigma(a_{SI}^i, A_S)$ − function of information exchange between server agents.

- Environments $M_{RS} \subset \{A_S^1, A_R^0\}$ - subset of edge router agents. Edge router agents a_{Rt}^0 interact with server agents a_{SI}^i for network perimeter protection. Edge router environment:

$$D_{RS} = \sum_{t=1}^{T} \lambda(a_{Rt}^0, a_{SI}^i), \qquad (4.63)$$

where D_{RS} − network perimeter protection solution, $\lambda(a_{Rt}^0, a_{SI}^i)$ − coordination function between edge router agents and servers.

- Environments,$M_R \subset A_R$ - internal and external router agents coordinate to provide protection against external attacks. Internal and external router environments:

$$D_R = \psi(a_{Rt}^0, A_R), \qquad (4.64)$$

where D_R − is the decision to coordinate between routers, (a_{Rt}^0, A_R) − function of interaction between internal and external routers.

This approach enables agents within a multi-agent system to detect and counter attacks on an enterprise information system and execute coordinated actions across different levels of the infrastructure, providing comprehensive protection against a wide range of threats (Wu et al., 2020). The hierarchical distribution of agents by infrastructure levels facilitates threat localization and minimizes the risk of their propagation throughout the network.

3. Graph Representation and Multilevel Message Propagation. A multiagent attack detection and counteraction system is represented as a graph $G = (V, E)$, where V is the set of graph vertices, and $V \cong A$; E - is the set of graph edges, and $\forall v_i \in V, v_j \in V, i \neq j, \exists e_i = (v_i, v_j) \Leftrightarrow a_i \in M_k \cap a_j \in M_k$ [26]. Let's divide the set of vertices of the graph into subsets $V = \{V_1, V_2 \dots, V_K\}$, so that $V_1 \cong A_W^1, \dots V_n \cong A_W^n$, $V_{n+1} \cong A_N, V_{n+2} \cong A_R^v, V_{n+3} \cong A_S^1, V_{n+m+2} \cong A_S^m$, $V_{n+m+3} \cong A_R^0$ with each subset including vertices that have no edges connecting them within the subset (Naseer et al., 2018; Rzaieva et al., 2024). This K-domain graph enables effective implementation of distributed threat analysis mechanisms, preventing excessive system load by limiting direct connections and ensuring hierarchical interaction (Skladannyi et al., 2025; Kostiuk & Bebeshko et al., 2024). Upon receiving a request for a coordinated decision, an agent distributes the request among neighbors not belonging to the immediate environment, forwarding to agents at different levels. This strategy aligns with scenarios where attackers target individual nodes and manipulate communication processes (Sokolov et al., 2025; Ma et al., 2009; Kostiuk & Bebeshko et al., 2024). The multi-level design localizes potentially compromised areas and confines malicious traffic impact. Agents at higher hierarchical levels perform filtering by verifying request source authenticity. Agents propagate messages following a hierarchical principle: requests are forwarded only to agents with higher trust or priority relative to the sender (Wu et al., 2020). Following request processing, agents receive return messages, augment them with risk assessments, correlate with historical data and trust levels, and transmit generalized decisions to the initiator (Kostiuk & Rzaieva et al., 2025).

Nevertheless, a principal limitation of traditional anomaly detection methods - particularly those based on neural networks or other machine learning algorithms - is an elevated incidence of false positives due to algorithmic sensitivity to deviations in user behavior and legitimate network interactions (Naseer et al., 2018). Conversely, employing a distributed approach to information system state analysis - where each agent independently collects, correlates, and reconciles data from diverse sources within the multi-agent environment - can compensate for these deficiencies, thereby improving overall situation assessment accuracy (Rzaieva et al., 2024; Skladannyi et al., 2025; Ricciato & Fleischer, 2004; Kostiuk & Sokolov et al., 2025).

4. Threat Evaluation Based on Neural Outputs and Priority Adjustment. Each agent's neural network generates output values in range $[a; b]$, divided into five sub-ranges $[a_i; b_i]$ corresponding to threat levels O_i, where $i = 1 \dots 5$. The first output follows the rule that lower levels indicate more critical deviations; the second output is inverted: lower levels indicate higher probability of benign events. Agents correlate events with threat levels, incorporating contextual analysis and information from other agents. This approach minimizes false classifications through coordinated decision-making in self-learning, adaptive architecture (Ma et al., 2009). Agents dynamically adapt $[a_i; b_i]$ boundaries based on current traffic, event type, and historical patterns, providing classification flexibility. Each agent applies weighted corrections incorporating confidence factors and alignment frequency between assessments and incidents. Agents with higher historical accuracy are assigned greater influence, enabling the system to learn and prioritize reliable sources. Advanced event analysis weights neural network outputs based on historical attack data, event correlation rules, and trust levels assigned to other agents (Almgren & Lindqvist, 2001).

For each agent, an ordered pair of its priority preferences is defined as $O_i \succ O_j \succ O_k \succ O_l \succ O_m$. The next level in the agent's priority system is set according to the closest interval to the values of l_1 and l_2, and then according to a similar principle. If the condition:

$$L > (a_i + 1/2(b_i - a_i)), if\ j = i + 1\ else\ j = i - 1, \qquad (4.65)$$

This means that the agent adjusts the level of its preferences according to the relative position of the value in the relevant threat interval, providing flexibility in decision-making and allowing adaptive adjustment of risk assessments in response to dynamic changes in the enterprise information system's state.

5. Collective Decision-Making: Condorcet and Nanson Methods. One effective approach to collective decision-making in a multi-agent system for detecting and counteracting attacks on an enterprise information system is a voting mechanism that allows agents to coordinate on the level of threat or the necessary response measures. In this approach, the accepted level, i.e., the joint decision of the system, is determined in accordance with the Condorcet criterion (Sokolov et al., 2025; Ma et al., 2009):

$$\forall \acute{o} \in O, \#(o \succ \acute{o}) \geq \#(\acute{o} \succ o), \tag{4.66}$$

where O is the set of possible solutions, o and $\acute{o}$ are individual solutions, $\#(o \succ \acute{o})$ is the number of agents who prefer solutions o to o'.

The Condorcet criterion guarantees that the selected solution is the one that outperforms all others by most votes, enabling the formulation of the most coherent and rational collective strategy to respond to threats (Sokolov et al., 2025; Ma et al., 2009). After performing local threat analysis and updating trust scores, each agent generates its threat level rating, transmitted to the collective decision-making module, which employs a voting mechanism to determine the final risk level. The Condorcet method selects the option that prevails in pairwise comparisons. However, classical Condorcet voting (Ma et al., 2009) suffers from the "Condorcet paradox," where cyclical preferences prevent a clear winner. To mitigate this, additional mechanisms include weighted voting, priority assignment based on trust levels, re-comparison procedures, stochastic methods, and voting delegation to more authoritative agents. Weighted voting schemes reflect agent trustworthiness, enabling dynamic adjustment of vote influence according to credibility, activity, and past effectiveness. Stochastic algorithms or vote delegation mechanisms increase robustness against informational noise and reduce conflict likelihood.

6. Architecture of the Multi-Agent System and Trust Processing. Figure 4.4 illustrates a DFD model for integrating artificial intelligence into a multi-agent attack detection and counteraction system. The model encompasses contextual interaction with the industrial environment via Modbus and Node-RED, and the internal process structure. The system comprises four primary subsystems: machine learning (ML), distributed attack detection (IDS), architecture optimization (OPT), and neural classification with decision-making (NNA). ML encompasses feature extraction, abnormality scoring, and dynamic parameter adjustment; IDS involves packet inspection, event correlation, and threat alert generation. The model depicts interactions with databases containing historical attack data and current network events, with feedback conveying threat assessments and responses to the industrial environment. For effective operation, trust relationships among agents are essential, as they directly influence the quality of collective decision-making. A specialized trust processing module performs adaptive updates based on behavioral analysis, neural network forecasting, and Markov transition models, flexibly regulating interactions and reinforcing cooperation between trustworthy agents while limiting the influence of those with lower trust levels.

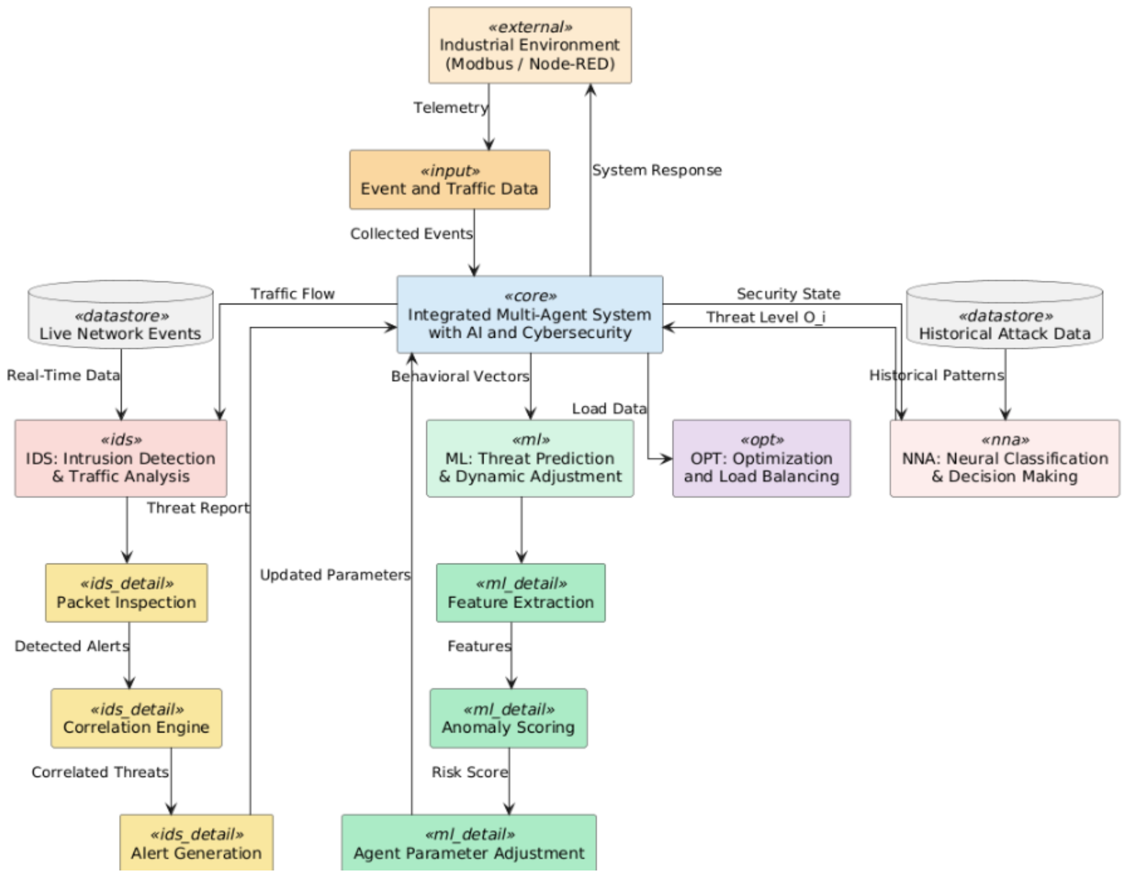
«external» Industrial Environment (Modbus / Node-RED)
Telemetry
System Response
«input» Event and Traffic Data
Collected Events
«datastore» Live Network Events
Traffic Flow
«core» Integrated Multi-Agent System with AI and Cybersecurity
Security State
Threat Level O_i
«datastore» Historical Attack Data
Real-Time Data
Behavioral Vectors
Load Data
Historical Patterns
«ids» IDS: Intrusion Detection & Traffic Analysis
«ml» ML: Threat Prediction & Dynamic Adjustment
«opt» OPT: Optimization and Load Balancing
«nna» NNA: Neural Classification & Decision Making
Threat Report
Updated Parameters
«ids_detail» Packet Inspection
«ml_detail» Feature Extraction
Detected Alerts
Features
«ids_detail» Correlation Engine
«ml_detail» Anomaly Scoring
Correlated Threats
Risk Score
«ids_detail» Alert Generation
«ml_detail» Agent Parameter Adjustment

Figure 4.4. DFD-Model for Integrating Artificial Intelligence Into a MAS for Detecting And Countering Attacks

Source: developed by the authors

Figure 4.5 presents a level 1 DFD model of the trust calculation subsystem. Inputs include neural analysis results of another agent (O_j), local context (T_i), and interaction history. The differential value of trust (Δw_{ij}) is calculated considering the current context and comparing expected and actual behavior. This is transferred to the Markov transition module, which determines possible trust state changes based on a probability matrix. The probabilities, current trust level, and Δw_{ij} are used to update trust, resulting in new level T_i', transferred to the aggregation layer for collective decisions. The system employs the Nanson voting method (Ma et al., 2009; Kostiuk & Bebeshko et al., 2024; Skladannyi & Kostiuk et al., 2025), which iteratively eliminates alternatives with lowest total support, ensuring the final winner coincides with the Condorcet winner if one exists, enhancing decision stability and reducing conflicts.

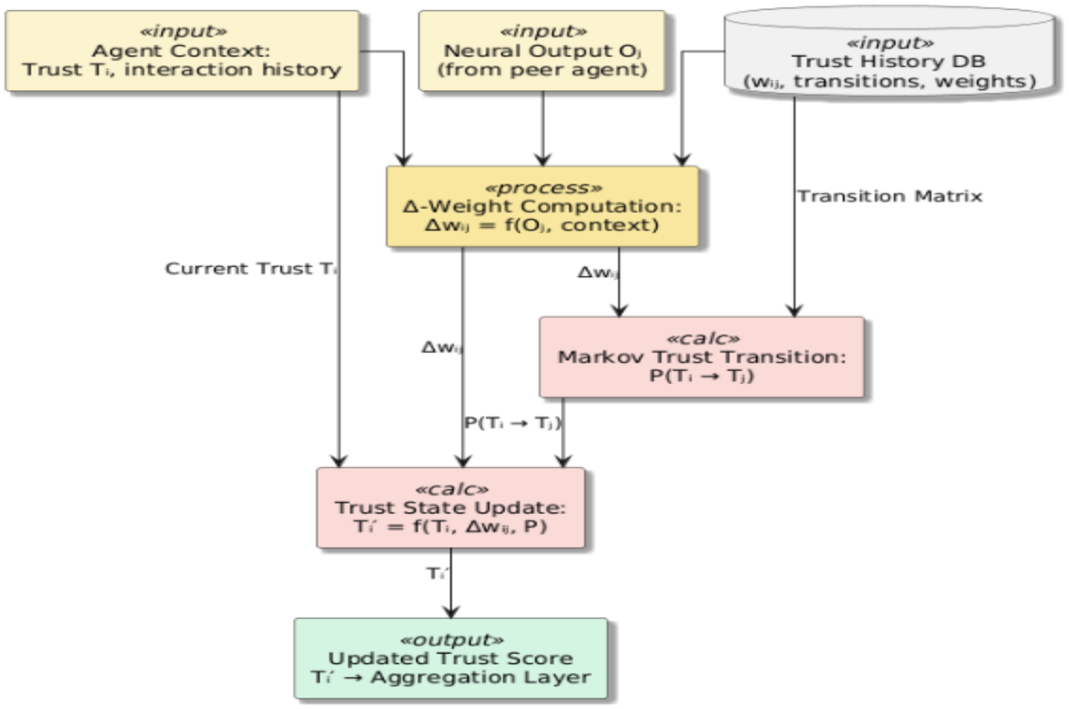

Figure 4.5. DFD- Level of the Subsystem of Trust Calculation in a Neural Multiagent System

Source: developed by the authors

The Nanson voting mechanism circumvents cyclical preference issues by recalculating weights after each elimination stage, particularly pertinent in dynamic network environments where priorities fluctuate. Each agent incorporates local information while maintaining confidence in collective decision stability. The outcome is a hierarchically coordinated decision satisfying trustworthiness and response effectiveness criteria. The effectiveness of a multi-agent system fundamentally depends on proper architectural design of individual agents. Architecture defines an agent's capability to collect, analyze, and evaluate threat information, make autonomous real-time decisions, and integrate into a shared environment (Figure 4.6).

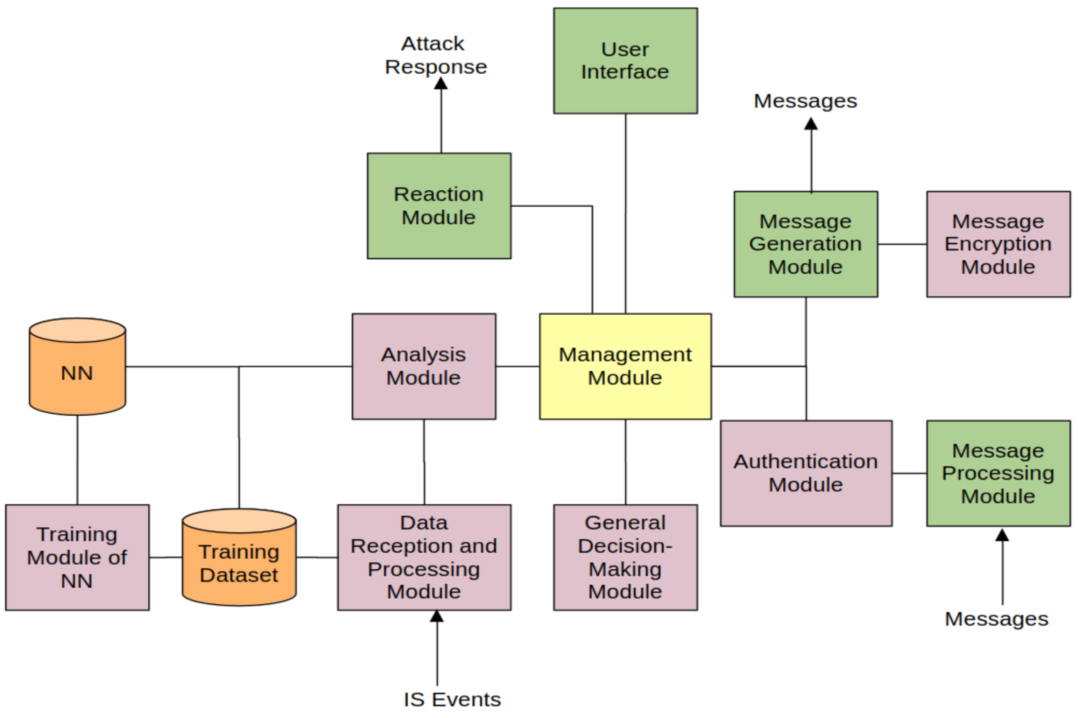

Figure 4.6. The Architecture of an Agent
Source: developed by the authors

The architecture of an agent within a multi-agent system encompasses several functional modules, including a sensing module responsible for collecting information from the network environment, an event analysis module that operates based on rule sets or machine learning models, a trust assessment module for evaluating other agents, a communication module facilitating data exchange with neighboring agents, and a decision-making module.

A crucial element is the secure local knowledge repository, which maintains the history of detected attacks, traffic processing parameters, and contextual patterns of anomalous behavior. In the context of threat detection, the agent conducts local analysis of network traffic and user activity, generates a risk assessment using an embedded neural network, and calculates confidence levels for the results obtained from other agents. Upon detecting suspicious activity, the agent initiates a data exchange protocol with neighboring agents using a weighted voting mechanism, wherein a Markov trust transition function determines each agent's influence.

To enhance response efficiency, the agent architecture supports dynamic reconfiguration of detection parameters, allowing adaptation to evolving traffic patterns or emerging attack scenarios. Additionally, the integration of self-learning mechanisms and continuous updating of behavioral profiles contributes to improved detection accuracy, reduced false positive rates, and increased operational flexibility within complex corporate environments. The internal structure of the agent consists of several principal components: a sensor mechanism for data acquisition from the network, an information processing and analysis system, and a decision-making unit.

The sensor mechanism enables the agent to monitor various elements of enterprise infrastructure, such as workstations, servers, routers, and access points, thereby acquiring essential data regarding traffic, user behavior, and other indicators relevant to anomaly detection and attack identification. The operational principles governing agents within the multi-agent system emphasize autonomy, adaptability, and interaction. Each agent autonomously makes decisions based on preliminary analysis of received data and employs machine learning or neural network algorithms. Adaptability allows agents to modify detection and response strategies in accordance with changes in the network environment, thereby effectively addressing novel threat vectors.

Interaction among agents is a vital architectural element; within the multi-agent system, agents exchange information and collaboratively assess situations to detect complex threats potentially overlooked by individual agents (Rzaieva et al., 2024). To this end, a "roundtable" mechanism is employed, facilitating collective decision-making based on the synthesis of individual analyses. Integration with the broader environment allows agents to communicate with other enterprise systems, including security monitoring platforms, incident response tools, and centralized information security management systems (Skladannyi et al., 2025; Ricciato & Fleischer, 2004). This connectivity enables agents to receive supplemental information or updates to security policies, thus refining their attack detection and response strategies (Kostiuk & Sokolov et al., 2025; Sokolov et al., 2025).

7. Multilevel Architecture and Functional Layers of the System. Multilevel Architecture and Functional Layers of the System. The multi-agent system implements a multi-level information processing approach, enabling threat identification at various developmental stages (Wu et al., 2020). The architecture is structured across several levels: (1) Sensor layer gathers data from diverse sources including network traffic, event logs, and user behavior, with preliminary filtering, normalization, and data structuring (Kostiuk & Rzaieva et al., 2025); (2) Analytical layer examines collected data using machine learning, neural networks, statistical modeling, and heuristic algorithms to detect anomalies, predict threats, and classify attacks

(Naseer et al., 2018); (3) Decision-making layer employs collective analysis mechanisms incorporating voting procedures, Bayesian inference, game theory, and fuzzy logic to determine appropriate response strategies (Rzaieva et al., 2024); (4) Communication layer ensures effective interaction, synchronizes processes, and coordinates decisions, managing command transmission and supporting rapid responses; (5) Executive level implements concrete measures including blocking suspicious connections, isolating compromised nodes, modifying security policies, and generating reports. The architecture may be augmented with adaptation, self-learning, and integration mechanisms, providing flexibility and resilience against novel attacks (Wu et al., 2020; Kostiuk & Rzaieva et al., 2025; Skladannyi et al., 2025).

8. Functional Modules of the System and Secure Communication. The control module performs comprehensive management of agent interactions, including receiving configurations, transmitting analysis results, authenticating subjects, and centralized agent management. It initiates data analysis, facilitates collective decision-making and information exchange, and coordinates responses to detected threats. The control module ensures coordinated operation, maintains integrity of monitoring, analysis, and response processes, and guarantees compliance with enterprise security policies. The data acquisition and processing module integrates with information sources (network event logs, traffic flows, user behavior). Incoming data undergoes primary processing, filtering, and normalization before neural network assessment, and is stored in a database as part of the training set. Historical information accumulation improves neural network training quality and anomaly detection efficiency. Continuous dataset updating enables adaptation to emerging threats and reduces false positive rates. The neural network training module updates parameters based on historical data using machine learning techniques, including backpropagation, improving threat classification accuracy. The learning process involves preprocessing, class balancing, dataset generation and validation, and training deep neural networks (recurrent or convolutional) tailored to detected threats. Dynamic retraining mechanisms adapt to evolving attacker behaviors, and ensemble learning methods aggregate outputs from multiple models, increasing accuracy and reducing false-positive and false-negative rates (Skladannyi et al., 2025; Ricciato & Fleischer, 2004; Kostiuk & Sokolov et al., 2025).

The analysis module forwards processed input to trained AI or neural network systems to assess risks and identify malicious activities. Analysis output is recorded and interpreted; events are either disregarded or classified as threats based on predefined thresholds. When a threat is detected, ordered agent priorities are established and transmitted to the management module, which coordinates decisions or directly initiates responses. This ensures continuous analysis and decision-making cycles, with agents independently evaluating threats and preparing proposals for collective deliberation. The system incorporates local risk assessments and global contextual factors from other agents and central analytical modules, facilitating dynamic adjustment of response strategies while maintaining balance between agent autonomy and coordinated protection. The reaction module executes actions aligned with agent intentions and defense strategy, including notifying cybersecurity specialists, blocking suspicious connections, activating host isolation, modifying security policies, transmitting ICMP packets, and generating detailed reports. The module supports flexible customization of response scenarios for various attack types and evolving threat environments.

The joint decision-making module (Figure 4.7) coordinates agent actions, facilitates collective threat analysis, and selects optimal responses. It generates ordered agent priorities, relayed to the management module with instructions to notify neighboring agents. Upon receiving priorities from other agents, the system conducts voting evaluating decision validity by considering historical data, trust levels, and current system state (Sokolov et al., 2025). This ensures consensus despite differing assessments and minimizes influence of unreliable or compromised nodes. The module identifies the threat level with highest support and dispatches unified response directives. The process is iterative; agents may reinitiate decision review in response to environmental changes or new attack data. If voting indicates agent error, a timer activates before re-evaluation. Should errors persist after multiple repetitions, reliability ratings are reduced, diminishing influence in future decisions (Ma et al., 2009; Kostiuk & Bebeshko et al., 2024). The message generation module creates messages including agent parameter configurations, system state inquiries, or notifications about collective decision-making procedures or urgent threat responses (Kostiuk & Sokolov et al., 2025). Messages are tagged with priority, authenticity, and source trust level attributes, and routed through optimal communication channels.

The message processing module analyzes incoming communications to determine type and corresponding actions. Based on content, the system updates agent configurations or provides responses containing current system state, ordered agent priorities, or decision-making data (Sokolov et al., 2025).

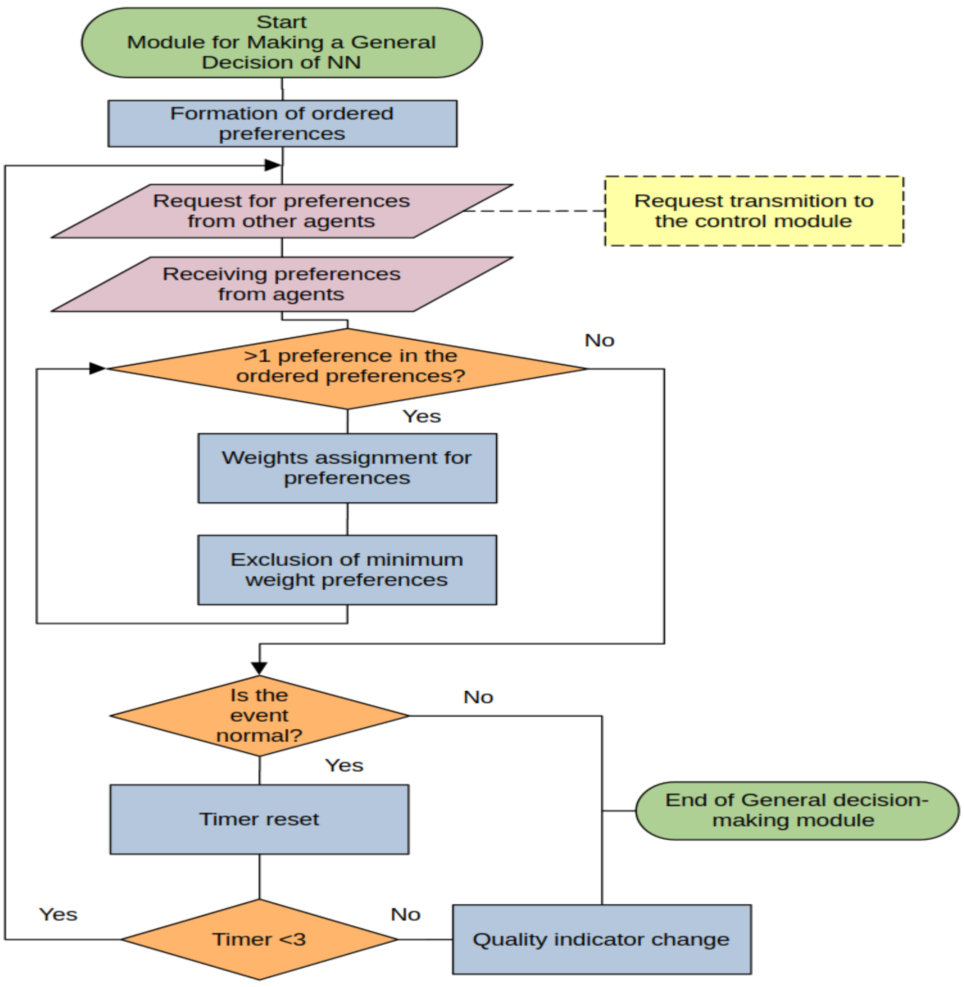
Start
Module for Making a General Decision of NN
Formation of ordered preferences
Request for preferences from other agents
Request transmition to the control module
Receiving preferences from agents
>1 preference in the ordered preferences?
No
Yes
Weights assignment for preferences
Exclusion of minimum weight preferences
Is the event normal?
No
Yes
Timer reset
End of General decision-making module
Yes
Timer <3
No
Quality indicator change

Figure 4.7. Flowchart of the General Decision-Making Algorithm
Source: developed by the author

The module maintains interaction consistency, preserves information flow integrity, and filters messages according to trustworthiness and authenticity criteria. When messages convey critical threat information, the module triggers appropriate procedures, activating local responses or delegating to the collective decision-making module. The encryption module provides cryptographic protection for all messages, ensuring confidentiality, integrity, and authenticity. Implementations employ symmetric and asymmetric encryption algorithms (AES, RSA) and protocols designed to resist man-in-the-middle attacks (Shameli-Sendi et al., 2018; Vigna et al., 2003; Assante & Lee, 2015). The module integrates with a key management system facilitating secure key exchanges, considering trust levels and authentication status, supporting encryption for direct agent-to-agent communication and interactions with centralized services. The authentication module verifies legitimacy of agents and personnel, ensuring only authorized users and trusted nodes access information resources or influence decision-making. Techniques include multi-factor authentication, digital certificates, one-time passwords, and biometric verification (Shulika et al., 2024; Skladannyi & Samoilenko et al., 2025; Kostiuk & Samoilenko et al., 2025). Adaptive authentication mechanisms adjust verification strictness based on threat levels or interaction contexts. Upon detection of anomalous activity, the system enforces stricter authentication requirements.

9. Methodology for Multi-Agent Modeling of Attack Defense. The methodology encompasses four stages: preparatory, parameter configuration, modeling implementation, and output parameter analysis (Figure 4.8).

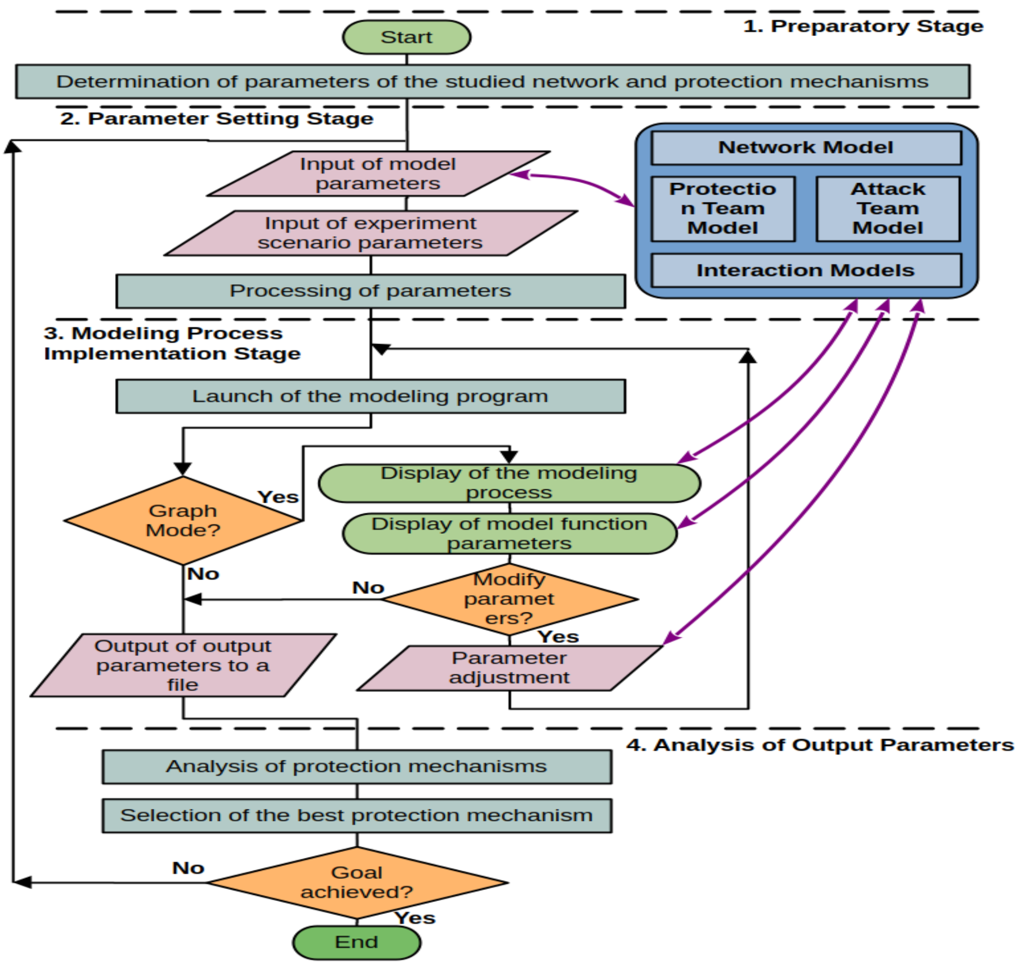
1. Preparatory Stage
Start
Determination of parameters of the studied network and protection mechanisms
2. Parameter Setting Stage
Input of model parameters
Input of experiment scenario parameters
Processing of parameters
Network Model
Protection Team Model
Attack Team Model
Interaction Models
3. Modeling Process Implementation Stage
Launch of the modeling program
Graph Mode?
Yes
No
Display of the modeling process
Display of model function parameters
Modify parameters?
No
Yes
Output of output parameters to a file
Parameter adjustment
4. Analysis of Output Parameters
Analysis of protection mechanisms
Selection of the best protection mechanism
Goal achieved?
No
Yes
End

Figure 4.8. A Generalized Presentation of the Methodology for Multi-Agent Modeling of Attack Defense Mechanisms

Source: developed by the author

The modeling implementation stage uses OMNET++ platform, while other stages are managed by developers (Vigna & Robertson et al., 2003; Kostiuk & Zhyltsov et al., 2025).

Models are interconnected via bidirectional arrows, illustrating integration pathways. Figure 4.8 presents the step-by-step modeling process. The preparatory stage defines network parameters and protection mechanisms. The parameter-setting stage inputs and processes model and scenario parameters. The modeling implementation stage launches the simulation, outputs process parameters, and configures functional characteristics; if objectives are not met, parameters are adjusted and simulation repeated. The final stage analyzes results and selects the optimal protection mechanism.Input parameters are specified using NED language and OMNET++ platform (Hughes et al., 2020; Kostiuk & Sokolov et al., 2025; Kostiuk & Bebeshko et al., 2024; Kruegel et al., 2002). Protection effectiveness is evaluated based on output parameters: incoming traffic volume before and after filtering, attack detection accuracy (percentage of false positives), successful attack penetrations, and system response time. The methodology enables systematic examination, configuration of modeling processes, and selection of effective protection mechanisms via mathematical optimization techniques, including the lexicographic method (Shameli-Sendi et al., 2018; Shulika et al., 2024). The modeling accounts for variability in attack scenarios, allowing assessment of system resilience. NED and OMNET++ tools allow detailed description of network topology, agent logic, and communication channel characteristics. During the preparatory stage, the enterprise information system structure is analyzed, and potential threats are identified and classified (Vigna & Robertson et al., 2003). At the parameter-setting stage, agents are configured according to security policies, countermeasure activation criteria are established, and inter-agent communication parameters are defined. The modeling implementation stage tests various attack scenarios and analyzes responses, considering dynamic adaptation to evolving attacker behaviors. The final stage compares effectiveness of different response strategies, evaluates system load, and generates recommendations (Assante et al., 2015; Huges et al., 2020).

10. Simulation Topology, Attack Scenarios and Agent Interaction Model. The network topology was developed based on a power function describing node connection density distribution, facilitating realistic modeling of an enterprise network. The topology includes 50 nodes: a secure server, 10 clients, and other network components (Liu et al., 2024;

Skladannyi & Samoilenko et al., 2025; Kruegel et al., 2002), generating typical traffic and replicating authentic enterprise conditions. Key parameters include client request volumes, network connection characteristics, and data transfer mechanisms. To simulate hostile activity, 10 attack agents were deployed, each conducting UDP flood attacks targeting server infrastructure. Several cooperative defense schemes were integrated, enabling interaction among defense agents. Experimental evaluations assessed the multi-agent security system effectiveness within realistic enterprise infrastructure, emphasizing agents' ability to interact, adapt behaviors, and maintain resilience against network resource overload. Results demonstrated advantages of cooperative security strategies.

Figure 4.9 depicts principal stages of agent interaction during threat processing, from abnormal traffic detection to risk evaluation, response activation, command confirmation, and event logging.

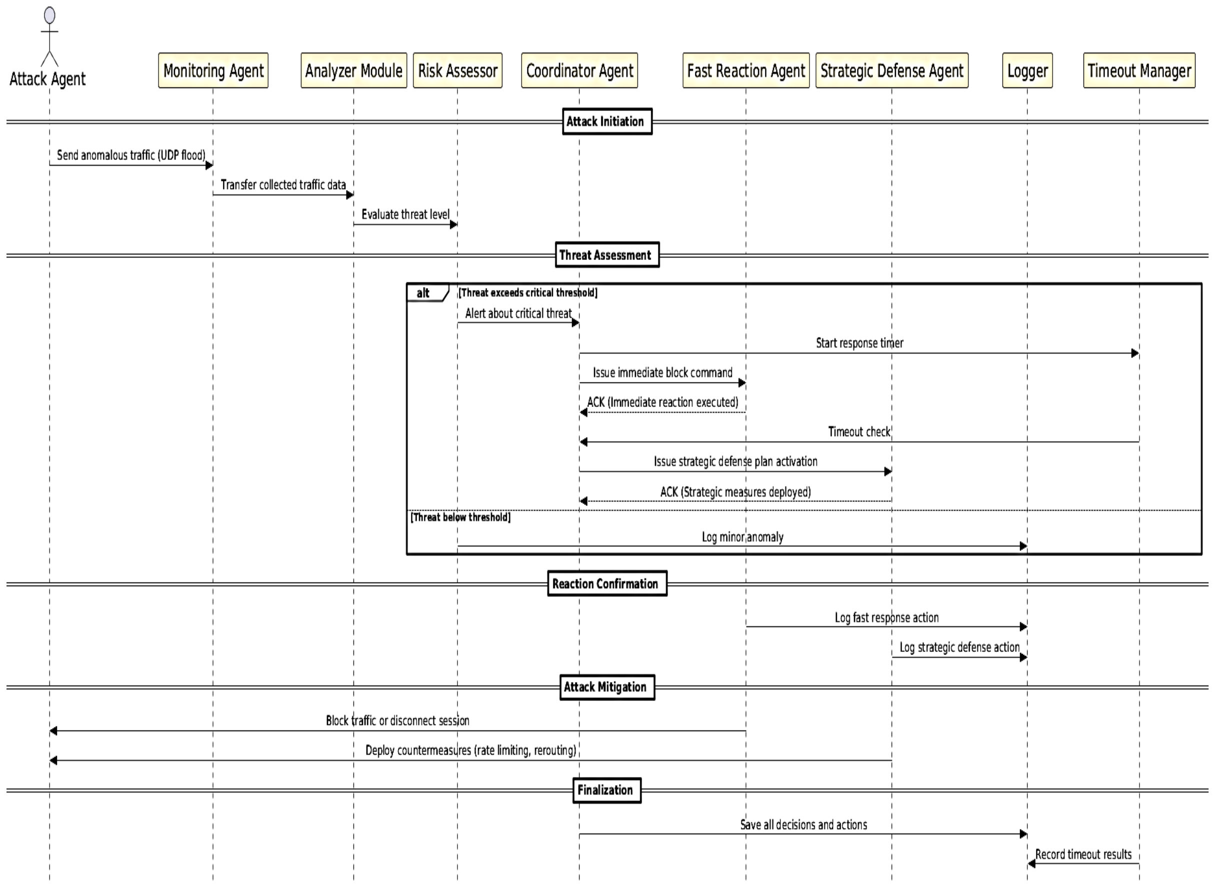
Attack Agent
Monitoring Agent
Analyzer Module
Risk Assessor
Coordinator Agent
Fast Reaction Agent
Strategic Defense Agent
Logger
Timeout Manager
Attack Initiation
Send anomalous traffic (UDP flood)
Transfer collected traffic data
Evaluate threat level
Threat Assessment
alt
[Threat exceeds critical threshold]
Alert about critical threat
Start response timer
Issue immediate block command
ACK (Immediate reaction executed)
Timeout check
Issue strategic defense plan activation
ACK (Strategic measures deployed)
[Threat below threshold]
Log minor anomaly
Reaction Confirmation
Log fast response action
Log strategic defense action
Attack Mitigation
Block traffic or disconnect session
Deploy countermeasures (rate limiting, rerouting)
Finalization
Save all decisions and actions
Record timeout results

Figure 4.9. Diagram of the Activity of a MAS Response to an Attack, Taking Into Account Delivery Confirmation, Timeout and Distribution of Protective Measures

Source: developed by the authors

ISBN 978-9916-9320-4-9 (pdf)

Response time management via a timeout module ensures timely activation of strategic measures. The architecture enhances resilience against complex attacks and facilitates effective real-time coordination. The model demonstrates rapid incident response, coordinated protection efforts, and comprehensive decision logging, distinguishing between immediate responses and strategic defense mechanisms.

11. Experimental Results and Comparative Evaluation of Defense Mechanisms. Experimental evaluations demonstrated superior efficacy of cooperative defense schemes compared to traditional isolated approaches. Across all scenarios, marked reduction in attack traffic intensity was observed, with the most effective scheme involving comprehensive interaction among all defense agents (Assante et al., 2015). Adopting an integrated cooperative mechanism substantially enhances resilience against DDoS attacks. Attack traffic decreased to a minimum approximately 450 seconds after the defense system became operational (Figure 4.10), demonstrating high adaptability and effectiveness of response algorithms. Samplers, which continuously analyze and dynamically exchange network state data among security agent teams, played a pivotal role in diminishing attack effectiveness, accelerating threat detection, and enhancing coordination. Results validate the feasibility of employing a multi-agent approach for detecting and countering attacks on enterprise information systems (Bhardwaj et al., 2002).

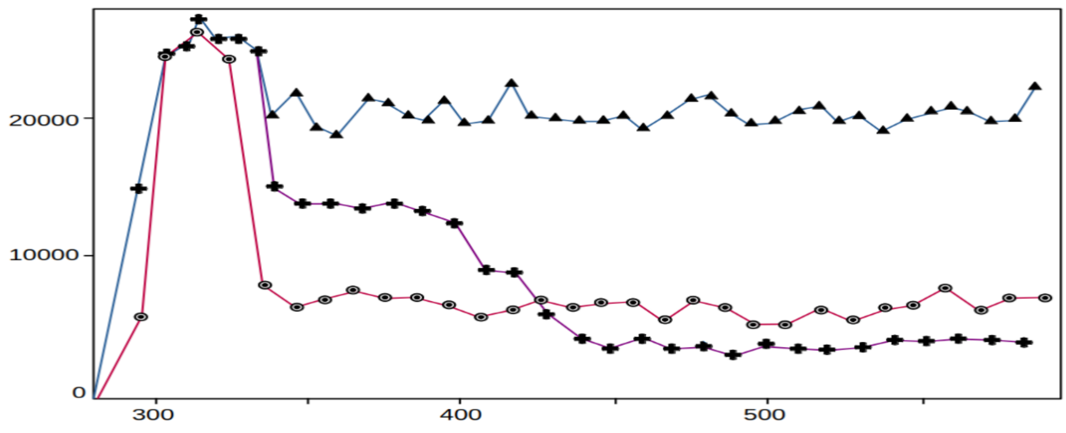

Figure 4.10. Dependence of Attack Traffic (Mbps) on Time (s) for DefCom (circles), COSSACK (triangles), and "Full Cooperation" (Crosses) Defense Mechanisms

Source: developed by the authors

The integration of sequential agent interaction modeling with empirical research findings substantiates the efficacy of a multi-level cooperative approach to attack detection and mitigation within enterprise information systems. The proposed model exemplifies the coherence of coordinated responses and lays the groundwork for implementing adaptive defense strategies in real-world operational contexts. Experimental outcomes confirm that such systems can maintain stable infrastructure performance even under sustained attack conditions, reducing risks and enhancing overall cyber resilience.

12. Deployment Methodology and Administrator-Guided Activation of Agents. According to the devised methodology for deploying a multi-agent system to detect and counteract attacks on enterprise information systems, each operational stage is executed under the direct supervision of the information system administrator. This role encompasses coordinating agent deployment, monitoring agent activities, training neural networks, and integrating analytical outcomes into the enterprise's overarching information security management framework. Each phase of system operation is critical to ensuring robust attack protection, particularly through machine learning techniques that afford high adaptability to emerging threat types (Figure 4.11) (Kostiuk & Khorolska et al., 2024). Experimental evidence confirms the developed multi-agent system's capacity to provide adaptive and coordinated protection of enterprise information infrastructure even under challenging conditions of active attacker engagement.

At the initial stage, agents are strategically deployed to relevant objects for monitoring, analysis, and protection. The administrator optimizes network distribution, positioning agents at critical nodes (servers, routers, access points, specialized hosts) to implement traffic analysis, anomaly detection, and real-time threat mitigation. The second stage involves passive data collection, with agents operating in monitoring mode, recording system parameters, examining user behavioral patterns, and characterizing network traffic to establish a baseline. A data collection period of at least two weeks is recommended to account for seasonal load variations and typical system interactions. Agents capture normal operational parameters and instances of attacks, policy violations, and anomalous behavior, generating a training dataset.

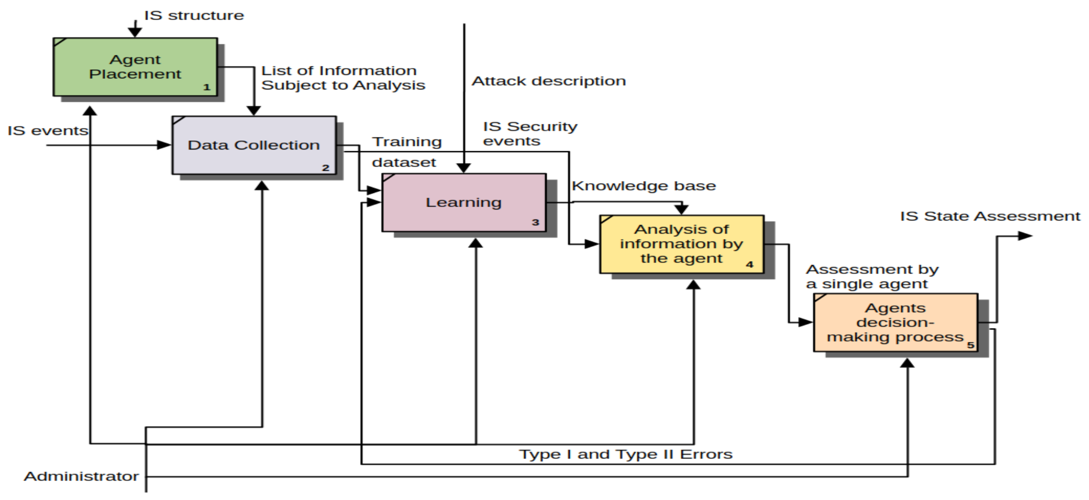

Figure 4.11. IDF- Diagram of the Methodology for Detecting Attacks Using the Developed Multi-Agent System for Detecting and Countering Attacks

Source: developed by the authors

This dataset underpins neural network training, which serves as the primary decision-making mechanism. Through cognitive modeling, agents encode "beliefs" regarding normal and abnormal system functioning, empowering autonomous decision-making. At the fourth stage, agents transition to active attack detection mode, continuously analyzing incoming data, performing preprocessing, and forwarding information to neural network input layers. Upon detecting anomalous patterns, agents initiate collective decision-making protocols. Through voting, if the collective assessment classifies the system state as hazardous, agents execute countermeasures including blocking suspicious connections, rerouting traffic, activating incident response protocols, or isolating compromised nodes (Almgren & Lindqvist, 2001; Callegari et al., 2017). The multi-agent detection system comprises agents operating at various infrastructure levels (workstations, servers, routers), ensuring comprehensive data collection. Agents operate both individually and collaboratively, enhancing effectiveness (Kriuchkova et al., 2024). Joint decision-making organizes a "roundtable" forum where each agent contributes analysis results, enabling holistic assessment and reducing errors from data scarcity or individual subjectivity. The methodology leverages multi-agent technologies to train and enhance adaptability, encompassing sequential stages from data collection to active anomaly detection and attack mitigation. Machine learning models analyze collected data to identify anomalies, enabling autonomous adaptation to novel attack types. By consolidating analyses and employing collective decision-making, the system accurately identifies threat locations and characteristics, ensuring high-precision detection.

Conclusions. The proposed methodology encompasses strategic agent placement, initial training based on collected data, integration of machine learning and behavioral analysis techniques, and continuous monitoring with collective decision-making. Autonomous agent decision-making synchronized across the system enables prompt responses while minimizing missed detections and false alarms. Feedback mechanisms facilitate adaptive behavioral adjustments informed by previous decisions and emerging threat intelligence, enabling continuous evolution and enhanced cyber resilience. Cooperative learning strategies and information exchange yield synergistic effects, improving detection accuracy of sophisticated attacks and ensuring rapid threat mitigation. Seamless integration with existing information security management tools, including SIEM systems, cloud security platforms, and Threat Intelligence services, is critical. Such modular architecture provides comprehensive protection and enables rapid adaptation to emergent attack vectors. Multi-agent threat detection and counteraction technologies are emerging as effective instruments for ensuring enterprise information system resilience, facilitating early threat detection, expeditious response, and substantial reduction of cyber incident risks.

Funding. The author declare that no financial support was received for the research, authorship, and/or publication of this article.

Conflict of interest. The author declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Generative AI statement. The author declare that no Generative AI was used in the creation of this manuscript.

References :

1. Almgren, M., & Lindqvist, U. (2001). Application-integrated data collection for security monitoring. In *Proceedings of Recent Advances in Intrusion Detection (RAID), LNCS* (pp. 22–36). Springer. http://dx.doi.org/10.1007/3-540-45474-8_2
2. Assante, M. J., & Lee, R. M. (2015). The industrial control system cyber kill chain. *SANS Institute.* https://doi.org/10.20935/AcadQuant7690
3. Bhardwaj, A., Chandok, S. S., Bagnawar, A., Mishra, S., & Uplaonkar, D. (2022). Detection of cyber attacks: XSS, SQLI, phishing attacks and detecting intrusion

using machine learning algorithms. *IEEE Global*. http://dx.doi.org/10.1109/GlobConPT57482.2022.9938367
4. Callegari, C., Giordano, S., & Pagano, M. (2017). Entropy-based network anomaly detection. In *International Conference on Computing, Networking and Communications (ICNC)* (pp. 334–340). doi: 10.1109/ICCNC.2017.7876150.
5. Hughes, K., McLaughlin, K., & Sezer, S. (2020). Dynamic countermeasure knowledge for intrusion response systems. In *2020 31st Irish Signals and Systems Conference (ISSC)* (pp. 1–6). https://doi.org/10.1109/ISSC49989.2020.9180198
6. Javaid, A., Niyaz, Q., Sun, W., & Alam, M. (2016). A deep learning approach for network intrusion detection system. *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies*, 21–26. DOI 10.4108/eai.3-12-2015.2262516
7. Kostiuk, Y., Skladannyi, P., Sokolov, V., Zhyltsov, O., Ivanichenko, Y. Effectiveness of Information Security Control using Audit Logs. Proceedings of the Workshop on Cybersecurity Providing in Information and Telecommunication Systems (CPITS 2025), 2025. Aachen: CEUR, 2025. Vol. 3991. P. 524–538. ISSN 1613-0073.
8. Kostiuk, Y., Dovzhenko, N., Mazur, N., Skladanny, P., & Rzaeva, S. (2025). Methods of protecting the grid environment from malicious code during the execution of computational tasks. *Cybersecurity: Education, Science, Technology, 3*(27), 22–40. https://doi.org/10.28925/2663-4023.2025.27.710
9. Kostiuk, Y., Rzaieva S., Khorolska, K., Mazur, N., Korshun, N. Architecture of the software system of confidential access to information resources of computer networks. Proceedings of the Workshop Cyber Security and Data Protection, July 31, 2025, Lviv, Ukraine (CSDP 2025). Vol. 4042. P. 37–53. ISSN 1613-0073.
10. Kostiuk, Y., Skladannyi, P., Sokolov, V., Rzaieva S.(2025) Intelligent System for Simulation Modeling and Research of Information Objects. Proceedings of the 1st Workshop Software Engineering and Semantic Technologies (SEST 2025), co-located with the 15th International Scientific and Practical Programming Conference (UkrPROG 2025), May 13□14, 2025, Kyiv, Ukraine, Vol-4053, P. 237-251. ISSN 1613-0073.
11. Kostiuk, Y., Bebeshko, B., Kryuchkova, L., Lytvynov, V., Oksanych, I., & Khorolska, K. (2024). Information protection and data exchange security in wireless mobile networks with authentication and key exchange protocols. *Cybersecurity: Education, Science, Technology, 1*(25), 229–252. https://doi.org/10.28925/2663-4023.2024.25.229252
12. Kostiuk, Y., Skladannyi, P., Samoilenko, Y., Khorolska, K., Bebeshko, B., & Sokolov, V. (2025). A system for assessing the interdependencies of information system agents in information security risk management using cognitive maps. *Cyber Hygiene & Conflict Management in Global Information Networks 2024, 3925*, 249–264. https://ceur-ws.org/Vol-3925/paper21.pdf
13. Kostiuk, Y., Skladannyi, P., Sokolov, V., Vorokhob, M. Models and technologies of cognitive agents for decision-making with integration of Artificial Intelligence. Proceedings of the Modern Data Science Technologies Doctoral Consortium (MoDaST 2025). Aachen: CEUR, 2025. Vol. 4005. P. 82–96. ISSN 1613-0073.
14. Kriuchkova, L., Sokolov, V., & Skladannyi, P. (2024). Determining the zone of successful interaction in RFID technologies. In *IEEE 29th International*

Seminar/Workshop on Direct and Inverse Problems of Electromagnetic and Acoustic Wave Theory, 168–171. https://doi.org/10.1109/diped63529.2024.10706153

15. Kruegel, C., Toth, T., & Kirda, E. (2002). Service specific anomaly detection for network intrusion detection. In *Proceedings of ACM Symposium on Applied Computing* (pp. 201–208). http://dx.doi.org/10.1145/508791.508835
16. Liu, N., Wu, H., Zhang, Y., & Wang, C. (2024). Adversarial attacks against black-box network intrusion detection based on heuristic algorithm. In *2024 10th International Conference on Computer and Communications (ICCC)* (pp. 1954–1958). https://doi.org/10.1109/ICCC62609.2024.10941941
17. Logesh, B., Perasani, B., Kumar, A. J., Genji, Y., Kiran, C. U., & Godara, J. (2023). Web attack detection using deep learning. In *Proc. KILBY 100 7th Int. Conf. Comput. Sci. (ICCS 2023)*. http://dx.doi.org/10.2139/ssrn.4483837
18. Naseer, S., Saleem, Y., Khalid, S., Bashir, M. K., Han, J., Iqbal, M. M., & Han, K. (2018). Enhanced network anomaly detection based on deep neural networks. *IEEE Access, 6*, 48231–48246. doi: 10.1109/ACCESS.2018.2863036
19. Ma, J., Dai, G., & Xu, Z. (2009). Network anomaly detection using dissimilarity-based one-class SVM classifier. In *International Conference on Parallel Processing Workshops, ICPPW 2009* (pp. 409–414). doi: 10.1109/ICPPW.2009.6.
20. Ricciato, F., & Fleischer, W. (2004). Bottleneck detection via aggregate rate analysis: A real case in a 3G network. In *Proc. IEEE/IFIP NOMS*. doi: 10.1109/NOMS.2006.1687628.
21. Roshan, K., Zafar, A., & Ul Haque, S. B. (2023). A novel deep learning based model to defend network intrusion detection system against adversarial attacks. In *2023 10th International Conference on Computing for Sustainable Global Development (INDIACom)* (pp. 386–391). https://doi.org/10.48550/arXiv.2308.00077
22. Rzaieva, S., Rzaiev, D., Kostyuk, Y., Hulak, H., & Shcheblanin, O. (2024). Methods of modeling database system security. *CPITS-2024: Cybersecurity Providing in Information and Telecommunication Systems*, Kyiv, Ukraine. https://ceur-ws.org/Vol-3654/short5.pdf
23. Samoilenko, Y., Smitiukh, Y., Kostiuk, Y., Stepashkina, K., & Hnatchenko, D., Yaremych, V. (2024). Using Node-Red to visualize dairy production data via Modbus. In R. Szewczyk, C. Zieliński, M. Kaliczyńska & V. Bučinskas (Eds.), *Automation 2024: Advances in Automation, Robotics and Measurement Techniques (Lecture Notes in Networks and Systems, Vol. 1219)* (pp. 81-90). Springer. https://doi.org/10.1007/978-3-031-78266-4_8
24. Shameli-Sendi, A., Louafi, H., He, W., & Cheriet, M. (2018). Dynamic optimal countermeasure selection for intrusion response system. *IEEE Transactions on Dependable and Secure Computing, 15*(5), 755–770. https://doi.org/10.1109/TDSC.2016.2615622
25. Shulika, K., Balagura, D., Smirnov, A., Nepokritov, D., & Lytvyn, A. (2024). A method of using modern endpoint detection and response (EDR) systems to protect against complex attacks. *Modern State of Scientific Research and Technologies in Industry, 2*(28), 182–195. http://dx.doi.org/10.30837/2522-9818.2024.2.182
26. Skladannyi, P., Kostiuk Y., Khorolska, K., Bebeshko, B., Sokolov, V. Model and methodology for the formation of adaptive security profiles for the protection of wireless networks in the face of dynamic cyber threats. Proceedings of the Workshop

ISBN 978-9916-9320-4-9 (pdf)

Cyber Security and Data Protection, July 31, 2025, Lviv, Ukraine (CSDP 2025). Vol. 4042. P. 17–36. ISSN 1613-0073.

27. Skladannyi, P. M., Kostiuk, Y. V., Mazur, N. P., & Pitaichuk, M. A. (2025). Investigation of characteristics and performance of access protocols to cloud computing environments based on universal testing. *Telecommunication and Information Technologies, 1*(86), 61–74. DOI: 10.31673/2412-4338.2025.014649
28. Skladannyi, P., Kostiuk, Y., Rzaeva, S., Samoilenko, Y., & Savchenko, T. (2025). Development of modular neural networks for detecting different classes of network attacks. *Cybersecurity: Education, Science, Technology, 3*(27), 534–548. https://doi.org/10.28925/2663-4023.2025.27.772
29. Sokolov, V., Kostiuk, Y., Skladannyi, P., Korshun, N. (2025). Adaptation of Network Traffic Routing Policy to Information Security and Network Protection Requirements. Proceedings of the 13th International Scientific and Practical Conference "Information Control Systems and Technologies", ICST 2025, Vol-4048, P. 397 - 411. ISSN 1613-0073.
30. Taher, K. A., Jisan, B. M. Y., & Rahman, M. M. (2019). Network intrusion detection using supervised machine learning technique with feature selection. *International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST)*, 643–646. http://dx.doi.org/10.1109/ICREST.2019.8644161
31. Vigna, G., Robertson, W., Kher, V., & Kemmerer, R. A. (2003). A stateful intrusion detection system for World-Wide Web servers. In *Proceedings of the 19th Annual Computer Security Applications Conference (ACSAC)* (pp. 34–43). http://dx.doi.org/10.1109/CSAC.2003.1254308
32. Vigna, G., Valeur, F., & Kemmerer, R. A. (2003). Designing and implementing a family of intrusion detection systems. In *Proceedings of the 9th European Software Engineering Conference held jointly with 11th ACM SIGSOFT International Symposium on Foundations of Software Engineering (ESEC/FSE-11)* (pp. 88–97). https://doi.org/10.1145/949952.940084
33. Wu, Y., Wei, D., & Feng, J. (2020). Network attacks detection methods based on deep learning techniques: a survey. *Security and Communication Networks*, 1–17. https://doi.org/10.1155/2020/8872923