Article

An Information Security Engineering Framework for Modeling Packet Filtering Firewall Using Neutrosophic Petri Nets

Jamal Khudair Madhloom [1], Zainab Hammoodi Noori [2,3], Sif K. Ebis [4], Oday A. Hassen [4,*] and Saad M. Darwish [5]

1 College of Arts, Wasit University, Kut 52001, Iraq; jamalkh@uowasit.edu.iq
2 Ministry of Education, Karbala Education Directorate, Karbala 56001, Iraq; fathayrt21@gmail.com
3 College of Engineering, University of Warith Al-Anbiyaa, Karbala 56001, Iraq
4 Ministry of Education, Wasit Education Directorate, Kut 52001, Iraq; saifkather@gmail.com
5 Department of Information Technology, Institute of Graduate Studies and Research, Alexandria University, 163 Horreya Avenue, Alexandria 21526, Egypt; saad.darwish@alexu.edu.eg
* Correspondence: odayali@uowasit.edu.iq; Tel.: +60-9647827554545

Abstract: Due to the Internet's explosive growth, network security is now a major concern; as a result, tracking network traffic is essential for a variety of uses, including improving system efficiency, fixing bugs in the network, and keeping sensitive data secure. Firewalls are a crucial component of enterprise-wide security architectures because they protect individual networks from intrusion. The efficiency of a firewall can be negatively impacted by issues with its design, configuration, monitoring, and administration. Recent firewall security methods do not have the rigor to manage the vagueness that comes with filtering packets from the exterior. Knowledge representation and reasoning are two areas where fuzzy Petri nets (FPNs) receive extensive usage as a modeling tool. Despite their widespread success, FPNs' limitations in the security engineering field stem from the fact that it is difficult to represent different kinds of uncertainty. This article details the construction of a novel packet-filtering firewall model that addresses the limitations of current FPN-based filtering methods. The primary contribution is to employ Simplified Neutrosophic Petri nets (SNPNs) as a tool for modeling discrete event systems in the area of firewall packet filtering that are characterized by imprecise knowledge. Because of SNPNs' symbolic ability, the packet filtration model can be quickly and easily established, examined, enhanced, and maintained. Based on the idea that the ambiguity of a packet's movement can be described by if–then fuzzy production rules realized by the truth-membership function, the indeterminacy-membership function, and the falsity-membership functional, we adopt the neutrosophic logic for modelling PN transition objects. In addition, we simulate the dynamic behavior of the tracking system in light of the ambiguity inherent in packet filtering by presenting a two-level filtering method to improve the ranking of the filtering rules list. Results from experiments on a local area network back up the efficacy of the proposed method and illustrate how it can increase the firewall's susceptibility to threats posed by network traffic.

Keywords: security engineering; intelligent firewall; neutrosophic Petri net; packet filtering; access control list

check for updates

Citation: Madhloom, J.K.; Noori, Z.H.; Ebis, S.K.; Hassen, O.A.; Darwish, S.M. An Information Security Engineering Framework for Modeling Packet Filtering Firewall Using Neutrosophic Petri Nets. *Computers* **2023**, *12*, 202. https://doi.org/10.3390/computers12100202

Academic Editors: Ömer Aslan and Refik Samet

Received: 23 July 2023
Revised: 30 September 2023
Accepted: 2 October 2023
Published: 8 October 2023

1. Introduction

The goal of security engineering is to ensure that a system can function reliably even if it is subjected to an attack, whether intentional or accidental. Whole systems may be designed, implemented, and tested, and existing systems can be modified to suit changing conditions by using the techniques and procedures studied in this field [1,2]. Cryptography, computer security, tamper-resistant hardware, formal techniques, economics, applied psychology, organizational knowledge, and the law are just a few of the many fields of study that contribute to successful security engineering. Business process analysis, software engineering, assessment, and testing are all components of system engineering that should not be overlooked, but they alone are not enough since they focus on accidents rather than malicious intent. In order to create secure software systems, security engineering must become a standard component of the software development process [1–4].

Security of the internal network against attacks, fraudulent traffic, and unauthorized access is becoming more important as computer technology and the usage of computer networks continue to grow. Given the importance of firewalls to overall network security, firewall technology has grown to become a significant focus of network security studies. When two or more networks need to communicate with one another securely, they may utilize a firewall, which is a system of hardware and software used to enforce a certain policy about security. A firewall, in its most basic form, is a protective barrier that is placed between an internal network and an external network to regulate traffic and connections. Network administration security policy dictates the criteria for approving or rejecting connections and service requests [5,6].

Many functions can be carried out by a firewall system, including the definition of a single choke point that prevents unauthorized users from accessing the protected network; services that may compromise the network's security are prevented from entering or departing; and security event monitoring infrastructure with built-in audits and alerts. In addition, virtual private networks (VPNs) and IP security may be set up and deployed via this platform [7]. The firewall has several drawbacks, however, such as not being able to prevent attacks that bypass it, not providing foolproof security against malicious insiders, and allowing unauthorized users to access an unprotected wireless network. In general, the complexity and needs of firewall design and maintenance are rising as the Internet continues to expand and attacks become more sophisticated [8–10].

The type of firewall varies from packet filtering and circuit level to a proxy service [11–13]. The most secure type of firewall is proxies, or application-level firewalls, which are CPU-intensive and have a considerable performance penalty. The penalty is incurred because every time a user initiates a new session, a new process must be launched. These application-layer firewalls are protocol-specific and operate at the application layer. The circuit-level firewall is another type of firewall that works at the session layer and provides a more complete type of protection. For TCP connections, these firewalls serve as relays. They monitor the handshake between packets to determine if a requested session is legitimate and finish the TCP connection handshake on behalf of the host behind them. Circuit-level firewalls are often less expensive than other types of firewalls. Due to more general concerns about filtering the packets, many packets may go undetected when using this kind of filtering [11].

A packet-filtering firewall is the first line of defense for every network. It analyzes each packet's header and passes those packets in accordance with the filter's rules provided in the access control list (ACL). It is a fundamental part of any network monitoring tool since it operates at the network layer. Since applications are unimportant in this type of firewall, individual packets are the main focus. The protocol field, source IP, source port, destination IP, destination port, TCP flags, ICMP type, and ICMP code are the most frequently utilized header fields for classifying packets [14,15]. Since it runs on low-level devices, a packet-filtering firewall is seen as less secure than its competitors. Packet filtering's main flaw is that it mostly relies on the packets themselves to accurately report their origin and destination. Most existing packet filtering algorithms simply consider the properties of the filtering rules rather than the traffic behavior in order to optimize their performance, despite the fact that network traffic includes a huge number of events and a lot of important information [11,16,17]. This is the paper's major issue, and our approach offers a way to address it.

As firewalls must filter all traffic entering and leaving the entire network, they must be capable of supporting a large volume of traffic, or they will become a bottleneck. Matching packets at a firewall may be thought of as a challenge of pinpointing location. In order to locate the first matching rule in a firewall, each packet's fields (dimensions) must be compared to all of the rules. Considering the wide variety of packets, the number and variety of filtering rules used to examine them might be overwhelming. Most rules are blind to the traffic characteristics of these packets; hence, there is room for improvement in the effectiveness of filtering and threat identification. As a result, there has been a shift towards providing methods and utilities that can handle ambiguity in packet filtering scenarios, ultimately aiming to build an intelligent firewall whose actions can adjust to the dynamic nature of networks and available computing resources [10,18].

Fuzzy logic has become a crucial way of handling ambiguity and working with poorly specified or undefined systems. The degree of certainty or ambiguity with which an element belongs to a set is quantified by fuzzy logic. Significant gains may be achieved by integrating fuzzy logic control strategies into current firewall technology. Several researchers use a fuzzy control strategy to enhance existing firewall architectures [19,20]. Neutrosophic logic was employed in recent studies as a usable extension of fuzzy logic. Beyond the ideas of fuzzy logic, the issue of indeterminacy is extensively considered in neutrosophic logic. The neutrosophic set, with its third component of indeterminacy, may be able to capture this ambiguity and provide realistic outcomes [21]. Under the umumbrella of neutrosophic reasoning, it is possible to make the firewall's behavior flexible to increase operating systems' resilience to attacks by considering not only the contents of network packets but also connection status information to make filtering decisions.

Petri nets are a graphical depiction of discrete event systems that may be used as a mathematical description tool. Petri nets have been the subject of a significant amount of research and have seen widespread application because of their notable benefits, which include asynchronous events, conflicts, and resource sharing [22–24]. The mapping of Business Process Modelling Notation (BPMN) components into Petri nets is shown in Figure 1. However, fuzzy data applied to complicated systems with uncertainty is beyond the capabilities of Petri nets. This shortcoming is addressed by integrating fuzzy sets into Petri nets, leading to the creation of a new knowledge model known as Fuzzy Petri Nets (FPNs), which are typically used when modeling information or knowledge with no well-defined measurements [25,26]. In recent years, FPNs have been the subject of substantial study by academics and have found widespread use in a variety of sectors, including fault diagnostics, sequence control, flexible manufacturing systems, and risk prediction [27]. FPNs consist of places, transitions, and directed arcs that represent propositions, fuzzy production rules, and connections between places and transitions, respectively; see [25] for more details.

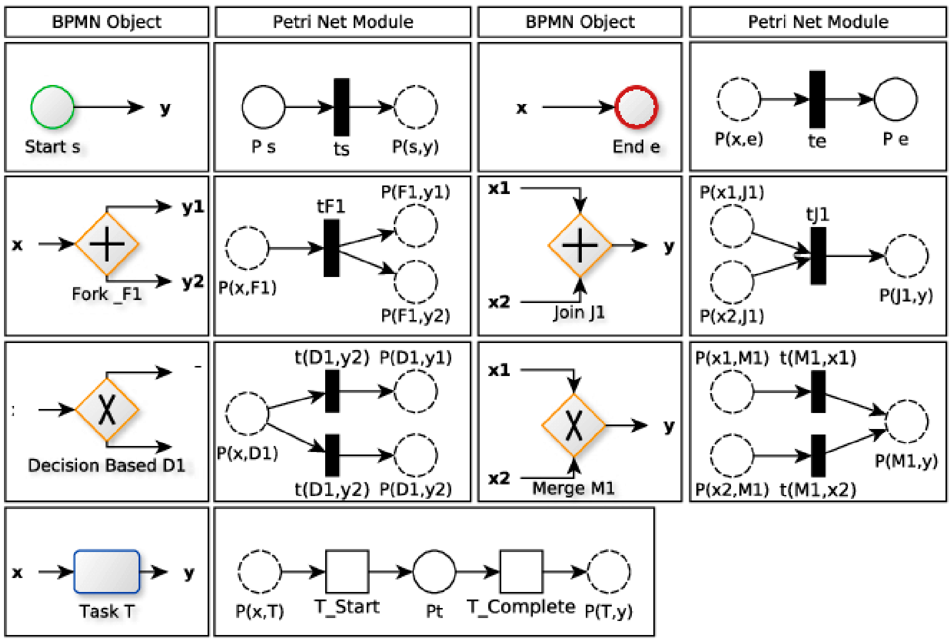

Figure 1. Mapping of Business Process Modeling Notation (BPMN) elements into Petri nets.

It is important to highlight that FPNs and their versions (e.g., interval type-2 FPNs) are powerless to deal with complex applications due to the existence of indeterminate and inconsistent information and the limitations of the knowledge reasoning operators [25–27]. Domain specialists are more likely to assign one of three distinct membership degrees—truth, indeterminacy, or falsity—to convey cognition or experienced knowledge because of the complexity and uncertainty of the practical applications. Simplified neutrosophic sets (SNSs) are more adaptable to uncertainty and cognitive variations in knowledge representation because of the independence of the three membership types of SNSs. Thus, Simplified Neutrosophic Petri nets (SNPNs) are used with SNS as a novel kind of FPN for knowledge representation. It is easier to manage uncertainty and inconsistency using a knowledge representation and reasoning (KRR) paradigm such as this [28].

Contribution

In this research, we provide a novel packet filtering model for firewalls that makes use of the traffic behavior of incoming packets to evaluate the external risk and improve the firewall's packet filtering process. Using the SNPN concept as a strategy to manage uncertainty and inconsistent states improves the firewall's efficiency and performance by allowing it to handle a wider range of packet types, which in turn improves the firewall's ability to detect and prevent malicious activity. This is the first attempt to use SNPNs in the formation of firewall rule sets that describe the operation of firewall technologies. The ability to theoretically describe, simulate, and analyze firewall packet filtering systems is greatly facilitated by SNPNs. Moreover, when comparing the generalized net technique to the neutrosophic Petri net approach that was used, one drawback becomes apparent: the former struggles to effectively express various forms of uncertainty. The definition of the neutrosophic Petri net is characterized by its simplicity, as well as its straightforward algorithm for functioning. This feature facilitates the streamlined development of simulation models [24,26–35].

The remainder of the paper is structured as follows: Recent relevant works are discussed in Section 2. Section 3 provides a comprehensive description of the suggested model. Discussions and simulation results for the syntactic dataset are presented in Section 4. Section 5 concludes the paper with results and implications.

2. State of the Art

Firewalls are divided into two types in the academic literature [36]: (1) static and dynamic firewalls. An administrator establishes and maintains all policies on a static firewall. The rules do not evolve in response to new information or circumstances. Rules in a dynamic firewall may be toggled on or off as needed. Time and traffic conditions are two examples that may inform such regulations. A dynamic firewall, for instance, may add malicious IP addresses to a blacklist in response to a distributed denial-of-service (DDoS) attack. (2) Stateful and stateless firewalls: A stateless firewall does not keep any logs of connections. These are limited to acting exclusively on the basis of the information contained in the IP packet and the previously stated rules. A stateful firewall may remember past events and keep track of currently open connections. The firewall may then make better choices based on this information. If a request was made from an internal client to an external server, the stateful firewall would record this and then let the incoming packet from the server through since it was part of an established connection. Alternatively, the firewall would reject the incoming reply as malicious if it came from an external server without a matching outbound request [11,14].

There have been multiple efforts to advance packet filtering firewall technology for the sake of improving factors such as network safety, efficiency, and service quality. In order to filter packets, a rule designer must first develop a set of criteria against which to check and match incoming data. Since packets and rules may vary and evolve over time, this seems to be the primary filtering challenge [11,37–43]. In Ref. [44], the authors presented a thorough overview of deep packet inspection and other forms of packet analysis used in network forensics, as well as a study of packet analysis, approaches driven by artificial intelligence that can classify and identify complex patterns in network traffic. In this article, we take a look at the capabilities of packet analyzer software and hardware appliances from the point of view of their possible use in network forensics.

For software-defined networks (SDNs), the authors in [45] developed a firewall that takes applications into account. The firewall's core module works in tandem with sub-modules that perform packet filtering, application recognition, and security enforcement. The system monitored data transfer at the network, transport, and application layers and then pushed down protective measures. The research described in [46] primarily focused on real-time packet filtering technologies to restrict service access only to authorized users. The distinctive, enhanced packet marking approach stops the flow of unknown packets at their source, giving users a more manageable wait time before they can begin using the service. In this task, the authors used the NS3 simulator and the Ethereum Blockchain platform to replicate their work. In accordance with the concept of the packet marking approach, after discarding the unknown flooding of packets, the incoming traffic constitutes 90% of the real traffic flow of packets to the blockchain network.

In Ref. [47], a novel strategy was suggested for IP packet filtering by fusing two data structures into a single one that is optimized for IP packet filtering in terms of speed, scalability, and flexibility (including dynamic access to filters and the ability to conduct an approximate membership query). In experiments, the suggested technique achieved a throughput of 10.8 Mbps/s with good filtering accuracy and minimal memory needs, even while processing large filter sets (up to 1 GB). The inefficient use of search space and data storage are two problems plaguing current approaches to packet filtering algorithms. In Ref. [48], the authors offer a fireworks-based method for packet filtering that gets over these limitations. Sparks from the fireworks are used by the so-called fireworks-based packet filtering (FWPF) algorithm to determine which rule in the firewall rule set best fits the incoming packet. When compared to other current packet filtering algorithms, FWPF has the benefit of a smaller search area, which speeds up the discovery process.

In Ref. [49], the authors provided a workable technique for network access control list (ACL) matching trie, which improves the efficiency of the multi-bit stride extension. To test their approach, they created synthetic ACLs to mimic real-world campus networks and Internet backbone restrictions. They also demonstrated that the issue of slow data structure construction times inherent in prior-art data structure-based filtering algorithms is addressed by this new approach. In Ref. [50], the authors compared the efficiency of the cuckoo hash table and the Bloom filter for classifying packets, both of which have practical uses in packet filtering and, more specifically, in legal interception systems. The data plane development kit includes these techniques as part of its library, in addition to making it possible to conduct rapid packet capture using commodity hardware. The cuckoo hash table is an accurate classification approach, whereas the Bloom filter is a probabilistic method that uses less space but may produce false positives, making it more suitable for use in performance-critical applications. The performance advantages of utilizing a probabilistic technique are weighed against its downsides by comparing the execution time and memory requirements of the two approaches.

In Ref. [51], the authors discussed the extended Berkley packet filter (eBPF) platform, which enables code to be dynamically loaded into the Linux kernel. Accelerating networking by letting the kernel handle certain packets without help from a user-space programmer. Recently, eBPF has only been used for basic packet filtering functions such as firewalls and DDoS protection. They demonstrated that it is possible to create a machine-learning-based, flow-based network intrusion detection system solely in eBPF. Their approach uses a decision tree to assess whether or not a packet is malicious, taking the whole context of the network flow into consideration. A speed increase of nearly 19% was realized when comparing this method to its implementation as a user-space programmer.

Unexpected blocking is a common problem when employing firewall technology to stop network traffic. The primary reason for this is that, with the widespread adoption of virtual hosts, a single IP address will serve as the host for several domain names. As a solution, the research described in [52] suggested using Server Name Indication (SNI) data to implement traffic blocking. To begin, a Linux kernel module is built using the infrastructure of net filtering and IP tables. Second, HTTPS communication takes advantage of the Transport Layer Security (TLS) protocol's unencrypted SNI information to prevent domain name traffic. Finally, the Linux kernel is set up to accept batch writes of domain name sets in accordance with the rules of certain IP tables. Their approach can be used to build up domain names and domain name traffic-blocking rules in IP tables.

In Ref. [53], the authors recommended a passive model for detecting anomalies in container networks. The model is based on eBPF technology, which collects process granularity data in the Linux kernel in a non-intrusive manner to obtain network performance data at the granularity of containers, which is then combined with machine learning classification techniques to determine if the container's network performance is abnormal. In order to provide automated detection and defense against synchronized sequence numbers (SYN) flood attacks, the authors in [54] developed a system architecture based on the eXpress Data Path (XDP) and the extended Berkeley Packet Filter (eBPF). Data on the rate of SYN request connections per unit time per IP address and the rate of server retransmission SYN request connections can be extracted by studying the SYN Flood attack's underlying principle and using eBPF to monitor the attack process in the Linux kernel's network protocol stack. The abnormal connection IP address is identified by sending the indicator data corresponding to each IP address to the detection algorithm for analysis and source tracing. Herein, XDP is utilized to protect against malicious connection IPs.

A smart packet filter powered by artificial intelligence is required since current solutions take too long to process incoming packets. The research published in [55] used an ensemble model (smart learner) on a maximum 11-depth random forest. Using the stacked K-gold cross-validation method, the model achieves an average accuracy of 99.76%. Partitioning and converting rule sets for hybrid packet processing are described by the authors in [56]. HyPaFilter+ was suggested as a solution. It is a hybrid classification system that utilizes both a hardware matcher implemented on a Field Programmable Gate Array (FPGA) and a software netfilter firewall implemented in Linux, resulting in a straightforward and efficient mechanism for switching packets between the two. Throughput improvements of up to 30 times were seen in comparison to software-based packet processing.

In Ref. [57], the authors presented a deep packet: a deep learning-based approach that combines feature extraction and classification into a single system capable of traffic characterization (wherein the traffic on a network is divided into broad categories) and application identification (wherein the applications used by end users are searched out). Deep Packet, in contrast to most existing approaches, can detect VPN and non-VPN network traffic, as well as identify encrypted communication. In order to categorize packets, the Deep Packet Framework uses a pair of different kinds of deep neural networks: stacked autoencoders (SAEs) and convolutional neural networks (CNNs). Their tests showed that Deep Packet performs best when using CNN as its classification model, with recalls of 0.98 and 0.94, respectively, for the application identification and traffic categorization tasks.

In Ref. [58], the authors suggested using a new cache-based reconnaissance approach to penetrate Linux-based firewalls that were virtualized. Because of the capability of source spoofing, this method increases stealth against intrusion detection. In addition, it facilitates the deduction of several filtering rules. Using just a small sample size of packets, this approach was able to infer the firewall rules with an accuracy of over 90% during an experiment conducted on the open-source router and firewall platform (VyOS). In addition, they demonstrated remedies to prevent cache-based attacks on virtualized network functions. In Ref. [59], the authors create a blacklist packet filter for WSNs that is based on the blockchain and has collaborative intrusion detection. A strong blacklist for limiting unwanted traffic is constructed with the use of blockchain technology. Using a real-world dataset and a realistic WSN environment, they analyzed how well their filter performed. The results showed that the proposed filter may strengthen blacklist creation.

The authors in [60] developed an effective firewall that makes use of a packet categorization algorithm based on a Learned Cuckoo Filter (LCF) to reduce the method's impact on the network. The search approach used by this algorithm is hierarchical. By using the machine learning model as a pre-filter before the cuckoo filter, the algorithm is able to reduce memory requirements and boost speed by avoiding the needless search for rules in the subsequent step. Extensive simulations confirmed the presented design's advantages with regard to false positive rate, memory usage, and memory access. In Ref. [61], a simulation model was presented for measuring the effectiveness of different filtering rules in a firewall. Different simulation model settings and network traffic behavior scenarios were used to assess the method's efficacy in establishing a filtering rule set. Table 1 summarizes the major categories that recent packet filtering firewall systems follow from the perspective of their strengths and limitations.

Table 1. Summary of state-of-the-art packet filtering firewall systems.

Packet Filtering Firewall Systems	Approach	Strengths	Limitations
Deep packet inspection [44–46,55–57]	Study of packet analysis approaches driven by artificial intelligence.	1. Easy to install. 2. Faster than other firewall technologies because they perform fewer evaluations.	1. Difficulty in setting up packet filtering rules for the router. 2. There is not any sort of user-based authentication.
Cuckoo filter-based IP packet filtering [47,50,60]	Fusing two data structures into a single one that is optimized for IP packet filtering in terms of speed, scalability, and flexibility.	1. Packet filters make use of current network routers. 2. Makes security transparent to end-users.	1. Difficulty in setting up packet filtering rules for the router. 2. There is not any sort of user-based authentication.
A Fireworks-Based Approach [48,49]	Sparks from the fireworks are used to determine which rule in the firewall rule set best fits the incoming packet.	1. Ease of use. 2. Cost effective.	1. User restriction. 2. Malware attack. 3. Difficulty in updating ACL.
Berkley packet filter platform [51,53,54]	Uses a decision tree to assess whether or not a packet is malicious, taking the whole context of the network flow into consideration.	1. Packet filters make use of current network routers. 2. Cost effective.	1. Packet filters do not understand application layer protocols. 2. Packet-filtering routers are not very secure.
Blacklist packet filter for WSNs [59]	A strong blacklist for limiting unwanted traffic is constructed with the use of blockchain technology.	1. Promotes privacy and security. 2. Monitors network traffic.	1. Lack of logging capabilities. 2. Challenging setup. 3. New rules may need to be added if an employee needs special requirements to connect to the internet.

Research Gap

We think there are many unexplored opportunities in the field of intelligence firewalls despite the numerous different packet filtering approaches now in use. This study introduces a novel model for intelligent packet filtering, one that takes advantage of SNPN's capabilities to create a double-tiered filtering architecture. With the help of risk levels, the first layer of packet filtering is able to recognize malicious packets and prevent them, while the second layer aims to boost filtering performance based on the rating of accepted and rejected packets. For fuzzy knowledge representation and reasoning, the SNPN theory provides tools for dealing with imprecise and ambiguous packets' information. In fact, the SNPN model's implementation enables the use of essential features such as testing for correctness, circular rules, consistency, and completeness. To represent the experience and knowledge of subject matter experts, independent truth, indeterminacy, and falsity membership functions define SNPNs.

3. Methodology

In order to effectively secure networks, firewalls must meet two competing requirements [62]: (1) They must be able to classify network packets at line speed. To avoid the firewall from becoming a bottleneck for network access, what throughput, maximum simultaneous connections, connections per second, and latency criteria must be satisfied for both current and future traffic needs; (2) They must be able to process packets in a variety of ways to accommodate varying filtering policies. Packet-filtering routers have restricted access to data. While packets know their host, they do not know who is using them. As a result, it is impossible to place limitations on certain users. High-level protocols provide for control over which users have access to the ports. To guarantee that no one else is using that port, these methods lock it down.

Systems constructed on special-purpose hardware are quick but restricted in their filtering capabilities, and unfortunately, most present classification systems do not satisfy both needs equally well. While software filters offer robust matching semantics, they often fall short when it comes to line speed. This justifies combining a slower but more flexible software firewall with faster but more sophisticated specialized hardware. The benefits of packet filtering firewalls as a network security solution are their low complexity, low cost, high speed, low resource consumption, predictability, and determinism. However, it has drawbacks, such as being ineffective against modern attacks, being vulnerable to spoofing, being unable to make decisions based on application or authentication, and rule lists growing too large to manage [63]. Existing packet classification methods still have issues with poor classification performance and huge storage space despite the fact that a lot of research has been conducted in this field.

The suggested model uses SNPNs as a visual representation of neutrosophic logic-based firewall packet management. Theoretically, the computing capability of PNs, FPNs, and SNPNs is the same; nevertheless, SNPNs have much greater modeling power because of their superior structural facilities. For any PN object (transition, place, or arc), neutrosophic logic may be used to construct logical expressions and functions. Specifically, we use the truth-membership function, the indeterminacy-membership function, and the falsity-membership function to describe the uncertainty in a packet's movement, a concept borrowed from neutrosophic logic for modelling PN transition objects.

To filter Internet traffic, neutrosophic logic was used at two levels: first, to evaluate the potential threat posed by incoming packets from the Internet; and second, to reorganize the access control list (ACL) by assigning acceptance and rejection ratings to each packet. ACLs are often established by the network administrator, who determines the permissions granted to various users and network resources based on a predetermined set of rules designed to achieve a number of security goals [64,65]. For simplicity, the suggested model relies on standard ACL (see Figure 2); for further information on the various types of ACL, the reader can refer to Ref. [64]. The packet's acceptance or rejection from the network's perspective depends on the rule set of access control lists. Packet filters are responsible for sorting data packets into predetermined categories. Table 2 displays an example of a typical access control list, which includes multiple filtering criteria. In general, the rules are evaluated in a top-down fashion. Figure 3 shows the high-level data flow diagram representing the two-level neutrosophic packet filtering system developed here, which offers superior filtering performance from a security perspective. Below, we have laid out all the steps involved in the proposed model in detail.

Table 2. An example of ACL.

No	Source IP	Destination IP	Source Port	Destination Port	Protocol Type	Action
R1	203.117.102.15	193.170.92.3	*	*	ICMP	Allow
R2	203.117.175.6	193.170.75.7	*	*	UDP	Allow
R3	203.117.175.4	193.170.62.29	*	*	TCP	Deny

* Mean any source/destination port.

Standard ACL
Dynamic ACL
Extended ACL
Reflexive ACL

Figure 2. Access control list categories.

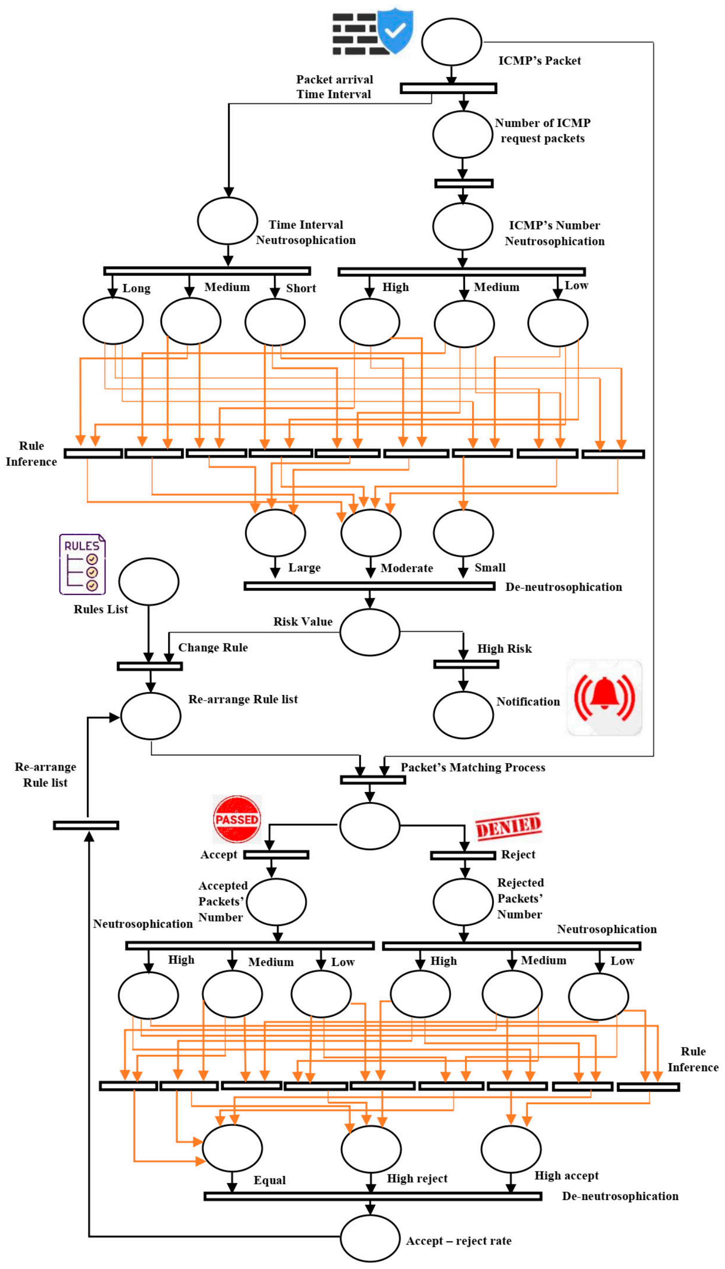

Figure 3. The proposed 2-level neutrosophic packet filtering model.

Formally, an SNPN structure is defined as an 11-tuple [28]

$$SNPNs = (P, T, I, O, \alpha, \beta, M, U, W, T) \quad (1)$$

where

- $P = \{p_1, p_2, \ldots\ldots, p_m\}$ is a finite set of places;
- $T = \{t_1, t_2, \ldots\ldots, t_n\}$ is a finite set of transitions;
- $D = \{d_1, d_2, \ldots\ldots, d_n\}$ is a finite set of propositions;
- $I = P \to T$ is an input incidence matrix, which defines a mapping from places to transitions;
- $O = T \to P$ is an output incidence matrix, which defines a mapping from transitions to places;
- $\alpha = P \to [0,\ 1]$ is an association function, which maps places to real values between 0 and 1;
- $\alpha = P \to D$ is a mapping between places and propositions;
- $M = P \to \Omega$ is the truth values of all places, expressed by the vector $M = (\alpha_1, \alpha_2, \ldots., \alpha_m)^T$, which map from places to simplified neutrosophic number (SNNs); and α_i is an SNN. The initial marking vector is denoted by M_0;
- $U = T \times P$ is a vector expressed by $U = (\mu_1, \mu_2, \ldots., \mu_n)^T$; μ_i is the certainty factor of transition t_i; and n is the numbers of transitions.
- $W = P \times T$ assigns a value to each arc between input place and transition, expressed by $W = [w_{ij}]_{m \times n}$ whose element $w_{ij} \in [0,\ 1]$ indicates how much the place p_i affects the following transition t_i, with $\sum_{i=1}^{m} w_{ij} = 1, j = 1, 2, \ldots.n$.
- T assigns a threshold value λ_i to each transition, expressed by $T = (\lambda_1, \lambda_2, \ldots., \lambda_n)$ whose elements take the form of simplified neutrosophic number (SNN).

An SNN A in U (a finite set of objects) is depicted by three functions including the truth-membership function $T_A(x)$, indeterminacy-membership function $I_A(x)$, and falsity-membership function $F_A(x)$, as illustrated in Figure 4 where a_T, b_T, and c_T are the parameters of the truth-membership function; a_I, b_I, and c_I are the parameters of indeterminacy membership function; and a_F, b_F, and c_F are the parameters of falsity-membership function. These parameters will be determined by the proposed framework. $T_A(x)$, $I_A(x)$ and $F_A(x)$ are all the standard subsets of $[0,\ 1]$. An SNS can be represented by the following [28]:

$$A = \{\langle x,\ T_A(x), I_A(x),\ F_A(x)\rangle | x \in U\} \tag{2}$$

where $T_A(x) \in [0,\ 1]$, $I_A(x) \in [0,\ 1]$, $F_A(x) \in [0,\ 1]$, and $0 \le\ T_A(x) +\ I_A(x) +\ F_A(x) \le 3$. In SNPNs, if a transition $t_i \in T$ is enabled (enabling rule), two conditions should be satisfied, including (a) the value of place is not negative; (b) the equivalent input value of transition is greater than or equal to the threshold value.

$$\begin{cases} \alpha(p_{ij}) \ge 0 \\ \theta(t) \ge \lambda_{ij} \end{cases} j = 1, 2, \ldots\ldots, m; j = 1, 2, \ldots\ldots, n \tag{3}$$

$\alpha(p_{ij})$ is the value of place p_{ij}, and it denotes the truth degree of proposition d_{ij}; and $\theta(t) = SNNsHA_w(\alpha(p_{j1}),\ \alpha(p_{j2}), \ldots\ldots \alpha(p_{jm}))$ is the input value of a transition t calculated by the SNNs hybrid averaging (SNNsHA) operator. The SNNsHA operator can not only weigh the SNNs but also weigh their ordered positions.

$$\text{SNNsHA}_w(A_1, A_2, \ldots\ldots, A_n) = \sum_{j=1}^{n} w_i A''_j\ = \langle 1 - \prod_{j=1}^{n}\left(1 - T_{A''_j}\right)^{w_j}, \prod_{j=1}^{n} I^{w_j}_{A''_j}, \prod_{j=1}^{n}, F^{w_j}_{A''_j}\rangle \tag{4}$$

A''_j is the j-th largest element of the weighted SNNs $(nw_1A_1, nw_2A_2, \ldots\ldots, nw_nA_n)$; the weight vector $w = (w_1, w_2, \ldots.., w_n), w \in [0,\ 1]$ and $\sum_{i=1}^{n} w_i = 1$; and n is the balancing factor.

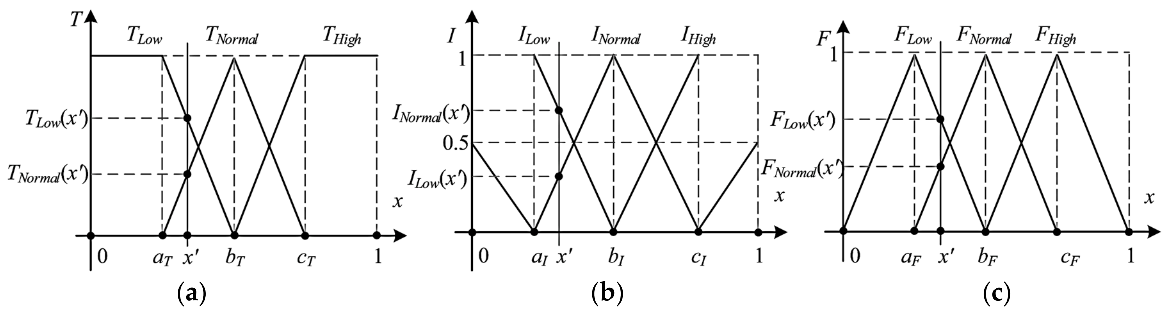

Figure 4. (**a**) Truth, (**b**) indeterminacy, and (**c**) falsity-membership functions.

3.1. Level 1 Filtering

In order to mimic and track the packet flow, this level depends on collecting and classifying all arriving packets based on the data they contain, such as their IP address, packet time, and protocol type. In our framework, a packet is symbolized by a token in an SNPN place, and the operation of a packet is demonstrated by an SNPN transition; it is responsible for transmitting the packet from one location to another. The packet is moved to a place where it is checked and compared to the ACL after being picked up by a gate (place). In addition, a copy (replica) of this packet is sent immediately to the traffic investigation phase so that parameters (features) may be extracted from it, such as the total number of ICMP packets received during the time frame in question. The neutrosophic logic engine that generates the risk level relies on these two factors (packet count and time period) as inputs. Threats posed by the movement of packets originating from untrusted sources are reflected in this risk level.

Due to its widespread usage by attackers at all stages of system penetration, the ICMP protocol is the focus of the proposed framework. Traffic-based attacks such as ICMP flood attacks utilize huge demands on servers to disrupt their regular operations. In addition, ICMP was exploited for system compromise and utilized as a backdoor by attackers in certain cases to communicate with one another (covert channel) [66]. The proposed framework is flexible enough to handle attacks using non-TCP protocols, such as TCP SYN and UDP Flood. IP packets containing UDP datagrams are sent by an attacker in an attempt to overwhelm a target system until it is unable to process legitimate connections, a phenomenon known as User Datagram Protocol (UDP) flooding [67]. SYN-Flood attacks are distinguished by the attackers' use of faked source IP addresses in a barrage of TCP SYN request packets. This causes the server to use excessive resources to keep track of a high number of partially open connections, which might prevent it from providing typical services in the long term [68].

The number of ICMP echo-request packets P_{no} and packet arrival time interval P_t were chosen because they are simple and considered in the majority of attack defense scenarios, particularly when we have a significant number of echo requests. For the membership grading function (MF) to be usable in the neutrosophic controller in use, feature vector measures must be transformed into the range [0, 1] via the Gaussian normalization method. When it comes to packet filtering, neutrosophic logic (NL) is perhaps the most effective and flexible way of dealing with a mix of measurements that range in uncertainty. NL is a theory that permits issues to be addressed using just natural descriptions in language, as opposed to only numerical numbers [69–71]. Neutrosophic controllers consist of four modules. First, data on all relevant aspects of the process under control are collected. Second, we capture the measures' truth, falsehood, and indeterminacy via neutrosophic set membership functions (neutrosophication phase). Third, the neutrosophied measurements are used by the inference engine to evaluate the control rules in the neutrosophic rule base. The last module of the cycle, known as the de-neutrosophication phase, reduces this neutrosophic set to a single (crisp) value that best represents the derived set [72].

The goal is to use a neutrosophic model to represent the uncertainty of packet features. In this case, we use a neutrosophic controller with two inputs and a single output, defined as

$f : U \subset R^n \longrightarrow V \longrightarrow R^n$ where $U = U_1 \times U_2$ is the input space and V is the output space. The characteristic of P_{no} is described using three neutrosophic variables: low, medium (normal), and high; the feature of P_t is described using three neutrosophic variables: long, medium, and short; and the characteristic of risk level R is described using three neutrosophic variables: small, moderate, and large. Their respective triangular membership functions are shown in Figure 4 [28]. The corresponding equations are expressed by Equations (5)–(13). All of the membership function's parameters are quantitatively derived based on past experiences to assess the impending risk level via packet traffic.

$$T_{low} = \begin{cases} 1, \quad 0 \le x < a_T \\ \frac{b_T - x}{b_T - a_T}, \quad a_T \le x < b_T \\ 0, \ otherwise \end{cases} \tag{5}$$

$$T_{medium} = \begin{cases} \frac{x - a_T}{b_T - a_T}, \quad a_T \le x < b_T \\ \frac{c_T - x}{c_T - b_T}, \quad b_T \le x < c_T \\ 0, \ otherwise \end{cases} \tag{6}$$

$$T_{high} = \begin{cases} \frac{x - b_T}{c_T - b_T}, \quad b_T \le x < c_T \\ 1, \quad c_T \le x \le 1 \\ 0, \ otherwise \end{cases} \tag{7}$$

$$I_{low} = \begin{cases} \frac{a_I - x}{2a_I}, \quad 0 \le x < a_I \\ \frac{x - a_I}{b_I - a_I}, \quad a_I \le x \le b_I \\ 0, \ otherwise \end{cases} \tag{8}$$

$$I_{medium} = \begin{cases} \frac{b_I - x}{b_I - a_I}, \quad a_I \le x < b_I \\ \frac{x - b_I}{c_I - b_I}, \quad b_I \le x < c_I \\ 0, \ otherwise \end{cases} \tag{9}$$

$$I_{high} = \begin{cases} \frac{c_I - x}{c_I - b_I}, \quad b_I \le x < c_I \\ \frac{x - c_I}{2(1 - c_I)}, \quad c_I \le x < 1 \\ 0, \ otherwise \end{cases} \tag{10}$$

$$F_{low} = \begin{cases} \frac{x}{a_F}, \quad 0 \le x < a_F \\ \frac{b_F - x}{b_F - a_F}, \quad a_F \le x < b_F \\ 0, \ otherwise \end{cases} \tag{11}$$

$$F_{medium} = \begin{cases} \frac{x - a_F}{b_F - a_F}, \quad a_F \le x < b_F \\ \frac{c_F - x}{c_F - b_F}, \quad b_F \le x < c_F \\ 0, \ otherwise \end{cases} \tag{12}$$

$$F_{high} = \begin{cases} \frac{x - b_F}{c_F - b_F}, \quad b_F \le x < c_F \\ \frac{1 - x}{1 - c_F}, \quad c_F \le x \le 1 \\ 0, \ otherwise \end{cases} \tag{13}$$

In order for the system to infer the risk of the packets, it must first learn the neutrosophic descriptions of the features included inside them. The existence or lack of association or interaction between the components of two or more sets is presented as a degree in neutrosophic reasoning, which is expressed by a collection of neutrosophic IF–THEN rules. In the same manner that people think, the nine rules collectively address the suggested weight allocations. The decision is more reasonable since the neutrosophic inference considers all nine situations simultaneously. The AND operator is used to obtain a single value when more than one part of an antecedent occurs in a rule. When this is carried out, there is only one possible value for truth. In this setting, min (minimum) represents the AND operator. The following rules, shown in Table 3, govern the application of logic in the proposed model.

Table 3. Neutrosophic logic-based packets' risk determination: a set of basic rules.

Rule	ICMP-Echo-Rate P_{no}	Time Duration P_t	Risk Level R
1	High	Long	Moderate
2	Medium	Long	Moderate
3	Low	Long	Small
4	High	Short	Large
5	Medium	Short	Large
6	Low	Short	Moderate
7	High	Medium	Large
8	Medium	Medium	Moderate
9	Low	Medium	Moderate

Next, a crisp value for the variable is calculated by de-neutrosophicating the neutrosophic outputs. Linear trapezoidal neutrosophic numbers may be de-neutrosophicated using the same area reduction method as is used for single-valued trapezoidal neutrosophic numbers; see Ref. [73] for more details. The initial ACL configuration and writing are the responsibility of the network administrator. If a packet matches a rule, that rule triggers an action. The actions specify whether the packet should be blocked "deny" or allowed "allow" across a certain interface. To improve packet filtering efficiency, the proposed system uses the obtainable risk level to adjust the action of ACL rules that, in turn, reflect the valid and trustable address list and determine the authentication level of all IP addresses. If the risk value is higher than the critical degree (threshold $T \geq 0.75$) set by the security administrator, the blacklisting rules switch from accept to deny, causing previously accepted packets to be dropped, as can be shown in Table 4. In contrast, the white-listing process does not affect the risk level beyond a certain threshold. In our case, the default value was determined according to preliminary experiments conducted to verify the best value of T to achieve a balance between the two goals of classifying network packets at line speed and processing packets in various ways to accommodate various filtering policies.

Table 4. An example of ACL after updating with risk level.

No	Source IP	Destination IP	Source Port	Destination Port	Protocol Type	Risk Value	Action
R1	203.117.102.15	193.170.92.3	*	*	ICMP	0.83	Block
R2	203.117.175.6	193.170.75.7	*	*	UDP	0.88	Block
R3	203.117.175.4	193.170.62.29	*	*	TCP	0.25	Block

* Mean any source/destination port.

3.2. Level-2 Filtering

Each firewall has its own set of packets that it accepts and its own set of packets that it rejects. Our solution takes use of this fact to improve packet filtering efficiency by using level-2 fuzzy filtering to track the acceptance/rejection rate of packets and reorganize ACL rules to reduce the amount of time it takes for rules to be matched. Also, a neutrosophic model is used in an effort to represent the uncertainty in the packets acceptance rate R_A or packets rejection rate R_R. A neutrosophic controller with only two inputs R_A and R_R and one output, calculated rate R_C, is employed here. The characteristics of both R_A and R_R are described using three neutrosophic variables: low, medium (normal), and high; the characteristic of calculated rate R_C that characterizes the rejection and acceptance rates in packets traffic is described using three neutrosophic variables: high reject, equal, and high accept. Their respective triangular membership functions are shown in Figure 4. To estimate the packet traffic's rejection and acceptance rates, all of the membership function's parameters are numerically determined based on experience. The following rules, shown in Table 5, are applied to the suggested model approach for reasoning. In the same manner, the de-neutrosophicated step is started to generate a crisp value for the variable. Here, if R_C = high accept, any rules whose actions include 'allow' are shuffled and placed at the front of the ACL, giving them the highest priority to run. In contrast, if R_C = high reject, the rules that have a 'block' action are shifted to the beginning of the ACL, giving them the greatest priority to satisfy, while the rules that have a 'allow' or 'block' action are fixed to the end of the ACL.

Table 5. Neutrosophic logic-based packets' traffic rate determination: a set of basic rules.

Rule	Acceptance Rate R_A	Rejection Rate R_R	Calculated Rate R_C
1	High	Low	High Accept
2	High	High	Equal
3	High	Medium	High Accept
4	Low	Low	Equal
5	Low	High	High Reject
6	Low	Medium	High Reject
7	Medium	Low	Equal
8	Medium	High	High Reject
9	Medium	Medium	Equal

In conclusion, the proposed model can dynamically alter the order of rules to reflect their highest priority based on the acceptance and rejection rates of packets, as well as the ability to change the rules' activities in two distinct phases, depending on the risk level of the incoming traffic. The suggested packet filtering model is resilient to network traffic attacks because it uses neutrosophic logic to deal with the uncertainty associated with system variables such as packet risk level and packet acceptance/rejection rate. Expert opinions were linked to linguistic factors in the proposed model using a neutrosophic method. These linguistic variables reflect the expert opinions more precisely.

4. Results and Discussions

The suggested method was implemented in Python for testing purposes. We utilized a modular approach to building the model and showed how the hierarchical ideas behind SNPNs can be used to integrate several firewall mechanisms into a single, powerful one. There are five distinct types of information that may be filtered out: IP address ranges, ports, protocols, and source and destination IP addresses. In the current version of the suggested model, the fuzzy rules in Tables 3 and 5 are defined according to what was presented in related work in Refs. [74–77]. The experiments were conducted on a laptop with an X64-based processor and 8 GB of DDR3 memory on an Intel® CoreTM i7-5500 CPU running at 2.50 GHz. Standard TCP, UPD, and ICMP protocols were used in the simulation to evaluate the proposed firewall packet filtering. The suggested model was tested using a variety of settings. We evaluate this model offline by utilizing Network Simulator 11.3 software (http://fr.lagache.free.fr/netsim/exemple_filtrage.php?lang=en, accessed on 1 March 2023). In this case, after we executed the prototype version of our model to build the ACL with re-configured rules, these rules were used within the simulator to filter the network packets. An evaluation of the suggested method vs. fuzzy and non-fuzzy filtering strategies is also provided. What follows is a discussion of the evaluation's results.

4.1. Experiment 1: (Computational Cost)

- The Aim

Table 6 compares the time it took for a group of data packets to pass through the firewall before and after the firewall rules in the ACL were optimized by rearranging them. The purpose is to evaluate the difference in processing times between randomly sequencing rules (the traditional firewall packet filtering strategy) and optimally reordering them (our suggested model). Our filtering method reorders the filtering fields (actions) from the beginning of the ACL.

Table 6. Computational cost comparison (ms).

No. of Packets	**500**	**1000**	**2000**	**3000**	**4000**	**5000**
ACL Random Sequence	40	60	110	200	280	320
ACL Rearranged Sequence (suggested model)	15	25	35	60	75	90

- Findings

According to Table 6, the time it takes for data packets to get through the firewall was reduced by around a third after filtering fields were optimized (reordered). More rules lead to a more significant outcome in terms of response time.

- Justification

The results demonstrate that the suggested algorithm significantly increases the effectiveness of firewalls by reducing the number of rule comparisons. The results also show that the system can handle large rule sets.

4.2. Experiment 2: (Comparative Study)

- The Aim

Experiment 2 compares the suggested approach to non-fuzzy and fuzzy firewall filtering methods. In the fuzzy firewall filtering method, fuzzy Petri Net is utilized as a tool for modeling discrete event systems characterized by imprecise knowledge. The acceptance and rejection rates of the package are the subject of this analysis. In the first scenario, each user sends ten legitimate and malicious packets. The firewall will decide what happens when the filtering rules detect certain behaviors (reject or allow). We then examined the percentage of legitimate packets that were accepted. Tables 7 and 8 provide the results of a comparison of the acceptance and rejection rates according to different thresholds T. In the second scenario, we examined the rate at which malicious packets were rejected; the results are shown in Table 9.

Table 7. Analyzing the comparative acceptance rate of legitimate packets (average).

Threshold	**0**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Neutrosophic-based filtering	0.94	0.93	0.91	0.90	0.85	0.83
Fuzzy-based filtering	0.71	0.71	0.67	0.67	0.65	0.56
Traditional filtering	0.52	0.51	0.38	0.34	0.29	0.15

Table 8. Analyzing the comparative rejection rate of legitimate packets (average).

Threshold	**0**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Neutrosophic-based filtering	0.06	0.08	0.10	0.12	0.14	0.17
Fuzzy-based filtering	0.32	0.35	0.39	0.42	0.42	0.42
Traditional filtering	0.62	0.64	0.65	0.68	0.71	0.83

Table 9. Analyzing the comparative rejection rate of malicious packets (average).

Threshold	**0**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Neutrosophic-based filtering	0.97	0.97	0.97	0.92	0.91	0.89
Fuzzy-based filtering	0.82	0.84	0.84	0.86	0.86	0.87
Traditional filtering	0.62	0.62	0.64	0.65	0.71	0.78

- Findings

According to the results, the suggested model is quite good at identifying legitimate packets and avoiding malicious packets. The results of neutrosophic logic-based filtering are consistently superior to those of other filtering policies with regard to uncertainty connected with packet traffic behavior. Despite their popularity, FPNs have shortcomings in the realm of security engineering due to the fact that it is difficult to express many forms of uncertainty.

- Justification

Our method more accurately represents the ambiguity of a packet's movement thanks to the truth-membership function, the indeterminacy-membership function, and the falsity-membership function, which are all realizations of the neutrosophic logic for modelling PN (SNPNs) transition objects.

5. Conclusions

Firewalls are a crucial part of any secure network infrastructure. In recent years, researchers have focused on how to optimize packet filtering. However, new inventive approaches are clearly required to help filtering devices such as firewalls keep up with increased capabilities for preventing attacks on network traffic. This research offers a novel method for optimizing packet filtering in network security rules by using information (statistics) about online traffic. We believe that incorporating fuzzy logic control methods into existing firewall technology will provide significant improvements. It is the first time a neutrosophic Petri net has been used to create and enhance firewall rules. Our firewall optimization tool takes into account both rule sets and traffic. A formalism known as neutrosophic Petri nets is used to explain the operation of firewall technology in this system. To describe, compose, simulate, and analyze firewall systems, SNPNs give us the theoretical foundation and tools necessary to do so. The suggested system's pioneering feature is the employment of two levels of neutrosophic reasoning-based filtering appropriate for network traffic behavior to manage varying levels of uncertainty connected to packet contents, with the goal of improving the security of firewalls and their efficiency via rule optimization.

The prototype implementation of our firewall system indicates that it is suitable for deployment in real-world networking. Together with previous successes in rule optimization, it demonstrates the promise of our research results in enhancing the efficacy of firewalls. The findings show that the proposed model successfully distinguishes between genuine and malicious packets. When it comes to filtering packet traffic under conditions of uncertainty, the results from neutrosophic logic-based filtering regularly outperform those of conventional filtering methods. Furthermore, after optimizing (reordering), the firewall filtering model cut the time it takes for data packets to get through the firewall by about a third. Response times become more important as the number of rules increases.

The subject of whether desired features of firewall systems may be stated as dynamic properties and then validated by SNPNs is one that might be explored in the future. Furthermore, comparing the present approach with the one based on generalized nets as an extension of Petri nets and the intuitionistic fuzzy set approach instead of the neutrosophic sets [78–82] might be explored, in addition to making more comparisons with the current methods [83–92] to show the efficiency of the proposed system.

Author Contributions: Conceptualization, S.M.D. and O.A.H.; methodology, S.M.D., O.A.H. and J.K.M.; software, J.K.M., Z.H.N. and S.K.E.; validation, S.M.D., O.A.H. and J.K.M.; formal analysis, S.M.D., Z.H.N. and S.K.E.; investigation, S.M.D., J.K.M., Z.H.N. and S.K.E.; resources, J.K.M., Z.H.N. and S.K.E.; data curation, J.K.M., Z.H.N. and S.K.E.; writing—original draft preparation, S.M.D. and O.A.H.; writing—review and editing, S.M.D., J.K.M. and Z.H.N.; visualization, J.K.M., Z.H.N. and S.K.E.; supervision, S.M.D. and O.A.H.; project administration, J.K.M., Z.H.N. and S.K.E.; funding acquisition, J.K.M., Z.H.N. and S.K.E. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Geismann, J.; Bodden, E. A systematic literature review of model-driven security engineering for cyber–physical systems. *J. Syst. Softw.* **2020**, *169*, 110697. [CrossRef]
2. Mikko, H.; Nyman, L. The internet of (vulnerable) things: On Hypponen's law, security engineering, and IoT legislation. *Technol. Innov. Manag. Rev.* **2017**, *7*, 5–11.
3. Aljawarneh, S.; Alawneh, A.; Jaradat, R. Cloud security engineering: Early stages of SDLC. *Future Gener. Comput. Syst.* **2017**, *74*, 385–392. [CrossRef]
4. Anderson, R. *Security Engineering: A Guide to Building Dependable Distributed Systems*; John Wiley & Sons: Hoboken, NJ, USA, 2020.
5. Prabakaran, S.; Ramar, R.; Hussain, I.; Kavin, B.; Alshamrani, S.; AlGhamdi, A.; Alshehri, A. Predicting attack pattern via machine learning by exploiting stateful firewall as virtual network function in an SDN network. *Sensors* **2022**, *22*, 709. [CrossRef]
6. Bringhenti, D.; Marchetto, G.; Sisto, R.; Valenza, F.; Yusupov, J. Automated firewall configuration in virtual networks. *IEEE Trans. Dependable Secur. Comput.* **2022**, *20*, 1559–1576. [CrossRef]
7. Aljabri, M.; Alahmadi, A.; Mohammad, R.; Aboulnour, M.; Alomari, D.; Almotiri, S. Classification of firewall log data using multiclass machine learning models. *Electronics* **2022**, *11*, 1851. [CrossRef]
8. Liang, J.; Kim, Y. Evolution of firewalls: Toward securer network using next generation firewall. In Proceedings of the IEEE Annual Computing and Communication Workshop and Conference, Las Vegas, NV, USA, 26–29 January 2022; pp. 0752–0759.
9. Bringhenti, D.; Valenza, F. Optimizing distributed firewall reconfiguration transients. *Comput. Netw.* **2022**, *215*, 109183. [CrossRef]
10. Amal, M.; Venkadesh, P. H-DOCTOR: Honeypot based firewall tuning for attack prevention. *Meas. Sens.* **2023**, *25*, 100664. [CrossRef]
11. Mukkamala, P.; Rajendran, S. A survey on the different firewall technologies. *Int. J. Eng. Appl. Sci. Technol.* **2020**, *5*, 363–365. [CrossRef]
12. Kim, S.; Yoon, S.; Narantuya, J.; Lim, H. Secure collecting, optimizing, and deploying of firewall rules in software-defined networks. *IEEE Access* **2020**, *8*, 15166–15177. [CrossRef]
13. Chao, C.; Yang, S. A Novel Mechanism for Anomaly Removal of Firewall Filtering Rules. *J. Internet Technol.* **2020**, *21*, 949–957.
14. Ullah, K.; Rashid, I.; Afzal, H.; Iqbal, M.; Bangash, Y.; Abbas, H. SS7 vulnerabilities—A survey and implementation of machine learning vs rule based filtering for detection of SS7 network attacks. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 1337–1371. [CrossRef]
15. Bagheri, S.; Shameli-Sendi, A. Dynamic firewall decomposition and composition in the cloud. *IEEE Trans. Inf. Forensics Secur.* **2020**, *15*, 3526–3539. [CrossRef]
16. Khairi, H.; Ariffin, S.; Latiff, N.; Yusof, K.; Hassan, M.; Rava, M. The impact of firewall on TCP and UDP throughput in an open flow software defined network. *Indones. J. Electr. Eng. Comput. Sci.* **2020**, *20*, 256–263.
17. Hakani, D. A Survey on Firewall for cloud security with Anomaly detection in Firewall Policy. In Proceedings of the International Conference on Artificial Intelligence and Smart Communication, Greater Noida, India, 27–29 January 2023; pp. 825–830.
18. Mambetov, S.; Begimbayeva, Y.; Joldasbayev, S.; Kazbekova, G. Internet threats and ways to protect against them: A brief review. In Proceedings of the International Conference on Cloud Computing, Data Science & Engineering, Noida, India, 19–20 January 2023; pp. 195–198.
19. Apiecionek, L.; Biedziak, M. Fuzzy Adaptive Data Packets Control Algorithm for IoT System Protection. *J. Univers. Comput. Sci.* **2020**, *26*, 1435–1454. [CrossRef]
20. Watkins, L.; Ballard, J.; Hamilton, K.; Chow, J.; Rubin, A.; Robinson, W.; Davis, C. Bio-Inspired, Host-based Firewall. In Proceedings of the International Conference on Computational Science and Engineering, Guangzhou, China, 29 December 2020–1 January 2021; pp. 86–91.
21. Hassan, M.; Darwish, S.; Elkaffas, S. An Efficient Deadlock Handling Model Based on Neutrosophic Logic: Case Study on Real Time Healthcare Database Systems. *IEEE Access* **2022**, *10*, 76607–76621. [CrossRef]

22. Yu, W.; Ding, Z.; Liu, L.; Wang, X.; Crossley, R. Petri net-based methods for analyzing structural security in e-commerce business processes. *Future Gener. Comput. Syst.* **2020**, *109*, 611–620. [CrossRef]
23. Ben Attia, H.; Kahloul, L.; Benhazrallah, S.; Bourekkache, S. Using hierarchical timed colored petri nets in the formal study of TRBAC security policies. *Int. J. Inf. Secur.* **2020**, *19*, 163–187. [CrossRef]
24. Tiwari, N.; Hubballi, N. Secure Socket Shell Brute Force Attack Detection with Petri Net Modeling. *IEEE Trans. Netw. Serv. Manag.* **2023**, *20*, 697–710. [CrossRef]
25. Liu, H.; You, J.; Li, Z.; Tian, G. Fuzzy Petri nets for knowledge representation and reasoning: A literature review. *Eng. Appl. Artif. Intell.* **2017**, *60*, 45–56. [CrossRef]
26. Lin, Y.; Yang, C.; Wang, S.; Chiou, G.; Shen, V.; Tung, Y.; Shen, F.; Cheng, H. Development and evaluation of an intelligent system for calibrating karaoke lyrics based on fuzzy Petri nets. *Appl. Artif. Intell.* **2022**, *36*, 2110699. [CrossRef]
27. Shi, H.; Wang, L.; Li, X.; Liu, H. A novel method for failure mode and effects analysis using fuzzy evidential reasoning and fuzzy Petri nets. *J. Ambient Intell. Humaniz. Comput.* **2020**, *11*, 2381–2395. [CrossRef]
28. Yue, W.; Wan, X.; Li, S.; Ren, H.; He, H. Simplified Neutrosophic Petri Nets Used for Identification of Superheat Degree. *Int. J. Fuzzy Syst.* **2022**, *24*, 3431–3455. [CrossRef]
29. Murata, T. Petri nets: Properties, analysis and applications. *Proc. IEEE* **1989**, *77*, 541–580. [CrossRef]
30. Atanassov, K.; Andonov, V. Generalized nets and intuitionistic fuzzy pairs as tools for modelling of flexible manufacturing systems. *Notes Intuition. Fuzzy Sets* **2020**, *26*, 40–69. [CrossRef]
31. Atanassov, K. Generalized nets and intuitionistic fuzziness as tools for modeling of data mining processes and tools. *Notes Intuition. Fuzzy Sets* **2020**, *26*, 9–52. [CrossRef]
32. Orozova, D.; Hristova, N. Generalized net model for dynamic decision making and prognoses. In Proceedings of the IEEE International Symposium on Electrical Apparatus & Technologies, Burgas, Bulgaria, 3–6 June 2020; pp. 1–4.
33. Stratiev, D.; Dimitriev, A.; Stratiev, D.; Atanassov, K. Modeling the Production Process of Fuel Gas, LPG, Propylene, and Polypropylene in a Petroleum Refinery Using Generalized Nets. *Mathematics* **2023**, *11*, 3800. [CrossRef]
34. Boyukov, T.; Atanassov, K. Generalized Nets as a Tool for Modelling of Railway Networks: Part 2. In *Uncertainty and Imprecision in Decision Making and Decision Support: New Advances, Challenges, and Perspectives, Proceedings of the International Workshop on Intuitionistic Fuzzy Sets and Generalized Nets, Warsaw, Poland, 10–11 December 2020*; Springer International Publishing: Cham, Switzerland, 2020; pp. 120–128.
35. Stratiev, D.; Zoteva, D.; Atanassov, K. Modelling the process of production of automotive gasoline by the use of Generalized Nets. In *Uncertainty and Imprecision in Decision Making and Decision Support: New Advances, Challenges, and Perspectives, Proceedings of the International Workshop on Intuitionistic Fuzzy Sets and Generalized Nets, Warsaw, Poland, 10–11 December 2020*; Springer International Publishing: Cham, Switzerland, 2020; pp. 349–365.
36. Rawal, B.; Manogaran, G.; Peter, A. Firewalls. In *Cybersecurity and Identity Access Management*; Springer Nature: Singapore, 2022; pp. 117–128.
37. Valijonovich, T.; Safoev, N. A Brief Overview of Packet Classification Techniques in Computer Networks. *Tex. J. Eng. Technol.* **2023**, *18*, 60–62.
38. Coscia, A.; Dentamaro, V.; Galantucci, S.; Maci, A.; Pirlo, G. An innovative two-stage algorithm to optimize Firewall rule ordering. *Comput. Secur.* **2023**, *134*, 103423. [CrossRef]
39. Lyu, Y.; Feng, Y.; Sakurai, K. A Survey on Feature Selection Techniques Based on Filtering Methods for Cyber Attack Detection. *Information* **2023**, *14*, 191. [CrossRef]
40. Rajaboevich, G.; Dilbar, K.; Azatovna, A.; Ismoilovna, Q. Comparative Analysis of Methods Content Filtering Network Traffic. *Int. J. Emerg. Trends Eng. Res.* **2020**, *8*, 1561–1569. [CrossRef]
41. Kim, M. A Study on the Attack Index Packet Filtering Algorithm Based on Web Vulnerability. In *Big Data, Cloud Computing, and Data Science Engineering*; Springer International Publishing: Cham, Switzerland, 2023; pp. 145–152.
42. Kailanya, E.; Mwadulo, M.; Omamo, A. Dynamic deep stateful firewall packet analysis model. *Afr. J. Sci. Technol. Soc. Sci.* **2022**, *1*, 116–123. [CrossRef]
43. Hitchcock, K. Access Control. In *The Enterprise Linux Administrator: Journey to a New Linux Career*; Apress: Berkeley, CA, USA, 2022; pp. 161–192.
44. Sikos, L. Packet analysis for network forensics: A comprehensive survey. *Forensic Sci. Int. Digit. Investig.* **2020**, *32*, 200892.
45. Nife, F.; Kotulski, Z. Application-aware firewall mechanism for software defined networks. *J. Netw. Syst. Manag.* **2020**, *28*, 605–626.
46. Sundareswaran, N.; Sasirekha, S. Packet filtering mechanism to defend against DDoS attack in blockchain network. In *Evolutionary Computing and Mobile Sustainable Networks, Proceedings of the International conference on Evolutionary Computing and Mobile Sustainable Networks, Bangalore, India, 28–29 September 2021*; Springer: Singapore, 2022; pp. 201–214.
47. Abdulhassan, A.; Shubbar, R.; Alhisnawi, M. Cuckoo filter based IP packet filtering using M-tree. *Bull. Electr. Eng. Inform.* **2023**, *12*, 958–968. [CrossRef]
48. Sreelaja, N. A Fireworks-Based Approach for Efficient Packet Filtering in Firewall. In *Handbook of Research on Fireworks Algorithms and Swarm Intelligence*; IGI Global: Hershey, PA, USA, 2020; pp. 315–333.
49. Asai, H. PALMTRIE: A ternary key matching algorithm for IP packet filtering rules. In Proceedings of the 16th International Conference on Emerging Networking Experiments and Technologies, Barcelona, Spain, 1–4 December 2020; pp. 323–335.

50. Sičić, I.; Slovenec, K.; Petricioli, L.; Mikuc, M. Comparison of cuckoo hash table and bloom filter for fast packet filtering using data plane development kit. In Proceedings of the International Conference on Software, Telecommunications and Computer Networks, Split, Croatia, 19–21 September 2019; pp. 1–5.
51. Pradhan, P.; Mannepalli, P. Machine Leaning for Flow Based Intrusion Detection Using Extended Berkley Packet Filter. *Int. J. Eng. Res. Curr. Trends* **2021**, *3*, 5–7.
52. Cheng, J.; Li, C. Design and Implementation of TLS Traffic Packet Filtering Technology Based on Net filter Framework. In Proceedings of the International Conference on Cyber Security and Information Engineering, Brisbane, Australia, 23–25 September 2022; pp. 18–22.
53. Liang, J.; Chen, L.; Li, Z.; Bai, J. Container Network Performance Anomaly Detection Based on Extended Berkeley Packet Filter and Machine Learning. In *Advances in Natural Computation, Fuzzy Systems and Knowledge Discovery, Proceedings of the International Conference on Advances in Natural Computation, Fuzzy Systems and Knowledge Discovery, Guiyang, China, 24–26 July 2021*; Springer International Publishing: Berlin/Heidelberg, Germany, 2022; pp. 1403–1415.
54. Zhang, X.; Chen, L.; Bai, J. SYN Flood Attack Detection and Defense Method Based on Extended Berkeley Packet Filter. In *Advances in Natural Computation, Fuzzy Systems and Knowledge Discovery, Proceedings of the International Conference on Advances in Natural Computation, Fuzzy Systems and Knowledge Discovery, Guiyang, China, 24–26 July 2021*; Springer International Publishing: Berlin/Heidelberg, Germany, 2022; pp. 1416–1427.
55. Dayal, M.; Chawla, A.; Khari, M.; Mahajan, A. Artificial Intelligence-Based Smart Packet Filter. In *Proceedings of Third International Conference on Computing, Communications, and Cyber-Security*; Springer Nature: Singapore, 2022; pp. 791–801.
56. Fiessler, A.; Lorenz, C.; Hager, S.; Scheuermann, B.; Moore, A. Hypafilter+: Enhanced hybrid packet filtering using hardware assisted classification and header space analysis. *EEE/ACM Trans. Netw.* **2017**, *25*, 3655–3669. [CrossRef]
57. Lotfollahi, M.; Jafari, M.; Shirali, R.; Saberian, M. Deep packet: A novel approach for encrypted traffic classification using deep learning. *Soft Comput.* **2020**, *24*, 1999–2012.
58. Shin, Y.; Koo, D.; Hur, J. Inferring firewall rules by cache side-channel analysis in network function virtualization. In Proceedings of the International Conference on Computer Communications, Toronto, ON, Canada, 6–9 July 2020; pp. 1798–1807.
59. Li, W.; Meng, W.; Wang, Y.; Li, J. Enhancing Blacks List-Based Packet Filtration Using Blockchain in Wireless Sensor Networks. In *Wireless Algorithms, Systems, and Applications, Proceedings of the 16th International Conference on Wireless Algorithms, Systems, and Applications, Part II, Nanjing, China, 25–27 June 2021*; Springer International Publishing: Berlin/Heidelberg, Germany, 2021; pp. 624–635.
60. Peng, H.; Gao, D.; Yang, M.; Ma, J. An Efficient Firewall Application Using Learned Cuckoo Filter. In *Emerging Networking Architecture and Technologies, Proceedings of the International Conference on Emerging Networking Architecture and Technologies, Shenzhen, China, 15–17 November 2022*; Springer Nature: Singapore, 2023; pp. 430–441.
61. Botvinko, A.; Samouylov, K. Firewall Simulator Development for Performance Evaluation of Ranging a Filtration Rules Set. In Proceedings of the International Conference on Distributed Computer and Communication Networks, Moscow, Russia, 26–29 September 2022; Springer Nature: Cham, Switzerland, 2023; pp. 190–201.
62. Karthikeyan, S.; Keerthivasan, M.; Lalitha, A.; Karan, R. Network Intrusion Detection System Based on Packet Filters. *I-Manag. J. Comput. Sci.* **2021**, *9*, 27–32. [CrossRef]
63. Hussein, M. A Proposed Multi-Layer Firewall to Improve the Security of Software Defined Networks. *Int. J. Interact. Mob. Technol.* **2023**, *17*, 153–165. [CrossRef]
64. Putra, N.; Riyanto, V.; Wijaya, G.; Herlinawati, N. Firewall Design Using Access Control List Method as Data Filtering. *J. Mantik* **2021**, *5*, 1684–1693.
65. Ramprasath, J.; Seethalakshmi, V. Mitigation of malicious flooding in software defined networks using dynamic access control list. *Wirel. Pers. Commun.* **2021**, *121*, 107–125.
66. Yaibuates, M.; Chaisricharoen, R. A combination of ICMP and ARP for DHCP malicious attack identification. In Proceedings of the International Conference on Digital Arts, Media and Technology, Pattaya, Thailand, 11–14 March 2020; pp. 15–19.
67. Jaszcz, A.; Połap, D. AIMM: Artificial Intelligence Merged Methods for flood DDoS attacks detection. *J. King Saud Univ.-Comput. Inf. Sci.* **2022**, *34*, 8090–8101.
68. Shah, S.; Khan, F.; Ahmad, M. Mitigating TCP SYN flooding based EDOS attack in cloud computing environment using binomial distribution in SDN. *Comput. Commun.* **2022**, *182*, 198–211. [CrossRef]
69. Karunakaran, H.; Bhumireddy, V. Utilizing Neutrosophic Logic in the Design of a Smart Air-Conditioning System. *Appl. Sci.* **2022**, *12*, 9776.
70. Ouallane, A.; Broumi, S.; Ayele, E.; Bakali, A.; Bahnasse, A.; Talea, M. Towards Intelligent Road Traffic Management Based on Neutrosophic Logic: A Brief Review. *Neutrosophic Sets Syst.* **2022**, *51*, 7.
71. Kaur, G.; Garg, H. A novel algorithm for autonomous parking vehicles using adjustable probabilistic neutrosophic hesitant fuzzy set features. *Expert Syst. Appl.* **2023**, *226*, 120101.
72. Mısır, O. Dynamic local path planning method based on neutrosophic set theory for a mobile robot. *J. Braz. Soc. Mech. Sci. Eng.* **2023**, *45*, 127.
73. Pai, S.; Prabhu, R. Safety modelling of marine systems using neutrosophic logic. *J. Eng. Marit. Environ.* **2021**, *235*, 225–235.
74. Naik, N.; Jenkins, P. Enhancing windows firewall security using fuzzy reasoning. In Proceedings of the 2016 14th International Conference on Dependable, Autonomic and Secure Computing, Auckland, New Zealand, 8–12 August 2016; pp. 263–269.

75. Swapna, A.; Rahman, Z.; Rahman, M.; Akramuzzaman, M. Performance evaluation of fuzzy integrated firewall model for hybrid cloud based on packet utilization. In Proceedings of the IEEE International Conference on Computer Communication and the Internet, Wuhan, China, 13–15 October 2016; pp. 253–256.
76. Naik, N.; Jenkins, P. Fuzzy reasoning based windows firewall for preventing denial of service attack. In Proceedings of the IEEE International Conference on Fuzzy Systems, Vancouver, BC, Canada, 24–29 July 2016; pp. 759–766.
77. Naik, N.; Jenkins, P.; Kerby, B.; Sloane, J.; Yang, L. Fuzzy logic aided intelligent threat detection in cisco adaptive security appliance 5500 series firewalls. In Proceedings of the IEEE International Conference on Fuzzy Systems, Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8.
78. Gohain, B.; Chutia, R.; Dutta, P. A distance measure for optimistic viewpoint of the information in interval-valued intuitionistic fuzzy sets and its applications. *Eng. Appl. Artif. Intell.* **2023**, *119*, 105747.
79. Patel, A.; Jana, S.; Mahanta, J. Construction of similarity measure for intuitionistic fuzzy sets and its application in face recognition and software quality evaluation. *Expert Syst. Appl.* **2023**, *14*, 21491.
80. Dwivedi, A.; Kaliyaperumal, U.; Kuruvilla, J.; Thomas, A.; Shanthi, D.; Haldorai, A. Time-series data prediction problem analysis through multilayered intuitionistic fuzzy sets. *Soft Comput.* **2023**, *27*, 1663–1671. [PubMed]
81. Yue, Q.; Zou, W.; Hu, W. A new theory of triangular intuitionistic fuzzy sets to solve the two-sided matching problem. *Alex. Eng. J.* **2023**, *63*, 57–73.
82. Yazdi, M.; Kabir, S.; Kumar, M.; Ghafir, I.; Islam, F. Reliability Analysis of Process Systems Using Intuitionistic Fuzzy Set Theory. In *Advances in Reliability, Failure and Risk Analysis*; Springer Nature: Singapore, 2023; pp. 215–250.
83. Dawadi, B.; Adhikari, B.; Srivastava, D. Deep Learning Technique-Enabled Web Application Firewall for the Detection of Web Attacks. *Sensors* **2023**, *23*, 2073. [CrossRef]
84. Liang, H.; Li, X.; Xiao, D.; Liu, J.; Zhou, Y.; Wang, A.; Li, J. Generative Pre-trained Transformer-Based Reinforcement Learning for Testing Web Application Firewalls. *IEEE Trans. Dependable Secur. Comput.* **2023**, 1–25. [CrossRef]
85. Sepczuk, M. Dynamic Web Application Firewall detection supported by Cyber Mimic Defense approach. *J. Netw. Comput. Appl.* **2023**, *213*, 103596.
86. Li, J.; Fan, Y.; Bian, X.; Yuan, Q. Online/Offline MA-CP-ABE with Cryptographic Reverse Firewalls for IoT. *Entropy* **2023**, *25*, 616.
87. Tudosi, A.; Graur, A.; Balan, D.; Potorac, A. Research on Security Weakness Using Penetration Testing in a Distributed Firewall. *Sensors* **2023**, *23*, 2683. [CrossRef]
88. Botvinko, A.; Samouylov, K. Firewall simulation model with filtering rules ranking. In Proceedings of the Distributed Computer and Communication Networks: Control, Computation, Communications, Moscow, Russia, 14–18 September 2020; Springer International Publishing: Berlin/Heidelberg, Germany, 2020; pp. 533–545.
89. Wang, C. Construction and Deployment of a Distributed Firewall-based Computer Security Defense Network. *Int. J. Netw. Secur.* **2023**, *25*, 89–94.
90. Chakir, O.; Sadqi, Y.; Maleh, Y. Evaluation of Open-source Web Application Firewalls for Cyber Threat Intelligence. In *Big Data Analytics and Intelligent Systems for Cyber Threat Intelligence*; River Publishers: Abingdon, UK, 2023; pp. 35–48.
91. Islam, M.; Uddin, M.; Hossain, D.; Dulal, M.; Ahmed, D.; Shakil, M.; Moazzam, D.; Golam, M. Analysis and Evaluation of Network and Application Security Based on Next Generation Firewall. *Int. J. Comput. Digit. Syst.* **2023**, *13*, 193–202. [CrossRef]
92. Lar, S.; Liao, X.; Rehman, A.; Qinglu, M. Proactive Security Mechanism and Design for Firewall. *J. Inf. Secur.* **2011**, *2*, 122–130. [CrossRef]